International Journal of Multimedia and Ubiquitous Engineering
Vol.10, No.4 (2015), pp.141-154
http://dx.doi.org/10.14257/ijmue.2015.10.4.15

Algorithmic Based and Non-Algorithmic Based Approaches to Estimate the Software Effort

WanJiang Han[1], TianBo Lu[1], XiaoYan Zhang[1], LiXin Jiang[2] and Weijian Li

[1]*School of Software Engineering, Beijing University of Posts and Telecommunication, Beijing 100876, China*
[2]*Department of Emergency Response, China Earthquake Networks Center, Beijing 100036, China*
hanwanjiang@bupt.edu.cn

Abstract

Software effort estimation is the key task for the effective project management. It is widely used for planning and monitoring software project development as a means to deliver the product on time and within budget. So far, no model has been proved to be successful at effectively and accurately estimating software development effort. So it is useful to research a particular model for a particular type of project. This paper present an approach for small organic project, based on our previous work. Besides using Gauss-Newton model to calibrate the parameters of the COCOMO, using Fuzzy logic algorithm to optimize it, we also imply Deming Regression, Expert judgment, and Machine learning to improve this model. This model is based on historical project data. Experimental results show that the model is effective for software estimation. The accuracy comparison of each model is presented.

Keywords: *Software effort estimation, Deming regression, Fuzzy, Gauss–Newton*

1. Introduction

With the rapid development of networks and information technology, software projects have become widely used and increasingly powerful in all fields. Software effort estimation is one of the important tasks for software project management. Several approaches for generating predictive models from collected metrics have been proposed throughout the years. But, it is very hard to estimate software development effort accurately.

Conventional approaches to software cost estimation have focused on algorithmic cost models, where an estimate of effort is calculated from one or more numerical inputs via a mathematical model. Analogy-based estimation has recently emerged as a promising approach, with comparable accuracy to algorithmic methods in some studies, and it is potentially easier to understand and apply.

Accurate software estimation can provide powerful assistance when software management decisions are being made. For instance, accurate cost estimation can help an organization to better analyze the feasibility of a project and to effectively manage the software development process, therefore, greatly reducing the risk.

Lots of attempts [1-7] have been made to solve the problem in the last few decades, no approach has proven to be successful in effectively and consistently predicting software effort.

So it is useful to research a model for particular type of project. This paper offers a new approach to estimate the software effort for organic project based on the data of historical projects. We have taken into consideration the features of the effort estimation problem

Manuscript received October 2, 2012; revised March 7, 2013.

ISSN: 1975-0080 IJMUE

and some techniques. We improved our previous job [1] which applied Gauss-Newton model and Fuzzy model to the COCOMO, and have validated with later project .

In this paper, we explore the possibility of improving the process of modeling. Gauss-Newton algorithm, Deming Regression, Expert judgment and Machine learning are used in our model, meanwhile we have given a comparison between them.

2. Related Work

Estimating effort involving expert judgment is one of the most common approach [8-9]. There are two major approaches to prediction using expert judgment, namely, top-down and bottom-up [10]. In top down approach, effort estimate is based on properties of the project as a whole and distributed over project activities. Whereas in bottom-up approach, effort is calculated as a sum of the project activities estimate. Bottom-up technique provides relatively more accurate estimates but is more time consuming.

Prediction by analogy is another common approach for effort estimation and is an application of a Case-Based Reasoning (CBR) approach [11-12]. It is based on the claim that ‘‘Project efforts to be estimated will probably behave like efforts of similar past projects’’[12]. Prediction using analogy involves characterizing the project for which an estimate is required. This characterization then forms the basis for finding similar or analogous completed projects for which effort is known. These effort values are then used, possibly with adjustment, to generate the predicted value.

Algorithmic models are one of the most widely used and experimented model in the research and practice [13]. They are widely used in the industry as well. There are many algorithmic models proposed in the literature among them the popular ones are COCOMO, SLIM, and SEER-SEM models [14-16]. Effort is estimated as a mathematical function of product, project and process attributes whose values are generally estimated by the project managers. There is plenty of literature discussing algorithmic models specially the COCOMO based models. Interested readers can refer to [14].

Non-algorithmic approaches that are based on machine learning and soft computing are relatively newer area of research [17-18]. These include Bayesian belief networks [19], fuzzy logic [20-22], artificial neural networks [23] and evolutionary computation [24-25]. There are some work using more than one soft computing technique in their prediction models like neuro-fuzzy and neuro-genetic [26-28].

The other area of research is to come up with software size metrics to estimate the size of software to be developed. There are a number of size metrics proposed in the literature to estimate the size of software to be developed [28].

Our approaches are related with the standard COCOMO model, Gauss–newton algorithm, Deming Regression and fuzzy logic model, we briefly review these techniques.

