Software Quality Journal, 13, 91–115, 2005

Preliminary Data Analysis Methods in Software Estimation

QIN LIU and ROBERT C. MINTRAM
School of Informatics, University of Northumbria, UK

Abstract. Software is quite often expensive to develop and can become a major cost factor in corporate information systems' budgets. With the variability of software characteristics and the continual emergence of new technologies the accurate prediction of software development costs is a critical problem within the project management context.

In order to address this issue a large number of software cost prediction models have been proposed. Each model succeeds to some extent but they all encounter the same problem, i.e., the inconsistency and inadequacy of the historical data sets. Often a preliminary data analysis has not been performed and it is possible for the data to contain non-dominated or confounded variables. Moreover, some of the project attributes or their values are inappropriately out of date, for example the type of computer used for project development in the COCOMO 81 (Boehm, 1981) data set.

This paper proposes a framework composed of a set of clearly identified steps that should be performed before a data set is used within a cost estimation model. This framework is based closely on a paradigm proposed by Maxwell (2002). Briefly, the framework applies a set of statistical approaches, that includes correlation coefficient analysis, Analysis of Variance and Chi-Square test, etc., to the data set in order to remove outliers and identify dominant variables.

To ground the framework within a practical context the procedure is used to analyze the ISBSG (International Software Benchmarking Standards Group data—Release 8) data set. This is a frequently used accessible data collection containing information for 2,008 software projects. As a consequence of this analysis, 6 explanatory variables are extracted and evaluated.

Keywords: data analysis in software engineering, software cost estimating models

1. Introduction

1.1. Preliminary data analysis

In the last two decades many software cost prediction techniques, models and tools have been proposed. For instance, SLIM (Putnam, 1978), COCOMO (Boehm, 1981), classification and regression trees (Porter and Selby, 1988; Boehm, 1981), Bayesian belief networks (Chulani et al., 1999; Fenton and Neil, 2000), artificial neural networks (Karunanithi et al., 1992; Samson et al., 1997). In order to build such models, a historical data set containing a reasonable number of observations is required (Boehm, 1981; Basili, 1985; Basili and Rombach, 1988; Srinivasan and Fisher, 1995). Although some success has been achieved all the prediction models have encountered the same problem, the inconsistency and inadequacy of the historical data sets. Furthermore, preliminary data analysis has rarely been emphasized. As Kitchenham et al. (2002) state:

> Analysts often plunge directly into using statistical techniques without first considering the nature of the data themselves. It is important to look first at the

> organization of the data, to determine whether any results might be due to outliers or data points that have an unreasonable influence.

Therefore, research has been complicated by the lack of either an accessible, standard historical data set or a systematically prelimary data analysis framework to assist in replicating experiments.

1.2. Data sets

Limited classical historical data sets have been published in the field, for example, Barry Boehm's COCOMO 81 data set (Boehm, 1981) and Kemerer's data set (Kemerer, 1987). However, most of these historical data sets used in software cost predicting research were company-specific data (Basili, 1985; Basili and Rombach, 1988). Because of the difficulty accessing such commercial databases, many researchers have been impelled to derive their own data collection instruments which they then employed to collected their own data. Moreover, some of the project attributes or their values are inappropriately out of date, for example the type of computer used for project development in the COCOMO 81 (Boehm, 1981) data set.

The inconsistency and inadequacy of the historical data sets is becoming a more and more significant problem in software cost estimation.

1.3. Analysis techniques

Many approaches have been proposed in software engineering data analysis. For example, statistical methods and tools have been used since the 1970's (Conte et al., 1986). Indeed, there is a dedicated book on this subject by Burr and Owen (1996) "Statistical Methods for Software Quality—Using Metrics for Process Improvement." Putnam (Putnam and Myers, 1992) also introduced using scatter diagrams and statistical trend lines in software projects data analysis. More recently, Briand et al. (1992) proposed the Optimal Subsets Reduction (OSR) model which has provided a data reduction method as a side product. However, there has no such a systematical framework of data analysis which can provide a sound basis for further research in software cost estimation till Kitchenham (1998) proposed a method of forward pass residual analysis to analyze unbalanced data sets. Although this paper presented a software project data analysis procedure, the analysis was still demonstrated on the COCOMO 81 data set. More recent research published in the field is Maxwell's approach (Maxwell, 2002), which introduces a set of statistical techniques, but has not yet been applied in a standard data set. Furthermore, it has been argued by Briand et al. (1999) that:

> No significant difference was found in accuracy between estimates derived from company-specific data and estimates derived from multi-organizational data.

This research strengthens the contention that using a standard, accessible historical data set to build software cost predicting models should contribute to the correctness and reliability of such models.

1.4. Summary of paper

This paper presents a data analysis framework derived from Maxwell's paradigm (Maxwell, 2002) and based on a new standard historical data set. The paper begins by examining the type and format of data that is typically found within the software engineering field. The ISBSG data set is then introduced and classified within the context of these standard data types. Then a set of cost estimation techniques are presented that attempt to show the major techniques that are extant within the field of cost estimation. We then present Maxwell's recipe and a framework which is based closely on Maxwell's work, except using a different statistical method (multivariate analysis). The framework shows how the predictive veracity of this model can be improved by describing a framework for preprocessing or pre-analyzing a standard data set, the ISBSG data. Then follows a discussion of the benefits that can be obtained by the use of a preprocessing technique and a standard data set.

Finally, some recommendations regarding further evaluation of the usability and applicability of the framework are suggested.

2. Software engineering data

Not all data is equal. What we mean by this is that some data types may contain more information than others. For example, an ordinal data type might indicate the programming language used on a project. In itself, the number allocated to a particular language has no meaning, it only has meaning insofar as it is different to the numbers allocated to all the other programming languages. This concept extends to other data types and is thoroughly discussed in Pfleeger et al. (1988).

Following the discussion in Pfleeger we recognize several scales of measurement—nominal, ordinal, interval, and ratio. Each captures more information than its predecessor. The data scales relate to different data types, each of which has the following characteristics.

Nominal data Nominal data relate to qualitative variables or attributes, such as gender or blood group, and are records of category membership. Nominal data is usually in the form of numbers. However, these numbers have no intrinsic meaning of their own but are in contrast only indicative. In software engineering data, variables such as business sector, application type, programming languages, etc., are nominal-scale variables, these variables differ in kind only. They have no ratio sense. There is no meaningful order. Most of the variables relate to a software project are nominal data which has less statistical power compared with other data types.

Ordinal data The value of an ordinal-scale variable can be ranked in order. The Data Quality Rating factors is an ordinal-scale variable. It is correct to say rate 1 is more reliable than rate 4. However, equal differences between ordinal values do not necessarily have equal quantitative meaning.

Interval data The value of an interval-scale variable can be ranked in order. In addition, equal distances between scale values have equal meaning. However, the ratios of interval-scale values have no meaning. This is because an interval scale has an

arbitrary zero point. Likert-type scale is an example of an interval scale. Factors are rated on a scale of equal-appearing intervals, such as very low, low, average, high, and very high, and are assigned Ratio values of 1, 2, 3, 4, and 5, respectively.

Ratio data Variables such as software project effort, application size, and duration are measured using a ration scale. Ratio scale variables can be ranked in order, equal distances between scale values have equal meaning, and the ratio of ratio scale values make sense.