2.1. COCOMO Model

The COCOMO model originally published by Boehm is one of most popular parametric cost estimation models of the 1980s [1], [29]. At present, the model is still the most important in the software field. In the middle of 1990’s, Boehm proposed COCOMO II [2-3] based on COCOMO81. Nowadays, it is considered as one of the most extensively used and approved software estimating model in academia and industrial area. The basic principle of COCOMO model is to express effort with software size and a series of cost factor, as the following equation:

$$PM = A \times (\sum Size)^{\sum_B} \times \prod (EM) \tag{1}$$

2.2. Gauss–Newton Algorithm

Gauss–Newton algorithm is a method used to solve non-linear least squares problems. The method is named after the mathematicians Carl Friedrich Gauss and Isaac Newton [29-30].

Non-linear least squares problems arise for instance in non-linear regression, where parameters in a model are sought such that the model is in good agreement with available observations.

Given m functions r = (r1,…,rm) of n variables β= (β1,…,βn), with m ≥ n, the Gauss–Newton algorithm finds the minimum of the sum of squares, as in (2).

$$S(\beta) = \sum_{i=1}^{m} r_i^2(\beta) \tag{2}$$

Starting with an initial guess β (0) for the minimum, the method proceeds by the iterations, as in (3).

$$\beta^{(s+1)} = \beta^{(s)} + \Delta, \tag{3}$$

where

$$S(\beta^{(S)} + \Delta) = S(\beta^{(S)}) + \left[\frac{\partial S}{\partial \beta_i}\right]\Delta + \frac{1}{2}\Delta^{T}\left[\frac{\partial^2 S(\beta)}{\partial \beta_i \partial \beta_j}\right]\Delta$$

Δ is a small step. We then have.

If we define the Jacobian matrix as in (4).

$$J_r(\beta) = \frac{\partial r_i}{\partial \beta_j}\Big|_{\beta}, \tag{4}$$

we can replace $\left[\frac{\partial S}{\partial \beta_i}\right]$ with $J_r^{T} r$

and the Hessian matrix can be approximated by (5).

$$S(\beta^{(S)} + \Delta) \approx S(\beta^{(S)}) + J_r^{T} r\Delta + \frac{1}{2}\Delta^{T} J_r^{T} J_r \Delta \tag{5}$$

(assuming small residual), giving: $J_r^{T} J_r$ .We then take the derivative with respect to Δ and set it equal to zero to find a solution as in (6).

$$S'(\beta^{(S)} + \Delta) \approx J_r^{T} r + J_r^{T} J_r \Delta = 0 \tag{6}$$

This can be rearranged to give the normal equations which can be solved for Δ as in (7).

$$(J_r^{T} J_r)\Delta = -J_r^{T} J_r \tag{7}$$

In data fitting, where the goal is to find the parameters β such that a given model function y=f(x, β) fits best some data points (xi, yi), the functions ri are the residuals.

$$r_i(\beta) = y_i - f(x_i, \beta)$$

Then, the increment Δ can be expressed in terms of the Jacobian of the function f, as in (8).

$$(J_f^{T} J_f)\Delta = J_f^{T} r \tag{8}$$

2.3. Deming Regression

Deming regression is a form of errors-in-variables model which tries to fit the best line on a dataset assuming measurement errors for both sets of measurements [31-32].

Let us assume that the available historical data are erroneously measured observations of the true, but unknown. The measured value is likely to deviate from the true value by

some small "random" amount. For a given dataset (x_i, y_i), the equations describing the model are:

$$x_i = X_i + \varepsilon_i \tag{9}$$

$$y_i = Y_i + \delta_i \tag{10}$$

Where error terms ε_i and δ_i are assumed to be independent normal variables with zero mean values. In order to estimate the regression line by the Deming methodology, it is necessary to evaluate or assign a value to the ratio λ of the variances of ε and δ for the x and y, respectively.

$$\lambda = \frac{S_{\varepsilon_x}^2}{S_{\delta_y}^2} \tag{11}$$

The objective of Deming regression is to find the best fitting line

$$Y_i = \beta_0 + \beta_1 X_i \tag{12}$$

such that the weighted SSR of the model is minimized.

$$SSR = \sum_{i=1}^{n} \left(\frac{\varepsilon_i^2}{S_{\varepsilon_x}^2} + \frac{\delta_i^2}{S_{\delta_y}^2}\right) = \sum_{i=1}^{n} \left((y_i - \beta_0 - \beta_1 X_i)^2 + \lambda (x_i - X_i)^2\right) \tag{13}$$

The value of λ determines the angle in which the points are projected onto the line in order to minimize SSR [33]. By setting λ=1, the results of Deming regression is equal to the results of orthogonal regression which takes into account the distance of each data point from the line .The final estimates of the parameters by Deming regression are given by

$$\hat{\beta} = \frac{s_{yy} - \lambda s_{xx} + \sqrt{(s_{yy} - \lambda s_{xx}) + 4\lambda s^2 xy}}{2 s_{xy}} \tag{14}$$

$$\hat{\beta}_0 = \bar{y} - \hat{\beta}_1 \bar{x} \tag{15}$$

$$X_i = x_i + \frac{\hat{\beta}_1}{\hat{\beta}_1^2 + \lambda} (y_i - \hat{\beta}_0 - \hat{\beta}_1 x_i) \tag{16}$$

Where

$$s_{xx} = \frac{1}{n-1} \sum_{i=1}^{n} (x_i - \bar{x})^2 \tag{17}$$

$$s_{yy} = \frac{1}{n-1} \sum_{i=1}^{n} (y_i - \bar{y})^2 \tag{18}$$

$$s_{xy} = \frac{1}{n-1} \sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y}) \tag{19}$$

It must be noted that the value of λ is not known in advance and it can be estimated only by designing a study where multiple measurements of the same projects are taken and these measurements vary. Since in historical data such information is not available, we take in our applications the simplest orthogonal case, λ=1 which usually gives better results.

2.4. Expert Judgment

Expert opinion can be defined as the judgment of an individual expert or group of experts with respect to a specific subject or unknown measure.

Expert judgment has been considered particularly useful in 1) areas where empirical data is not easily available, and 2)when estimating complex, ill defined or poorly understood problems. Both these dimensions can be seen as underlying reasons why expert judgment is widely used as an approach to software estimation [34-35].

2.5. Machine Learning

We will briefly describe the main characteristics of the machine learning algorithms compared [36-41].

Linear regression (LR) is a method to model the linear relationship between a scalar dependent variable y and one or more independent variables x.

Least Median Square (LMS) method is one of the statistical methods for solving the equations which are more than unknown. This method almost used in analytical regression. In fact, Least Median Square is a method for fitting the dataset. The least Median Square must yield the smallest value for the median of squared residuals computed for the entire data set. It means the residuals, the difference between real data and predicted data.

M5P is a tree learner and consists of binary decision tree which learns a "model" tree. This is a decision tree with linear regression functions at the leaves. It is used to predict a numeric target attribute. It may be piecewise fit to the target [40-41].

REPtree (RT) Builds a decision tree using information gain and prunes it using reduced-error pruning (with back fitting). Only sorts values for numeric attributes once. Missing values are dealt with by splitting the corresponding instances into pieces [41].

Sequential minimal optimization (SMO) [36-37] was proposed as an iterative algorithm. This algorithm uses SVM for solving regression problem and SVM classifier design. The SMO algorithm has two worthy aspects: implementation is easy and computational speed is fast [38].

The multilayer perceptron (MLP) is a very simple model of biological neural networks. The network is structured in a hierarchical way. Multilayer Perceptron includes some nodes located in different layers where the information flows only from one layer to the next layer. The first and the last layers are the input and output. Layers between the input and output layer are called hidden layers. This network is trained based on back propagation error in that the real outputs are compared with network outputs, and the weights are set using supervised back propagation to achieve the suitable model [42].

2.6. Fuzzy Logic Models

A fuzzy system [43] is a mapping between linguistic terms, such as ‘‘high complexity’’ and ‘‘low cost’’ that are attached to variables. Thus an input into a fuzzy system can be either numerical or linguistic with the same applying to the output. A typical fuzzy system is made up of three major components: fuzzifier, fuzzy inference engine (fuzzy rules) and defuzzifier. The fuzzifier transforms the input into linguistic terms using membership functions that represent how much a given numerical value of a particular variable fits the linguistic term being considered. The fuzzy inference engine performs the mapping between the input membership functions and the output membership functions using

fuzzy rules that can be obtained from expert knowledge. The greater the input membership degree, the stronger the rule fires, thus the stronger the pull towards the output membership function.

Triangular fuzzy numbers are a subset of fuzzy sets with properties that make them well suited for modeling and design-type activities. Specifically, it has a triangular shape represented by the triple <a, b, c>,like Figure1.

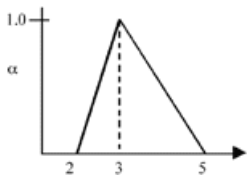

Figure 1.Triangular Fuzzy Numbers

3. Algorithmic Based Estimation Model

The estimation model for small organic project is a process that takes Line of Codes (LOC) as inputs in order to estimate the workloads. So our goal is to fit a curve to data from actual software projects.

3.1. Gauss–Newton based Model

The results of research and practice show that the relationship between Code Line and effort is nonlinear [21-22]. And from the trend of the data fitting curve, we also got that non-linear regression analysis is fit for our model. Hence, we applied nonlinear regression fitting method to form our model in this paper. We tried to gain an equation as a relation function between LOC and effort, shown as in (20).

$$y = f(x) \tag{20}$$

On the other hand, basing on COCOMO 81 model, we adjusted the form of the model, shown as in (21).

$$y = a \times x^b \tag{21}$$

In the Equation (21), y is represented for the human resource needed (person hour), x is represented for Code Lines, “a” and “b” are both parameters.