Numerical or continuous data are ratio or interval in the nature. In contrast, the categorical or discrete data are ordinal or nominal in nature.

2.1. Problems related to cost estimation

A common problem in software estimation is the lack of reliable and accessible historical data sets. Various historical data sets have been introduced in the past. The following two data sets containing very limited samples:

- COCOMO 81 data set (Boehm, 1981), containing 63 projects.
- Kemerer'87 data set (Kemerer, 1987), containing 15 projects.

have been quoted and reused by many researchers, e.g., Chulani (Chulani et al., 1999), Srinivasan (Srinivasan and Fisher, 1995), Samson (Samson et al., 1997), Briand (Briand et al., 1998), Kemerer (1987) and Kitchenham (1998). However, an immediate problem is their small sample size. Other historical data sets that have been employed but rarely used in published research are:

- Boetticher, applied *Electronic Commerce* and *Fleet Management* data set (Boetticher, 2001).
- Porter used the *NASA System* data sets (Porter and Selby, 1990).
- Finnie used *Desharnais* data sets (Finnie et al., 1997) from 17 organizations.

Moreover, many researchers developed customized data collection instruments and collected their own data. For example, Basili (1985) introduced methods for collecting and validating software project data.

In general, the major problems with historic data sets are:

- Limited sample size.
- Inconsistent or out of date attributes.
- Missing values.
- Limited accessability.

2.2. The ISBSG data set

To address the above problems, it is logical to employ a frequently used, large and widely accessible data set for software estimation models. The International Software Benchmarking Standards Group, ISBSG Release8-data set contains data for 2008 projects. The ISBSG project has developed and refined its data collection standard

Table 1. ISBSG R8 data examples.

FPs	Time	R Level	Cou Tech	E method	Dev Type	Dev Platform	Prg Lang	DBMS	Org Type	Meth acquired	SWE
1587	4	4	1	3	1	1	1	1	1	1	7490
48	4	6	2	4	1	2	5	1	3	3	648
260	2	17	2	3	2	2	2	1	4	1	4150
152	1	9	1	1	1	2	6	1	5	1	668
252	1	4	1	1	2	3	7	1	6	1	3238
97	2	7	1	1	1	2	3	1	4	1	1158
303	2	12	1	2	1	1	7	0	3	1	3570
1306	1	19	1	1	1	3	9	1	6	1	73501

over a ten year period based on the metrics that have proven to be the most useful in helping to improve software development management and processes. ISBSG offers its data and materials to educational institutions, researchers and research students free of charge, subject to receipt and approval of a completed application.

The ISBSG R8 data set records true project values in the real world, and can be used to extract information to predict new projects' cost in terms of Effort. ISBSG rates the reliability of samples as 4 levels from A to D where level A represents the most reliable data. In the ISBSG R8 data set, there are 681 A-rated observations, each containing 50 explanatory variables (project size, time, programming language, etc.) and 2 response variables (summarized work effort and standardized work effort). After removing the observations that contain missing attribute values there are 345 project records remaining. The two response variables are duplicated, one is summarized work effort and the other is normalized work effort. Since both essentially encode the same information we have chosen summarized work effort as the response variable in this paper. Table 1 presents some examples from this refined data corpus where each observation comprises 11 explanatory variables and 1 response variable.

2.2.1. The explanatory and response variables A prediction model can be developed by learning from past experience from a historical data set. The model could be mathematical functional relationships, statistical statements, rule based systems, dynamic tree structures or even indefinable relationships, for instance, neural networks. To clarify the roles of the data, constructing a software effort prediction model means finding a mapping of project attributes to project effort. In effect we are looking for some form of functional relationship as:

$$\textit{Effort} = \textit{Function}(\textit{project attributes}). \tag{1}$$

We call *project attributes* the explanatory or independent variables x_i, the project *Effort* as the response or dependent variable y.

The intention of this paper is to identify the influential explanatory variables x_k with $(k \leq i)$, which contribute to the underlying relationships between the explanatory variables and the response variable:

$$y = F(x_k). \tag{2}$$

Table 2. Project attributes description.

Abbreviation	Stands for	Type	Description
FPs	Function Points	Ratio	The adjusted function point count number (adjusted by the Value Adjustment Factor).
Time	Project Elapsed Time	Ratio	Total elapsed time for project in months.
Rlevel	Resource Level	Nominal	Data is collected about the people whose time is included in the work effort data reported. Four levels are identified in the data collection instrument.
CouTech	Count Approach	Nominal	A description of the technique used to count the function points. Helps you to compare apples with apples (e.g. IFPUG, MKII, NESMA, COSMIC-FFP etc.).
Emethod	Work Effort recording method	Nominal	Three basic methods of recording are: Method-A—Staff Hours (recorded), Method-B—Staff Hours (derived), Method-C—Productive Time Only.
DevType	Development Type	Nominal	This field describes whether the development was a new development, enhancement or re-development.
DevPlatform	Development Platform	Nominal	Defines the primary development platform (as determined by the operating system used). Each project is classified as either, a PC, Mid Range or MainFrame.
PrgLang	Language Type	Nominal	Defines the language type used for the project: e.g., 3GL, 4GL, Application Generator, etc.
DBMS	DBMS used	Nominal	Whether the project used a DBMS.
OrgType	Organization Type	Nominal	This identifies the type of organisation that submitted the project (e.g.: Banking, Manufacturing, Retail).
MethAcquired	How methodology-acquired	Nominal	Describes whether the methodology was purchased or developed in-house.
SWE	Summary Work Effort	Ratio	Provides the total effort in hours recorded against the project by the development organisation.

2.2.2. The nature of the data used in this paper Within this paper we employ 345 observations of software projects, as previously indicated. Each observation contains 12 variables, 11 of which are the explanatory variables, such as project size (Function Points), project durations (Elapsed Time), etc., and the remaining attribute is the response variable that is a record of project effort (Man-Hours). Some examples are illustrated in the Table 1. The abbreviations of the variables are described in Table 2.

The size of the project is determined by the number of function points (FPs), which is a ratio variable. The most common method of counting FPs is IFPUG (IFPUG, 1994). There are 270 or 78% projects out of 345 A rating observations that were counted by IFPUG. The second major counting method is MARK II, which was adopted by 75 or 22% projects out of 345 A rating observations. The elapsed time is defined as total duration of the project in months. This is also a ratio variable. Other explanatory variables are detailed in Table 1.

Table 3. Classification of the projects with quality rating A and size measurement method IFPUG according to the methods used to measure the work effort.

		Recording Method				
		Staff Hours (recorded)	Staff Hours (derived)	Productive time only	Combination	Total
Rlevel	1	134	1	45	2	182
	2	36	29	3	–	68
	3	10	–	–	–	10
	4	8	–	2	–	10
Total		188	30	50	2	270

Table 4. Classification of the projects with quality rating A and size measurement method MARK II according to the methods used to measure the work effort.