Now let’s look at the project data, as in Table 1, which are for the type of small organic project.

Table 1. Historical Project Data

No.	1	2	3	4	5	6	7	8	9
LOC	41	132	144	176	194	255	291	378	591
Person-Hour	6	10	11	16	22	30	32	35	42

It is desired to find a model function of the form of (21), that is to say, the model function is $PH = a \times LOC^b$,which fits best the data in the least squares sense, with the parameters a and b to be determined.

Denote by x_i and y_i the value of LOC and the Person-Hour from the Table 1, i=1,…,9. We will find a and b such that the sum of squares of the residuals,

$r_i = y_i - a \times x_i^b$, (i=1,…,9) is minimized.

After seven iterations of the Gauss–Newton algorithm the optimal values a=0.3709 and b=0.7547 are obtained. The plot in Figure 2 shows the curve determined by the model for the optimal parameters versus the observed data.

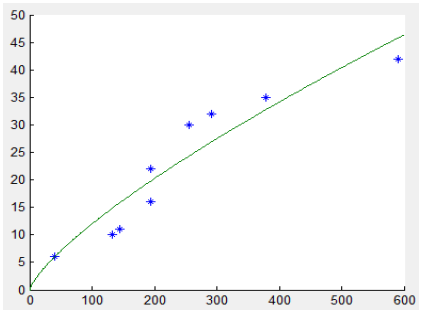

Figure 2. LOC-Person Hour

So, we have achieved the expression of our estimation model as follows:

$$E = 0.3709 \times LOC^{0.7547} \quad (22)$$

Where E is the effort of Person Hour, and LOC is the size of Line of Codes.

3.2. Deming-Regression based Approach

In this section, we present the results of Deming regressions described earlier as applied to our datasets. The building process is just like Gauss–Newton based [44-46].

As a result, both the dependent effort and size variables are transformed to a natural logarithmic scale providing the new variables $\ln E$ and $\ln LOC$, respectively. Finally, the form of Deming regressions is given in Eq. (23).

$$\ln E = a_0 + a_1 \ln LOC \quad (23)$$

Based on the estimated parameters, we construct the regression lines, shown in (24)

$$E = 0.511 \times LOC^{0.653} \quad (24)$$

4. The Multiplier of Model

In order to improve estimating accuracy of our model, we also consider the factors that impact the effort. Therefore, the equation of our model can be adjusted into (25).

$$E_{new} = E \times F \quad (25)$$

In Equation (25), F is a multiplier which is a correction factor to our model. And F is expressed as in (26).

$$F = \sum_{i=1}^{5} (w_i \times CD_i) \times \prod_{i=1}^{5} CD_i \quad (26)$$

In Equation (26), CD means Cost Driver, w is the Weight of Cost Driver. According the character of small organic project, we choose five cost drivers, which are PREX (experience), PERS (skill or capability), RCPX (reliability and complexity), PDIF (platform difficulty), SCED (Required development schedule), referred to cost drivers of CocoMo II. Each cost driver represents one factor that contributes to the development effort. We use “CD1”, “CD2”, “CD3”, “CD4”, “CD5” to represent “PREX”, “PERS”, “RCPX”, “PDIF”, “SCED’’, respectively.

Since the importance of every cost driver is different, we maintain the merits of correction factor of COCOMO model.

Rating of every cost driver is linguistic terms such as “very low”, “low”, “nominal”, “high”, “very high”, “extra high”, and the value of every rating can get by referring COCOMO II model.

w is the ratio of these cost driver, also a key of fuzzy evaluation. When we compare with each of the two cost driver, it is difficult to describe their importance by number, so we use the Triangular Fuzzy Number, by the sequence from high to low to ensure the weights of each cost driver [47-48].

Step 1: For these cost drivers, considering the importance for them, there should be PERS (CD2) > PREX (CD1)> RCPX (CD3)> PDIF (CD4)> SCED (CD5). Here comes to the Triangular fuzzy judgment matrix, like Table 2 .

Table 2. Triangular Fuzzy Judgment Matrix of These Cost Driver Project Data

CD	CD1	CD2	CD3	CD4	CD5
CD1	(1,1,1)	(1/3,1/2,1)	(2,3,4)	(3,4,5)	(4,5,6)
CD2	(1,2,3)	(1,1,1)	(3,4,5)	(4,5,6)	(5,6,7)
CD3	(1/4,1/3,1/2)	(1/5,1/4.1/3)	(1,1,1)	(2,3,4)	(3,4,5)
CD4	(1/5,1/4,1/3)	(1/6,1/5,1/4)	(1/4,1/3,1/2)	(1,1,1)	(2,3,4)
CD5	(1/6,1/5,1/4)	(1/7,1/6,1/5)	(1/5,1/4,1/3)	(1/4,1/3,1/2)	(1,1,1)

Step 2: Computing the weights of cost driver.

(1) From Table 2,we can get the triangular fuzzy weight vector:

w1= (0.29, 0.29, 0.30),
w2= (0.43, 0.44, 0.42),
w3= (0.15, 0.15, 0.15),
w4= (0.082, 0.081, 0.080),
w5= (0.049, 0.045, 0.045),

(2) For every row, Adding each element , we can get :

s1= (2.61, 3.78, 5.08),
s2= (1.84, 1.95, 2.78),
s3= (6.45,8.58,10.83),
s4= (10.25, 13.33, 16.5),
s5= (15, 19, 23)

(3) Calculating the maximum Eigen value vector λmax , we can get the Max Eigen value $E(\lambda max)$= 5.27.

(4) Consistency checking

CI=0.067, CR=0.054< 0. 1, And E (λmax) < order critical maximum eigenvalue. Hence, this Triangular fuzzy matrix satisfy the consistency check.

Step 3: Calculating the Triangular fuzzy weight vectors expectations value
E(w1) =0.30,
E(w 2) = 0.43,
E(w 3) = 0.15,

E(w 4) = 0.08,
E(w 5) = 0.04,

From these expectations value, weights of each cost driver can be shown as in Table 3.

Table 3. Weights of Cost Diver

Cost driver	PREX	PERS	RCPX	PDIF	SCED
w	0.30	0.43	0.15	0.08	0.04

Above all, our estimation model for organic software project is shown as (27) or (28).

$$E = 0.3709 \times LOC^{0.7547} \times \sum_{i=1}^{5} (w_i \times CD_i) \times \prod_{i=1}^{5} CD_i \quad (27)$$

$$E = 0.511 \times LOC^{0.653} \times \sum_{i=1}^{5} (w_i \times CD_i) \times \prod_{i=1}^{5} CD_i \quad (28)$$

Where (27) is based on Gauss–Newton and (28) is based on Deming regressions. We can get w_i from Table 3, CD_i from COCOMO Ⅱ model.

5. Evaluation Criteria

In order to evaluate these models and the fitting accuracy of Deming regression in comparison to that of Gauss–Newton based Model, we use the common evaluation criteria in the field of software cost estimation. They are Magnitude Relative Error (MRE), Mean magnitude relative error (MMRE) and MdMRE [49-51].

1) Magnitude Relative Error (MRE) computes the absolute percentage of error between actual and predicted effort for each reference project.

$$MRE_i = \frac{Actual_i - Estimated_i}{Actual_i}$$

2) Mean magnitude relative error (MMRE) calculates the average of MRE over all reference projects. Despite of the wide use of MMRE in estimation accuracy, there has been a substantive discussion about the efficacy of MMRE in estimation process. MMRE has been criticized that is unbalanced in many validation circumstances and leads often to overestimation (Shepperd and Schofield, 1997). Moreover, MMRE is not always reliable to compare between prediction methods because it is quite related to the measure of MRE spread (Foss et al., 2003).

$$\mathrm{MMRE} = \frac{1}{\mathrm{n}} \sum_{i=1}^{n} MRE_i$$

3) MMRE is sensitive to individual predictions with excessively large MREs. The median of MREs (MdMRE), is less sensitive to extreme values and is used as another measure [52]. We adopt median of MREs for the n projects (MdMRE) which is less sensitive to the extreme values of MRE.

$$\mathrm{MdMRE} = \underset{i}{\mathrm{median}}\, MRE_i$$

6. Experimental Results

In this section, we conduct a series experiments to evaluate the approaches, and give the comparison among approaches.

We collected 10 historical project data for our experiments, described in Table 4.

Table 4. Experimental Data

Project	*Staff*	*KLOC*	*Effort*
1	*8*	*300*	*2000*
2	*6*	*250*	*300*
3	*9*	*520*	*600*
4	*12*	*890*	*690*
5	*10*	*780*	*600*
6	*7*	*420*	*520*
7	*8*	*450*	*390*
8	*16*	*890*	*2100*
9	*15*	*560*	*2200*
10	*17*	*670*	*2300*

First, we describe one of software project. For instance, a case of organic project, the input is 500 LOC, we estimate the effort by (21) will be 40.3872 PH before adaptation, shows in Fig. 3. Then we take correction to the value with cost drivers. The values of each factor are shown below: PREX =1 (nominal), PERS =1.1 (low), RCPX =1.3 (high), PDIF =1 (nominal), SCED =1 (nominal), that is to say, F=0.903. So, the final result of estimation is Effort=36.5 Person-Hours. In this case, the project actually took 35 Person-Hours, MRE=9.6%, this value basically fits the evaluation value from the model.