		Recording Method				
		Staff Hours (recorded)	Staff Hours (derived)	Productive time only	Combination	Total
Rlevel	1	23	7	10	2	42
	2	3	4	17	–	24
	3	–	–	–	1	1
	4	5	1	1	1	8
Total		31	12	28	4	75

The work effort (SWE) is a ratio response variable, and is counted by various methods. The variables Resource Level (denoting which team effort—development team, development team support, computer operations involvement, end users or clients—has been counted) and Recording Method indicate what was actually measured for the work effort. For 345 samples, the combinations of the categories for these two variables are shown in Tables 3 and 4.

3. Cost estimation statistical techniques

Statistical methods and tools have been used in software engineering data analysis since the 1970's. From the application of basic scatter diagrams, and statistical trend lines (Putnam and Myers, 1992), to Analysis of Variance (ANOVA) (Maxwell et al., 1996) and testing residuals (Kitchenham, 1998), researchers have endeavoured to investigate the application of statistical techniques in analyzing software engineering data. Several common statistical techniques are discussed in this paper as a basis of the data analysis framework.

3.1. Data visualization

To avoid invalid or incorrect data misleading the conclusions, it is necessary to validate the data before starting analysis. Most of the historical databases contain extreme samples, for example, a few projects with a very high project effort; or a very big

project size. They may also contain many low effort, and small size projects. The data visualization technique, for example, the data plot, is very helpful in detecting and removing these outliers. This technique is also used to observe whether the data is normally distributed.

3.2. Data transformation

Data transformation is another technique to validate the data. Many statistical techniques assume that the underlying data is normally distributed. However, the distributions of some variables, for example, software project effort and size are not normally distributed. They might have a wide range. For example, software project size of the data set used in this paper spreads from 31 to 4887 function points. To approximate a normal distribution, transformation techniques must be used. Some common transformations include taking the natural log or subtracting the population standard deviation from the data values. Transformation methods generally make large values smaller and bring the data closer together.

3.3. Recode data

It is sometimes convenient to combine or alter the categories that make up a variable. We can construct a new variable with the new category assignments. For example, in Kermerer (1987), he recoded the "Appl" variable as 6 indicator variables, "Lang" as 7 indicator variables, "Type" as 4 indicator variables, and "Count" as 3 indicator variables. As another example we can recode whether a DBMS has been used in the project by "0" or "1" to indicate "No" or "Yes," respectively.

3.4. Correlation analysis

The independence of explanatory variables is a common assumption when building a multi-variable model.

A correlation coefficient measures the strength and direction of the relationship between two continuous variables (data should be ratio or interval in nature). The correlation coefficient can have any value between -1 and $+1$. If the correlation coefficient is -1, this means that the two variables are perfectly negatively correlated. If the correlation coefficient is $+1$, this means that the two variables are perfectly positively correlated. If the correlation coefficient is 0, this means that the two variables are not correlated at all.

Two measures of correlation are commonly used when analyzing software project data. Spearman's rank correlation coefficient must be used when the data is ordinal, or when the data is far from normally distributed. Pearson's correlation coefficient can be used when the data is of an interval or ratio type. It has been argued by Putnam (1978) that software size is positively correlated to project effort.

Spearman's rank correlation coefficient tests order relationships rather than actual values. This correlation coefficient is also less sensitive to extreme values than the

standard Pearson correlation coefficient. This correlation analysis is very efficient for ordinal variables and quasi-interval Likert scale variables. If the absolute value of Spearman's ρ is greater than or equal to 0.75, and the $Pr > |t|$ value equals 0.05 or less, then the two variables are strongly correlated and should not be included in the final model together.

3.5. Stepwise regression analysis

Ratio or interval scale project attributes, in another words, explanatory variables, can easily be included in the analysis procedure. Their impact on the response variable can be assessed using simple linear regression. Stepwise regression analysis (Gravetter and Wallnau, 1996) can be used to determine the relative importance of such explanatory variable's relationship to the response variable.

3.6. Analysis of variance (ANOVA)

ANOVA is concerned with the testing of hypotheses about means. The ANOVA F-statistic is calculated by dividing an estimate of the variability between groups by the variability within groups:

$$F = \frac{\textit{Variance between}}{\textit{Variance within}}. \tag{3}$$

The null hypotheses H_0 states that there is no difference between the populations means. If we reject H_0, then we must infer that the experimental manipulation does have an effect.

If there are large differences among the explanatory variables' means then the numerator of F will be inflated and the null hypothesis is likely to be rejected. We can use ANOVA to test whether there is an effect on the response variable of each of the explanatory nominal variables. A significant *F-value* (the conventional significance level of 0.05 or 0.01) tells us that the population means are probably not all equal.

Based on above description, ANOVA could be used to test the independence between the numerical and categorical variables.

3.7. Chi-Square test for independence

Two events are independent whenever the probability of one happening is unaffected by the occurrence of the other. This concept can be extended to nominal variables. The Chi-Square test for independence compares the actual and expected frequencies of occurrence to determine whether or not two nominal variables are independent.

The assumption underpinning the Chi-Square test for independence is that, if two variables are independent, the proportion of observations in any category should be the same regardless of what attribute applies to the other variable. The null hypothesis is that there is no relationship between the two variables. The Chi-Square test is used

to test the independence between categorical variables in software engineering data analysis.

4. Applying a stepwise analysis to the ISBSG data

The ISBSG data set, because of its size and scope of its variables, is highly suited to the investigation of software cost estimation. Using this data overcomes the problem of the inconsistency and inadequacy of the historical data sets, and in addition, a preliminary data validation and analysis can be performed to remove non-dominated or interactive variables.

This section represents a framework for performing this validation and analysis that is based closely on a paradigm proposed by Maxwell (2002). Briefly, the framework applies a set of statistical approaches, that includes correlation coefficient analysis, analysis of variance and Chi-Square test, etc., to the data set in order to remove outliers and identify dominant variables.

4.1. Maxwell's recipe

Maxwell's approach (Maxwell, 2002) can be summarized as three stages:

1. *Validating software project data.* Data visualization and transformation.
2. *Analyzing the variance of the data.* Build a multi-variable model.
3. *Evaluating the data.* Extract the equation and test the residues.

4.1.1. Validating software project data To carry out a valid statistical analysis, we should remove outliers and missing data, and test the normality of the data. To visualize data, we can plot histograms or boxplots for the explanatory variables and the response variable individually. In the test of normality, some variables were found to violate the statistical test. In each case, transformation by taking logarithm was applied.

4.1.2. Analyzing the variance of the data Maxwell's approach expects to identify the relative significance of explanatory variables's which explain the most variation in the response variable by building a multi-variable model step by step. Adjusted R^2-values will be calculated to determine this relative significance. To acquire adjusted R^2-values, correlation analysis is applied for the ratio or interval variables, and ANOVA procedures for the nominal variables.

The idea is to run a stepwise analysis procedure, which allows us to determine the influence of explanatory (independent) variables on the response (dependent) variable. The model starts "empty" and then the variables most related to response variable are added one by one in order of importance until no other variable can be added to improve the model.

To find the explanatory variable which explains the most variation in the response variable (project effort), we perform regression procedures for the ratio or interval scale variables, and ANOVA procedures for the nominal variables. The importance

of the explanatory variables to the response variable depends on the significant value, $P > |t|$ for regression procedures (between numerical variables) and $Prob > F$ for ANOVA procedures (between categorical variables, or between numerical variables and categorical variables).