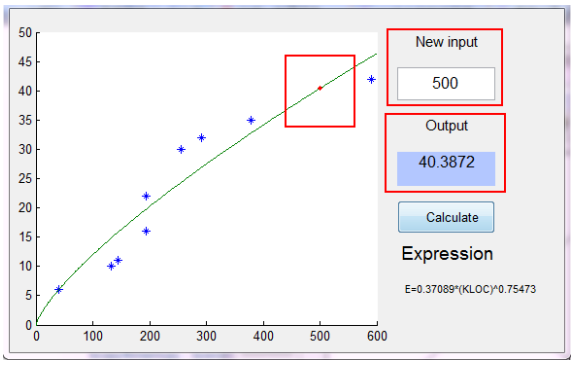

Figure 3. A Case of Software Estimation

On the other hand, we also applied (22) to estimate the effort of this case, MRE=8.9%.Then using Expert judgment based , the MRE is 9.5%.Then using Machine learning based, the MRE is 9.9%.

The following provides a summarized view of the results with table 4. We calculated out that MMRE and MdMRE using these four approaches, shown in Table 5. As we can see from Table 5, the Deming regressions based model seems to be fitted better than others.

Table 5. Performance Figures of Comparison

Model	MMRE	MdMRE
Gauss–Newton	9.1%	9.0%
Deming regressions based	8.8%	8.6%

Expert based	9.3	9.1
Machine learning based	9.5	10.2

7. Conclusions

Software development is a complex process, so is it very difficult to predict software development effort accurately. This paper presents a software effort estimation approach for small organic project, which is based on my previous work. This approach is based on actual project data and well-established theories. And our model used algorithms of Gauss-Newton, Deming Regression, Expert judgment and Machine learning. This paper gives a comparison between them. Result shows that Deming Regression based approach is more accurate than others. In particular, this model has been successfully used in some organic project, and has demonstrated great potential to predict software cost more accurately. So our approach is more suitable to this type of project. It can provide assistance in project planning.

As future work, we plan to integrate the search autonomously, to improve the algorithm and the search for solutions. Also, we will investigate more model for other type of project.

Acknowledgements

This work was supported in part by the National Natural Science Foundation of China (Grant No. 61170273).

References

[1] W.-J. Han, T.-B. Lu, and X.-Y. Zhang, "A New Estimation Model for Small Organic Software Project", 2013.9, Journal of Software, vol. 8, no. 9, (**2013**), pp. 2218-2222.
[2] B. W. Boehm, "Software Engineering Economics", Prentice Hall PTR, (**1981**).
[3] B. W. Boehm, "Software Cost Estimation with COCOMOII", Prentice Hall, (**2000**).
[4] S. Chulani, "Bayesian analysis of software cost and quality models," Ph.D. Dissertation, University of Southern California, Los Angeles, (**1999**)0.
[5] M. Shepperd and G. Kadoda, "Comparing software prediction techniques using simulation," IEEE Trans. Software Eng, vol. 27, no. 11, (**1999**) November, pp. 1014–1022.
[6] A. B. Nassif, L. F. Capretz, and D. Ho, "Software estimation in the early stages of the software life cycle," in International Conference on Emerging Trends in Computer Science, Communication and Information Technology, (**2010**), pp. 5-13.
[7] R. J. Madachy, "Heuristic risk assessment using cost factors," IEEE Software, vol. 14, no. 3, May/June (**1997**), pp. 51–59.
[8] M. Jorgensen, "Practical Guidelines for Expert-Judgment-Based Software Effort Estimation", IEEE Software, vol. 22, no. 3, (**2005**), pp. 57–63.
[9] M. Jorgensen and M. Shepperd, "A systematic review of software development cost estimation studies", IEEE Transactions on Software Engineering, vol. 33, no. 1, (**2007**), pp. 33–53.
[10] M. Jorgensen, "Top-down and bottom-up expert estimation of software development effort", Journal of Information and Software Technology, vol. 46, no. 1, (**2004**), pp. 3–16.
[11] J. Li, G. Ruhe, A. Al-Emran and M.M. Richter, "A flexible method for software effort estimation by analogy", Journal of Empirical Software Engineering, vol. 12, no. 1, (**2007**).
[12] M. Shepper and C. Schofield, "Estimating software project effort using analogies", IEEE Transactions on Software Engineering, vol. 23, no. 11, (**1997**), pp. 736–743.
[13] B. Boehm, C. Abts and S. Chulani, "Software development cost estimation approaches – a survey", in: J.C. Baltzer (Ed.), Annals of Software Engineering,vol. 10, Science Publishers, (**2000**), pp. 177–205.
[14] B. Boehm, C. Abts, A. Brown, S. Chulani, B. Clark, E. Horowitz, R. Madachy and D.Reifer, "Software Cost Estimation With COCOMO II", Prentice Hall, (**2000**).
[15] D. Galorath, "SEER-SEM", <http://www.galorath.com/>.
[16] L.H. Putnam, "A general empirical solution to the macro software sizing and estimating problem", IEEE Transactions on Software Engineering, vol. 4, (1**978**), pp. 345–361.
[17] M.O. Saliu, M. Ahmed and J. AlGhamdi, "Towards adaptive soft computing based software effort prediction", in: IEEE Annual Meeting of the Fuzzy Information,2004. Processing NAFIPS'04, vol. 1, (**2004**) 27–30 June, pp. 16–21.