4.1.3. Evaluating the data To evaluate the independence between explanatory variables, correlation coefficient analysis, analysis of variance (ANOVA) procedure and Chi-Square test are applied according to the variables' scale.

To evaluate the goodness of fit of the model, Maxwell's approach (Maxwell, 2002) extracts the final model and tests the residues. In a well-fitted model, there should be no pattern to the errors (residuals) plotted against the fitted values. The term "fitted value" refers to the project effort predicted by the model. The term "residual" is used to express the difference between the actual effort and the predicted effort for each project. Therefore, there should be no pattern in the residuals of our final model.

4.2. A framework for data analysis in software project data

A framework based on Maxwell's recipe together with a flow chart, see Figure 1, is presented as a sequence of steps.

Step 1: Data visualization. Plot the histograms for each variable to see if the variables are normally distributed. If they are, use a boxplot to identify the outliers, then prune these outliers. If they are not normally distributed, they need to be transformed as described in the following step.

Step 2: Data transformation. To approximate a normal distribution, transformation techniques are used. Some common transformations include taking the data's natural log or subtracting the population standard deviation from the data values. These methods make large values smaller and bring the data closer together. Transformed data must be re-visualized and the outliers must be pruned.

Step 3: Correlation analysis. A common assumption about building a multi-variable model is that explanatory variables are independent of each other. To check if the ratio or interval variables are independent, a correlation analysis is used. If the absolute value of the correlation coefficient is greater than or equal to 0.75, and the $Pr > |t|$ value equals 0.05 or less, then the two variables are strongly correlated and should not be included in the final model together. Strongly related categorical variables can cause problems similar to those cased by numerical variables. Unfortunately, strong relationships involving categorical variables are much more difficult to detect. Examining the independence between categorical data is described in the following step.

Step 4: Regression analysis. Test the correlation between the explanatory numerical variables and the response variable by running a stepwise regression analysis. This procedure allows us to determine the relative importance of each explanatory numerical variable's relationship to the response variable.

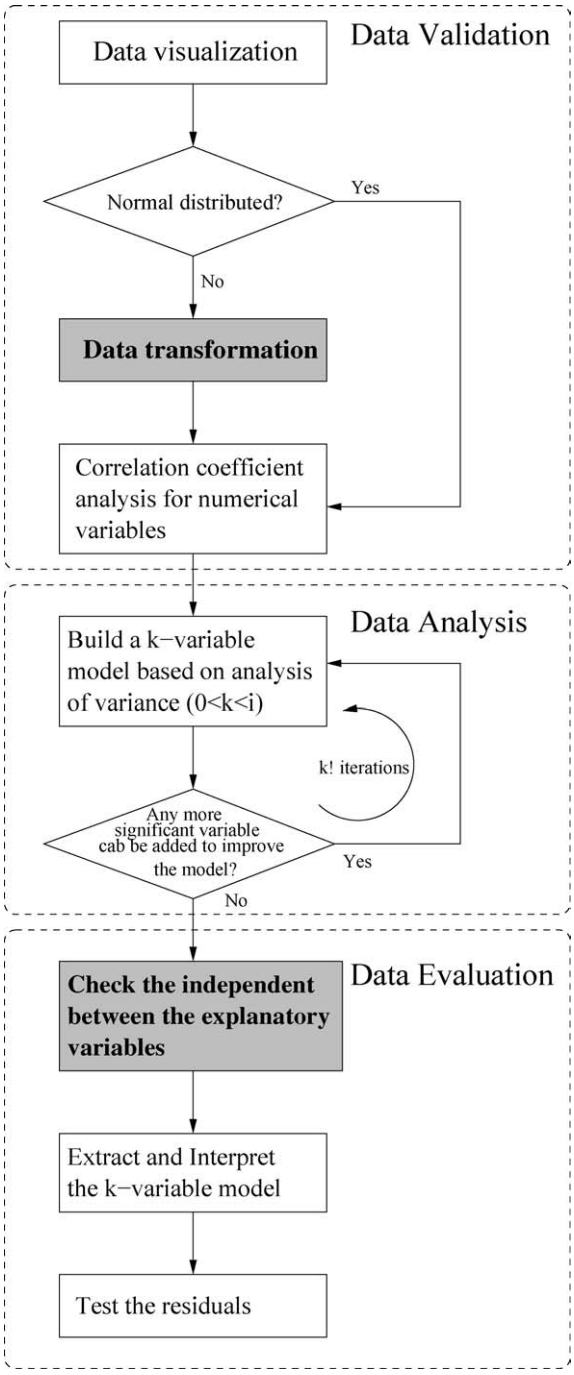

Figure 1. A data analysis framework in software project data.

Step 5: Stepwise ANOVA analysis. The following steps are to be followed.

- Find the best one-variable model. If there are i, where $i \geq 1$ explanatory variables, then we need to build i one-variable models. The best one-variable model will be determined by the adjusted R^2-values. This value is calculated for each explanatory variable with the response variable (project effort). The regression adjusted correlation coefficient square value will be calculated when the explanatory variable is type ratio or interval and the ANOVA adjusted correlation coefficient square value will be calculated when the explanatory variable is ordinal or nominal. The explanatory variable that has the greatest adjusted R^2-value explains most of the variation in the response variable and is kept in the model, we call it z_1, where $z_1 \in \{z_k\}$ and $\{z_k\} \subseteq \{x_i\}$.
- Find the best two-variable model by the same procedure. Determine which variable, x_i, in addition to z_1, explains the most variation in the response variable. This means that $i - 1$ two-variables models must be built. The most significant variable is selected as the second variable to be added into the model: z_2, where $z_2 \in \{z_k\}$ and $\{z_k\} \subseteq \{x_i\}$.
- Find the best three-variable model by the same procedure, in addition to z_1 and z_2. The process is iterative, and the number of the variables in the model is increasing until no further improvement in the model is possible. If there are two n variable $1 \leq n \leq i$ models that explain nearly the same amount of variation in the response variable then develop $n + 1$ variable models based on each of them. This process is continued iteratively until there is no significant variables that can be added to improve the model.
- Finally, we will get the best k-variable model as Equation (4)

$$Y = F(z_k) \tag{4}$$

where $\{z_k\} \subseteq \{x_i\}$ and $k \leq i$.

Step 6: Independence between explanatory variables. The independence between the numerical variables has been tested, however, at this stage it is not known whether these numerical variables are correlated to any other categorical variables. Also, it must be established whether the categorical variables are independent.

As suggested by Maxwell, to determine if there is a relationship between a categorical variable and a numerical variable, analysis of variance (ANOVA) procedures can be used. To determine if there is a relationship between two categorical variables, Chi-Square test can be used.

If significant relationships have been identified between two explanatory variables they must be studied closely and a further judgement must be made. Although Maxwell suggests making a 100% bar chart showing the percentage of each factor level of both variables, this paper proposes the application of a multivariate analysis to examine the interaction between those two variables against the response variable. This procedure will be detailed in the following section.

Step 7: Extract and interpret equations. Normally, there are transformed variables or categorical variables which have different multiplier (coefficients) for different levels.