[18] E. Kocaguneli, A. Tosun and A. Bener, "AI-based models for software effort estimation", in: 36th EUROMICRO Conference on Software Engineering and Advanced Applications (SEAA), (**2010**), pp. 323–326.
[19] K. Hamdan, S. Bibi, L. Angelis and I. Stamelos, "A bayesian belief network cost estimation model that incorporates cultural and project leadership factors", IEEE Symposium on Industrial Electronics & Applications (**2009**), pp. 985–989.
[20] M. W. Nisar, W. Yong-Ji and M. Elahi, "Software development effort estimation using fuzzy logic – a survey", in: Fifth International Conference on Fuzzy Systems and Knowledge Discovery, FSKD '08, (**2008**), pp. 421–427.
[21] Z. Muzaffar and M. Ahmed, "Software development effort prediction: a study on the factors impacting the accuracy of fuzzy logic systems", Journal of Information and Software Technology (IST), vol. 52, no. 1, pp. 92–109. http://dx.doi.org/10.1016/ j.infsof.2009.08.001.
[22] M. Ahmed and Z. Muzaffar, "Handling imprecision and uncertainty in software development effort prediction: a type-2 fuzzy logic based framework", Journal of Information and Software Technology (IST), vol. 51, no. 3, (**2008**), pp. 640–654, http:// dx.doi.org/10.1016/j.infsof.2008.09.004.
[23] H. Park and S. Baek, "An empirical validation of a neural network model for software effort estimation", Journal of Expert Systems with Applications, vol. 35, no. 3, (**2008**), pp. 929–937.
[24] S. L. Pfleeger, F. Wu and R. Lewis, "Software Cost Estimation and Sizing Methods", Issues and Guidelines, RAND Project Air Force, (**2005**).
[25] M. O. Saliu and M. Ahmed, "Soft computing based effort prediction systems – a survey", in: E. Damiani, L.C. Jain, M. Madravio (Eds.), Soft Computing in Software Engineering, Springer-Verlag Publisher, (**2004**).
[26] J. Wong, D. Ho and L. E. Capretz, "An investigation of using Neuro-Fuzzy with software size estimation", in: ICSE Workshop on Software Quality, WOSQ'09, (**2009**), pp. 51–58.
[27] K. K. Shukla, "Neuro-genetic prediction of software development effort", Information & Software Technology, vol. 42, no. 10, (**2000**), pp. 701–713.
[28] M. A. Ahmed, I. Ahmad and J. S. AlGham, "Probabilistic size proxy for software effort prediction: A framework", Information and Software Technology, vol. 55, (**2013**), pp. 241–251.
[29] P. K. Subramanian, "Gauss-Newton methods for the complementarity problem", Journal of Optimization Theory and Applications, vol. 77, no. 3, (**1993**) June, pp. 467-482.
[30] M. Y. Yerina and A. F. Izmailov, "The Gauss-Newton method for finding singular solutions to systems of nonlinear equations", Computational Mathematics and Mathematical Physics, vol. 47, no. 5, (**2007**) May, pp. 748-759.
[31] W. Deming, "Statistical adjustment of data", Wiley, NY (Dover Publications edition, 1985) (**1943**).
[32] N. Mittas, "Modeling the Relationship between Software Effort and Size Using Deming Regression", PROMISE2010, (**2010**) September 12-13. Timisoara, Romania.
[33] K. Linnet, "Performance of Deming regression analysis in case of misspecified analytical error ratio in method comparison studies", Clin Chem, vol. 44, no. 5, (**1998**), pp. 1024–1031.
[34] S.-W. Lin and V. M. Bier, "A study of expert overconfidence," Reliability Engineering & System Safety, vol. 93, no. 5, (**2008**) May, pp. 711-721.
[35] P. Faria and E. Miranda, "Expert Judgment in Software Estimation during the Bid Phase of a Project – An Exploratory Survey", 2012 Joint Conference of the 22nd International Workshop on Software Measurement and the 2012 Seventh International Conference on Software Process and Product Measurement.
[36] A. J.Smola and B.Schölkopf, "A Tutorial on Support Vector Regression", Royal Holloway College, London, U.K., NeuroCOLT Tech. Rep.TR 1998-030, (**1998**).
[37] S. K. Shevade, S. S. Keerthi, C. Bhattacharyya and K. R. K. Murthy, "Improvements to the SMO Algorithm for SVM Regression", IEEE TRANSACTIONS ON NEURAL NETWORKS, vol. 11, no. 5, (**2000**) September.
[38] D. J. C. Mackay, "Introduction to Gaussian Processes", Dept. of Physics, Cambridge University, UK, (**1998**).
[39] Y. Wang, and I. H. Witten, "Inducing Model Trees for Continuous Classes", the European Conf. on Machine Learning Poster Papers, (**1997**).
[40] Weka 3.6.8 http://www.cs.waikato.ac.nzlmllwekal.
[41] H. Ian and E. Frank, "Data Mining Practical Machine Learning Tools and Techniques", Morgan Kaufmann Publishers Is an Imprint of Elsevier, (**2005**), pp. 187-427.
[42] A. Macedo, T. B. Ferreira and R. Matias, "The Mechanics of Memory-Related Software Aging" , Second IEEE International Workshop on Software Aging and Rejuvenation, (**2010**), November 2.
[43] R. Fuller, "Introduction to Neuro-Fuzzy Systems", Physica-Verlag, (**2000**).
[44] E. Castillo, A. S. Hadi, and R. Minguez, "Diagnostics for non-linear regression, Department of Applied Mathematics and Computational Sciences, University of Cantabria, Santander; Spain, University of Castilla-La Mancha, Ciudad Real, Hadi; Minguez," Journal of Statistical Computation and Simulation, (**2009**), September.
[45] M. Azzeh, D. Neagu and P. I. Cowling, "Analogy-based software effort estimation using Fuzzy numbers", The Journal of Systems and Software, vol. 84, (**2011**), pp. 270–284.