Therefore we need to extract equations of the model. For example, if the equation read from the final model's output

$$LnEffort = (-2.406596 - 0.772367) \cdot (LnTime + 0.4475299) \cdot 77(LnSize - 0.190931) \cdot t09$$

This can be transformed into the following non-linear equation for effort.

$$Effort = \left(0.091 Time^{-0.7224}\right) \cdot \left(Size^{0.4475} \mathrm{e}^{-0.1909 \cdot t09}\right).$$

Step 8: Test residuals. In a well-fitted model, there should be no pattern to the errors (residuals) plotted against the fitted values. The term "fitted value" refers to the project effort predicted by the model, the term "residual" is used to express the difference between the actual effort and the predicted effort for each project. There should be no pattern in the residuals of our final model. Finally, plot a histogram of the residuals, if they are normal distributed, then the model is well fitted.

4.3. Applying the framework to the ISBSG data set

Step 1: Data visualization. After plotting the 11 explanatory variables, we found that none of the numerical variables was normally distributed. For example, Figure 2 shows the distribution of the raw data of Function Points (FPs). These variables needed

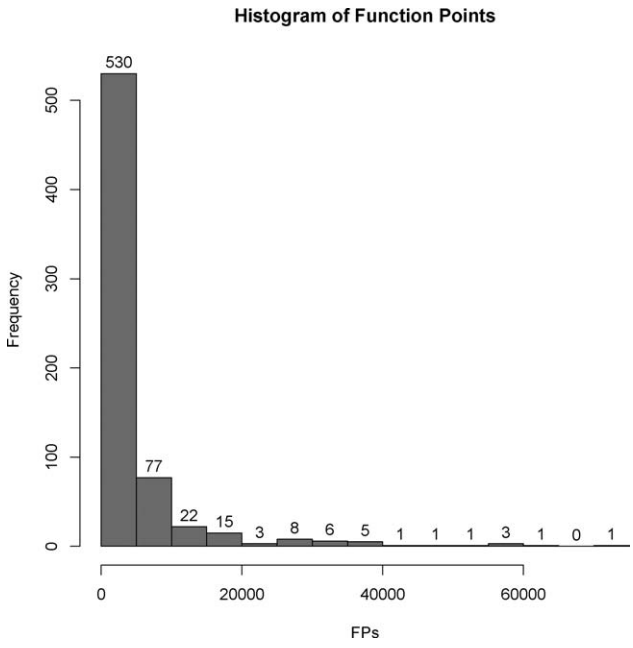

Figure 2. Histogram of original project size—Function Points (FPs).

to be transformed. A boxplot of each variable resulted in the removal of a further 18 outliers.

Step 2: Data transformation. To approximate a normal distribution, transformation techniques are used, i.e., take the data's natural log (ln). The software size (Function Points), Elapsed Time and Summarized Project Effort were transformed in this way. After transformation Step 1 is repeated. Figure 3 shows the distribution of the transformed Function Points (FPs) variable. The iterative process of Steps 1 and 2 results in a data set without outliers and which is suitable for the application of statistical techniques. The data set was further restricted to 328 projects. Each observation comprises 11 explanatory variables and 1 response variable. The explanatory variables are project attributes or their transformed values, such as the transformed project size (Ln FPs), the transformed project elapsed time (Ln Time), etc., and the response variable is recorded as transformed summarized work effort (Ln SWE). See Table 1.

Step 3: Correlation analysis. There are 3 numerical variables out of the 11 explanatory variables, LnFPs, LnTime and RLevel. We use the Spearman correlation coefficient to test their independence. According to the experiment results, there is no strong relationship between those 3 numerical variables.

Step 4: Regression analysis. We then tested the correlation between the 3 explanatory numerical variable and the respond variable by running a stepwise regression analysis. According to this analysis, one numerical variables, the RLevel was dropped. In effect, the response variable effort (LnSWE) is correlated only to the two numerical

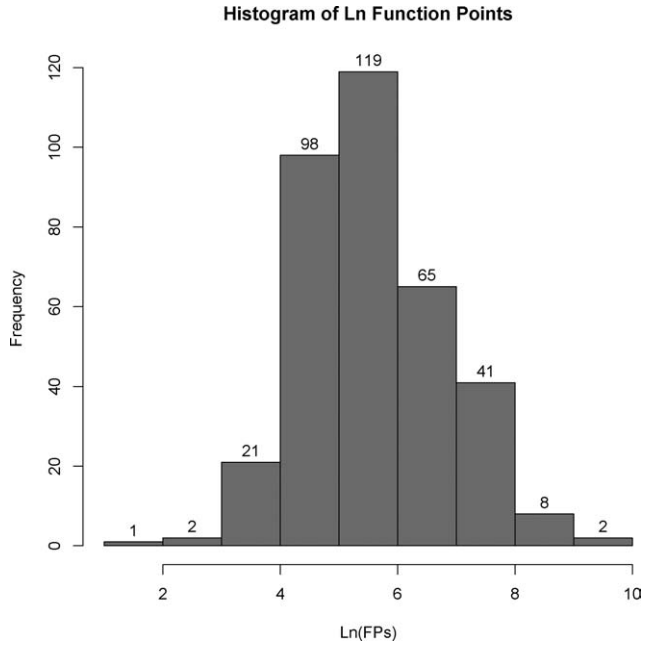

Figure 3. Histogram of transformed project size—Ln Function Points (LnFPs).

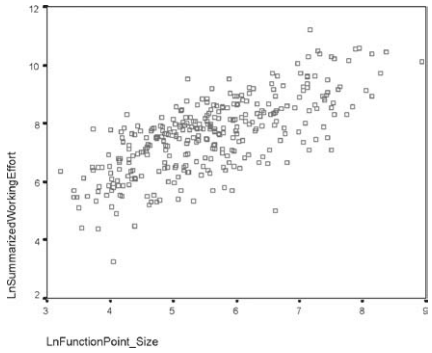

Figure 4. Scatter plot LnFPs against LnSWE.

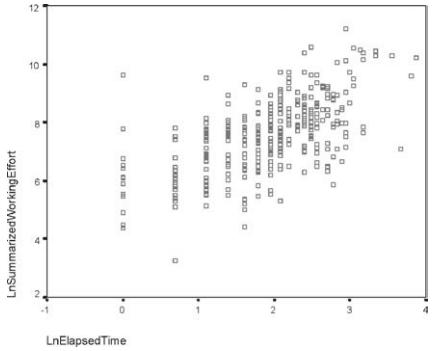

Figure 5. Scatter plot LnTime against LnSWE.

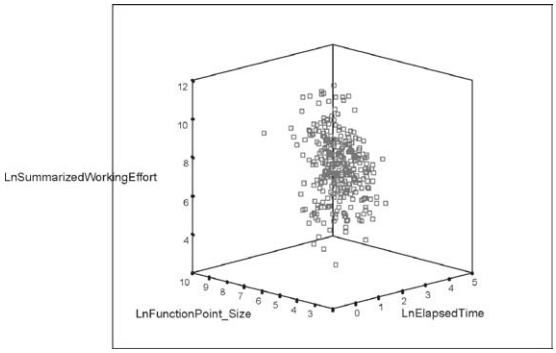

Figure 6. 3-D scatter plot LnTime and LnFPs against LnSWE.

explanatory variables: Size (LnFPs) and Duration (LnTime). These relationships can also be visualized in Figures 4–6. At this stage we have remaining 10 explanatory variables and 1 response variable.

Table 5. Stepwise analysis results summary—part 1.

Variables	Effect	Adjusted R squared	Significance of added variable	Methods	Significant code
1-variable models					
LnFPs	+	0.495	<2.2E−16	Regression	***
LnTime	+	0.3583	<2.2E−16	Regression	***
CouTech	+	0.002867	0.1649	ANOVA	
Emethod	−	0.002639	0.7091	ANOVA	
DevType	+	0.02211	0.004022	ANOVA	**
DevPlatform	+	0.07281	4.19E−07	ANOVA	***
PrgLang	+	0.001352	0.2306	ANOVA	
DBMS	+	0.008426	0.05277	ANOVA	
OrgType	−	0.001607	0.491	ANOVA	
MethAcquired	+	0.01418	0.0175	ANOVA	*
2-variable model with LnFPs					
LnTime	+	0.5542	1.22E−10	Regression	***
CouTech	+	0.4975	0.1068	ANOVA	
Emethod	+	0.5049	0.006534	ANOVA	**
DevType	+	0.4939	0.6162	ANOVA	
DevPlatform	+	0.5481	1.16E−09	ANOVA	***
PrgLang	+	0.5039	0.009613	ANOVA	**
DBMS	+	0.5022	0.01775	ANOVA	*
OrgType	+	0.5098	0.00113	ANOVA	**
MethAcquired	+	0.4941	0.5388	ANOVA	
A: 3-variable models with LnFPs and LnTime					
CouTech	+	0.5678	0.0008961	ANOVA	***
Emethod	+	0.5591	0.03293	ANOVA	*
DevType	+	0.5535	0.4742	ANOVA	
DevPlatform	+	0.6013	1.11E−09	ANOVA	***
PrgLang	+	0.559	0.0347	ANOVA	*
DBMS	+	0.5581	0.05021	ANOVA	
OrgType	+	0.5622	0.008957	ANOVA	**
MethAcquired	+	0.5546	0.2607	ANOVA	
B: 3-variable models with LnFPs and DevPlatform					
LnTime	+	0.6013	1.18E−10	ANOVA	***
CouTech	+	0.5473	0.5203	ANOVA	
Emethod	+	0.5612	0.001197	ANOVA	**
DevType	+	0.5468	0.9211	ANOVA	
PrgLang	+	0.5543	0.0198	ANOVA	*
DBMS	+	0.5493	0.1797	ANOVA	
OrgType	+	0.5556	0.01142	ANOVA	*
MethAcquired	+	0.5476	0.4233	ANOVA	

Table 6. Stepwise analysis results summary—part 2.

Variables	Effect	Adjusted R squared	Significance of added variable	Methods	Significant code
4-variable models, with LnFPs, LnTime and DevPlatform					
CouTech	+	0.6069	0.01874	ANOVA	*
Emethod	+	0.6088	0.00775	ANOVA	**
DevType	+	0.6002	0.7529	ANOVA	
PrgLang	+	0.6043	0.06312	ANOVA	
DBMS	+	0.6011	0.3553	ANOVA	
OrgType	+	0.6045	0.05932	ANOVA	
MethAcquired	+	0.6022	0.1881	ANOVA	
5-variable models, with LnFPs, LnTime, DevPlatform and Emethod					
CouTech	+	0.6193	0.001761	ANOVA	**
DevType	+	0.6079	0.622312	ANOVA	
PrgLang	+	0.6119	0.058058	ANOVA	
DBMS	+	0.6082	0.485335	ANOVA	
OrgType	+	0.6162	0.007589	ANOVA	**
MethAcquired	+	0.6091	0.264873	ANOVA	
D: 6-variable models, with LnFPs, LnTime, DevPlatform, Emethod and CountingTech					
DevType	+	0.6185	0.558137	ANOVA	
PrgLang	+	0.6214	0.098799	ANOVA	
DBMS	+	0.619	0.400667	ANOVA	
OrgType	+	0.6275	0.004757	ANOVA	**
MethAcquired	+	0.6192	0.341372	ANOVA	
E: 6-variable models, with LnFPs, LnTime, DevPlatform, Emethod and OrgType					
CouTech	+	0.6275	0.001121	ANOVA	**
DevType	+	0.615	0.914067	ANOVA	
PrgLang	+	0.6169	0.19996	ANOVA	
DBMS	+	0.6162	0.310981	ANOVA	
MethAcquired	+	0.6161	0.326111	ANOVA	

Step 5: Stepwise ANOVA analysis. Tables 5, 6 illustrate the procedure in building multi-variable model. The best one variable model is:

$$LnSWE = F(LnFPs) \tag{5}$$

which is indicate the project size is the most influential variable upon the project effort. The best two-variable model is:

$$LnSWE = F(LnFPs, LnTime). \tag{6}$$

Finally, we get the best 6-variable model:

$$LnSWE = F(LnFPs, LnTime, CountingTech, Emethod, DevPlatform, OrgType).$$

Table 7. Explanatory variables' independence evaluations.

	LnFPs	LnTime	CouTech	Emethod	DevPlatform	OrgType
LnFPs	–	$p = 0.582$ $p < 0.01$	$R^2 = -0.002$ $p = 0.729$	$R^2 = 0.029$ $p = 0.001$	$R^2 < 0.001$ $p = 0.277$	$R^2 = 0.013$ $p = 0.0236$
LnTime	$R^2 = 0.316$ $p < 0.01$	–	$R^2 = 0.028$ $p < 0.01$	$R^2 = -0.003$ $p = 0.907$	$R^2 = 0.004$ $p = 0.140$	$R^2 = -0.001$ $p = 0.521$
CouTech	–	–	–	$\chi^2 = 24.273$ $\chi - p < 0.01$	$\chi^2 = 3.382$ $\chi - p = 0.184$	$\chi^2 = 19.741$ $\chi - p < 0.01$
Emethod	–	–	–	–	$\chi^2 = 11.167$ $\chi - p = 0.083$	$\chi^2 = 63.066$ $\chi - p < 0.011$
Dev Platform	–	–	–	–	–	$\chi^2 = 6.694$ $\chi - p = 0.153$
OrgType	–	–	–	–	–	–

Table 8. Test of Between-Subject Effects 1, dependent variable: LnSWE.

Source	Type III sum of squares	df	Mean square	F	Sig.
Corrected model	17.922(a)	7	2.560	1.588	0.138
Intercept	3759.508	1	3759.508	2331.92	0.000
CouTech	8.427E−02	1	8.427E−02	0.052	0.819
Emethod	1.672	3	0.557	0.346	0.792
CouTech * Emethod	8.251	3	2.750	1.706	0.166
Error	515.903	320	1.612		
Total	19349.095	328			
Corrected Total	533.825	327			
R squared = 0.034					
Adjusted R squared = 0.012					

Step 6: Independence between explanatory variables. We have tested the independence between the numerical variables, however, we still do not know whether these numerical variables are correlated to other categorical variables. Also it must be established whether the categorical variables are independent of each other. Chi-Square tests are applied between the ratio and nominal variables, also between the nominal and nominal variables. The test results are shown in Table 7. As indicated in the table, there are 3 significant chi-squared relationships between CouTech, Emethod and OrgType. We need to look in more detail at these relationships. Instead of making graphs as suggested by Maxwell, we applied univariate analysis to examine the interaction between explanatory variables when they simultaneously influence the response variable. Although we could always present the 4-variable models. As there were no strong interactions between those variables, see Tables 8–10, we can keep them in the model. Therefore, we finally get a 6-variable model.