[46] K. Linnet, "Performance of Deming regression analysis in case of misspecified analytical error ratio in method comparison studies", Clin Chem., vol. 44, no. 5, (**1998**), pp. 1024–1031.
[47] L.-X. Jiang and W.-J. Han, "Research on Size Estimation Model for Software system Test based on testing steps and Its Application", Computer Science and Information Processing (CSIP), 2012 International Conference , pp. 1245-1248.
[48] H.W.-Jiang and L. Tian-bo, "Study On Quality Evaluation Model Of Communication System", System Science, Engineering Design and Manufacturing Informatization (ICSEM), 2012 3rd International Conference, pp. 1-4.
[49] T. Foss, E. Stensrud, B. Kitchenham and I. Myrtveit, "A simulation study of the model evaluation criterion MMRE", IEEE Trans Softw Eng, vol. 29, no. 11, (**2003**), pp. 985-995.
[50] M. A. Ahmed, I. Ahmad, J. S. Al Ghamdi, "Probabilistic size proxy for software effort prediction: A framework", Information and Software Technology, vol. 55, (**2013**), pp. 241–251.
[51] N. S. Gill and P. S. Grover, "Software size prediction before coding", ACM SIGSOFT, Software Engineering, Notes, vol. 29, no. 5, (**2005**).
[52] W. Zhang, Y. Yang and Q. Wang, "Handling missing data in software effort prediction with naive Bayes and EM algorithm", PROMISE '11, (**2011**) September 20-21, Banff, Canada.

Authors

Wan-Jiang Han, she was born in Hei Long Jiang province, China, 1967. She received her Bachelor Degree in Computer Science from Hei Long Jiang University in 1989 and her Master Degree in Automation from Harbin Institute of Technology in 1992.

She is an assistant professor in School Of Software Engineering, Beijing University of Posts and Telecommunication, China. Her technical interests include software project management and software process improvement.

Li-Xin Jiang, he was born in HeiLongJiang province, China, 1966. He received his Bachelor Degree and Master Degree in physical geography from Beijing University in 1989 and 1992.

He is a professor in the department of Emergency Response of China Earthquake Networks Center. His technical interests include software cost estimation and Emergency Response software development.

Tian-Bo Lu, he was born in Guizhou Province, China, 1977. He received his Master Degree in computer science from Wuhan University in 2003 and his PH.D Degree in computer science from the Institute of Computing Technology of the Chinese Academy of Sciences in 2006.

He is an Associate professor in School of Software Engineering, Beijing University of Posts and Telecommunications, China. His technical interests include information and network security, trusted software and P2P computing.

Xiao-Yan Zhang, she was born in Shandong Province, China, 1973. She received her Master Degree in Computer Application in 1997 and her PH.D Degree in Communication and information system from Beijing University of Posts and Telecommunication, China, in 2011.

She is an Associate professor in School of Software Engineering, Beijing University of Posts and Telecommunications, Beijing, China. Her technical interests include software cost estimation and software process improvement.