Step 7: Extract and interpret equations. The equation read from the 6-variable model's output is

$$\begin{aligned} \mathit{LnEffort} = {} & 2.06298 + 0.68954 \cdot \mathit{LnSize} + 0.55010 \cdot \mathit{LnTime} + \mathit{DevPlatform} \\ & + \mathit{Emethod} + \mathit{OrgType} + \mathit{CouTech}. \end{aligned}$$

Table 9. Test of Between-Subject Effects 2, dependent variable: LnSWE.

Source	Type III sum of squares	df	Mean square	F	Sig.
Corrected model	75.314(a)	23	3.275	2.171	0.002
Intercept	5979.145	1	5979.145	3964.267	0.000
CouTech	7.555	1	7.555	5.009	0.026
Orgtype	40.733	13	3.133	2.077	0.015
CouTech * Orgtype	15.437	9	1.715	1.137	0.336
Error	458.511	304	1.508		
Total	19349.095	328			
Corrected Total	533.825	327			
R squared = 0.141					
Adjusted R squared = 0.076					

Table 10. Test of Between-Subject Effects 3, dependent variable: LnSWE.

Source	Type III sum of squares	df	Mean square	F	Sig.
Corrected model	83.396(a)	33	2.527	1.650	0.017
Intercept	2968.157	1	2968.157	1937.348	0.000
Orgtype	38.333	13	2.949	1.925	0.027
Emethod	9.439	3	3.146	2.054	0.106
Orgtype * Emethod	28.156	17	1.656	1.081	0.372
Error	450.429	294	1.532		
Total	19349.095	328			
Corrected Total	533.825	327			
R squared = 0.156					
Adjusted R squared = 0.062					

Table 11. Effort factor multipliers—DevPlatform, where * means significant codes: 0 '***', 0.001 '**', 0.01 '*'.

Definition	Value	DevPlatform multiplier
PC	1	0.00
Mid range	2	**0.39056
Main frame	3	***0.47019

This can be transformed into the following equation for effort:

$$Effort = 7.869385673 \cdot Size^{0.68954} \cdot Time^{0.55010} \cdot e^{DevPlatform} \cdot e^{Emethod} \cdot e^{OrgType} \cdot e^{CouTech}.$$

The multipliers for each categorical variable are listed in Tables 11–15. For example, when the DevPlatform is MainFrame, the multiplier is 0.47019.

Step 8: Test residuals. There is no pattern to the errors (residuals) plotted against the fitted values in the final model, see Figure 7. The residuals are normally distributed as shown in Figure 8. Therefore, this model is valid, and we can conclude these 6 variables explain most of the variance of the response variable.

Table 12. Effort factor multipliers—Emethod.

Definition	Value	Emethod multiplier
Staff Hours (recorded)	1	0.00
Staff Hours (derived)	2	0.12368
Productive Time Only (recorded)	3	***0.60890
Combination	4	**1.08632

Table 13. Effort factor multipliers—CountingTech.

Definition	Value	CouTech multiplier
Convert Supported by a tool	1	0.00
Wholly Manual	2	*−0.25934
Automated	3	0.00

Table 14. Effort factor multipliers—OrgType.

Definition	Value	OrgType multiplier
Financial, Property and Business Services	1	0.00
Public administration	2	−0.25934
Manufacturing	3	0.03281
Insurance	4	−0.03528
Electricity, Gas, Water	5	−0.20037
Communication	6	0.53783
Banking	7	**0.88708
Public administration	8	−0.05632
Oil	9	0.00
Wholesale and Retail Trade	10	0.00
Aerospace or Automotive	11	−0.28934
Credit Card Processor	12	0.00
Chemicals	13	−0.25988
Medical and Health Care	14	−0.20090
Computers	15	0.00
Transport and Storage	16	−0.01241
Computer Consultants	17	0.00
Occupational Health and Safety	18	−0.25934
Revenue	19	0.00
Professional Services	20	−0.19375
Agriculture, Forestry, Fishing, Hunting	21	−0.70455
Community Services	22	0.00
Recreation, Personnel and other Services	23	0.00
Mining	24	*−1.02574
Distribution	25	0.00
Coronial Services	26	0.00
Government	27	0.00
IS-Metrics collection system	28	0.00

Table 15. Effort factor multipliers.

Definition	Value	t09 multiplier
Very low	1	0.826190
Low	2	0.682589
Average	3	0.563948
High	4	0.465928
Very high	5	0.384945

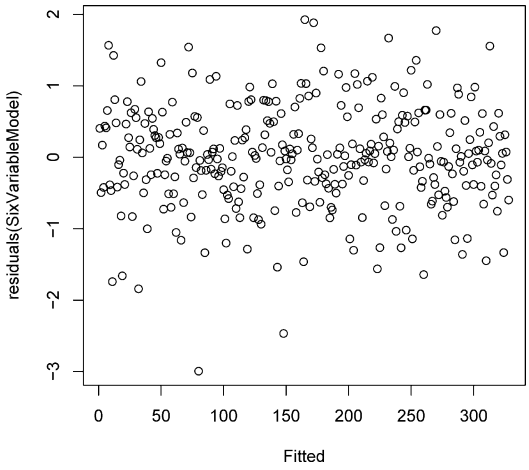

Figure 7. No pattern in the residuals of the final model.

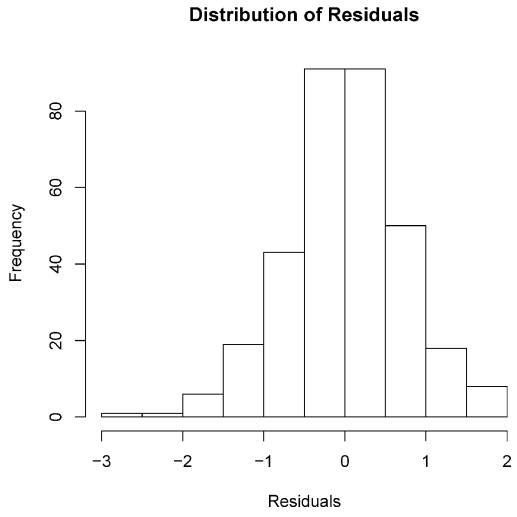

Figure 8. Residuals are normally distributed.

5. Discussion

5.1. Independence and interaction

In Maxwell's approach, we use correlation coefficients to test independence between numerical variables; ANOVA to test the independence between numerical and categorical variables, Chi-Square to test between ratio and nominal variables, and also between nominal and nominal variables. All of these tests concentrate on testing the interaction between the explanatory variables. Attention should be focused on the interaction between explanatory variables when they simultaneously effect the response variable.

5.2. Use ANOVA with caution

If there are large differences among the explanatory variables' means, the numerator of F will be inflated and the null hypothesis is likely to be rejected. There remains a problem, however, if the null hypothesis states that all the means are equal, the alternative hypothesis is simply that they are not. If the ANOVA F-test gives significance, we know there is a difference somewhere among the means, but that does not justify the conclusion that any particular comparison is significant. Further analysis is necessary to localize whatever differences there may be among the individual explanatory variables' means. That is, although we can use ANOVA to test whether there is an effect on the response variable by each explanatory nominal variables, the null hypothesis could be rejected if any pair of means is unequal. We need to locate where the significant differences lie. This requires post-hoc analysis. Unfortunately, Maxwell's paradigm does not account for this possibility.

5.3. Extract categorical data with caution

In Maxwell's recipe, there is a step called extraction of the model. However, if there are more than two multi-level variables in the final model, it will be unrealistic to extract it to a series of equations without a computerized method. According to the empirical results in this paper, not every level of a categorical variable in the final model is significant. It could be helpful if we split the entire data set partitioned based on different levels in the categorical variables.

6. Conclusions

This paper has described a general framework for the analysis of software engineering data. The framework is based closely on Maxwell's work, except using different statistical method (multivariate analysis) to test interaction between the categorical explanatory variables. Within the framework, a set of statistical experiments have been run based on a frequently used accessible data set, which contains data for 2,008 software projects. As a result of the analysis, 6 explanatory variables are extracted and evaluated. The generality of the approach and the empirical results support the usability of the framework. In addition to assessing the importance of each variable, the framework also affords the analyst a means of assessing the nature of the relationships

between the predictors and the response variable. The framework handles all scales of software project and identifies confounded variables. Furthermore, the framework provides insight into the relationships among explanatory variables in their prediction of the response measure.

Although the framework provides a systematical analysis method, there are still some aspects that need to be further considered. In particular, the analysis of the interaction between explanatory variables when they simultaneously effect the response variable.

7. Further work

To further evaluate the usability and applicability of the framework, some standard predicting techniques should be applied to construct cost estimation models. Some examples are General linear model (GLM), Classification and regression tree (CART), k-neariest neighbor (kNN) and artificial intelligent neural network (ANN). The 6-fold and 2-fold cross validation methods should be applied in constructing these models. Such experiments are currently being conducted and will be reported in a forthcoming paper entitled "Usability and Applicability of Preliminary Data Analysis in Software Cost Estimation."

Further research is necessary to extend and evaluate the accuracy of the preliminary data framework based on further historic data sets. Another outcome of the research could be the provision of benchmarking results, i.e. allowing future proposed cost estimation methods to be compared against known standards.

References

Basili, V.R. 1985. Quantitative evaluation of software methodology, *Proceedings of the 1st Pan-Pacific Computer Conference*.

Basili, V.R. and Rombach, H.D. 1988. The TAME project: Towards improvement-oriented software environments, *IEEE Transactions on Software Engineering* 14(6): 758–773.

Boehm, B.W. 1981. *Software Engineering Economics*. Englewood Cliffs, NJ, Prentice Hall.

Boetticher, G. 2001. Using machine learning to predict project effort: Empirical case studies in data-starved domains, *Proceedings of the Model Based Requirements Workshop*, pp. 17–24.

Briand, L.C., Basili, V.R., and Thomas, W. 1992. A pattern recognition approach for software engineering data analysis, *IEEE Transactions on Software Engineering* 18(11).

Briand, L.C., Emam, K.E., Surmann, D., and Wieczorek, I. 1998. An assessment and comparison of common software cost estimation modelling techniques, Technical Report ISERN-98-27, Fraunhofer Institute for Experimental Software Engineering, Germany.

Briand, L.C., Langley, T., and Wieczorek, I. 1999. A replicated assessment and comparison of common software cost modeling techniques, Technical Report, IESE-Report 073.99/E.

Burr, A. and Owen, M. 1996. *Statistical Methods for Software Quality Using Metrics for Process Improvement*. Thomson Computer Press.

Chulani, S., Boehm, B.W., and Steece, B. 1999. Bayesian analysis of empirical software engineering cost models, *IEEE Transactions on Software Engineering* 25(4): 573–583.

Conte, S.D., Dunsmore, H.E., and Shen, V.Y. 1986. *Software Engineering Metrics and Models*. Benjamin/Cummings.

Fenton, N.E. and Neil, M. 2000. Software metrics: Roadmap, "The Future of Software Engineering," *Proceedings of the 22nd International Conference on Software Engineering*, pp. 357–370. ACM Press.

Finnie, G.R., Wittig, G.E., and Desharnais, J.M. 1997. Reassessing function points, *Australian Journal of Information Systems* 4(2): 39–45.

Gravetter, F.J. and Wallnau, L.B. 1996. *Statistics for the Behavioral Science: A First Course for Students of Psychology and Education*, 4th ed. St. Paul, West.

IFPUG. 1994. Counting Practices Manual, Release 4.0, International Function Point Users Group, Westerville, OH.

Karunanithi, N., Whitley, D., and Malaiya, K.Y. 1992. Using neural networks in reliability prediction, *IEEE Software* 9(4): 53–59.

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

Kitchenham, B.A. 1998. A procedure for analyzing unbalanced datasets, *IEEE Transactions on Software Engineering* 24(4): 278–301.

Kitchenham, B.A., Pfleeger, S., Pickard, L., Jones, P., Hoaglin, D., Emam, K.E., and Rosenberg, J. 2002. Preliminary guidelines for empirical research in software engineering, *IEEE Transactions on Software Engineering* 28(8): 721–734.

Maxwell, K. 2002. *Applied Statistics for Software Managers*. UpperSaddle River, NJ, Pearson Education.

Maxwell, K., Wassenhove, L.V., and Dutta, S. 1996. A software development productivity of European space, military and industrial applications, *IEEE Transactions on Software Engineering* 22(10): 704–718.

Pfleeger, S.L., Jeffery, R., Curtis, B., and Kitchenham, B. 1997. Status report on software measurement, *IEEE Software* 14(2): 33–43.

Porter, A.A. and Selby, R.W. 1988. Learning from examples: Generation and evaluation of decision trees for software resource analysis, *IEEE Transactions on Software Engineering* 14(12): 1743–1757.

Porter, A.A. and Selby, R.W. 1990. Empirically guided software development using metric-based classification trees, *IEEE Software* 7(2): 46–54.

Putnam, L.H. 1978. A general empirical solution to the macro software sizing and estimating problem, *IEEE Transactions on Software Engineering* 4(4): 345–361.

Putnam, L.H. and Myers, W. 1992. *Measures for Excellence: Reliable Software on Time, within Budget*. Yourdon Press.

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

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

Qin Liu obtained her B.Sc. from Dalian University of Technology in China. After moving to the UK she obtained an M.Sc. in software engineering from Southampton Institute. She is now a Senior Lecturer at Northumbria University in Newcastle, UK, and is at the moment writing up her Ph.D. in software engineering metrics. Ms. Liu's principle research interest is in software project effort prediction and she has conference publications in this area.

Robert Mintram obtained a B.Sc. in mathematical physics from Sussex University and an M.Sc. in software engineering from Southampton Institute in the UK. He has also completed a Ph.D. in neural networks and machine learning from Southampton Institute. Dr. Mintram holds a senior lectureship at Northumbria University in the UK. His principal research interests are machine learning and its application to software project effort prediction. Dr. Mintram has several publications in the field of neural networks and software engineering.