Effort Estimation Using Analogy

Martin Shepperd and Chris Schofield
Department of Computing
Bournemouth University
Talbot Campus
Poole, BH12 5BB
UK

Barbara Kitchenham
National Computer Centre
Oxford Road
Manchester, M1 7ED
UK

Abstract

The staff resources or effort required for a software project are notoriously difficult to estimate in advance. To date most work has focused upon algorithmic cost models such as COCOMO and Function Points. These can suffer from the disadvantage of the need to calibrate the model to each individual measurement environment coupled with very variable accuracy levels even after calibration. An alternative approach is to use analogy for estimation. We demonstrate that this method has considerable promise in that we show it to out perform traditional algorithmic methods for six different datasets. A disadvantage of estimation by analogy is that it requires a considerable amount of computation. This paper describes an automated environment known as ANGEL that supports the collection, storage and identification of the most analogous projects in order to estimate the effort for a new project. ANGEL is based upon the minimisation of Euclidean distance in n-dimensional space. The software is flexible and can deal with differing datasets both in terms of the number of observations (projects) and in the variables collected. Our analogy approach is evaluated with six distinct datasets drawn from a range of different environments and is found to out perform other methods. It is widely accepted that effective software effort estimation demands more than one technique. We have shown that estimating by analogy is a candidate technique and that with the aid of an automated environment is an eminently practical technique.

Keywords: cost estimation, analogy, project management, metrics, software tool

1. Introduction

Accurately predicting software project costs has long exercised industrialists and researchers alike. The problems are exacerbated by the size, duration and complexity of such projects coupled with the fact that at least superficially we are concerned with building "one offs". The software engineering industry has a poor track record for estimating both effort and delivery dates [6, 10]. Nonetheless project managers and software development organisations have a great need for estimates at a very early stage in a project in order to appropriately tender for business and to properly manage resources.

Although we frequently talk about predicting costs, strictly speaking almost all approaches focus upon effort since this is generally the largest and least predictable component of project costs. This paper only addresses development effort.

Approaches to effort prediction fall into three general categories:

- expert judgement
- algorithmic models
- analogy

First, and very widely practised is expert judgement. In one sense it is not a method since the means of deriving an estimate are not explicit and therefore not repeatable. Sometimes more than one expert's opinion is combined using a Delphi type approach [4].

Next, and most popular — at least in the literature — are algorithmic models. Examples of these models include COCOMO [4], schedule compression models such as SLIM [12] and function points [2, 13]. Although the precise formulation of each model varies, at their most abstract they may all be considered as derivatives of:

$$\text{effort} = \alpha . \text{ size}^{\beta}$$

where α represents a productivity coefficient and β an economies (or diseconomies) of scale coefficient. Typically size is measured in terms of estimated lines of code (LOC) or function points. In addition most models introduce many cost drivers to try to characterise the local environment. It is, however, worth remarking that such additional complexity has not always been conspicuously successful. Kitchenham discusses the problems of lack of independence between factors [8]. Likewise, there is no evidence that the technical complexity factors for function points improve prediction accuracy (see for example Albrecht and Gaffney [2] or Abran [1]).

0270-5257/96 $5.00 © 1996 IEEE
Proceedings of ICSE-18

An important observation regarding algorithmic models is the need to adjust or calibrate the model to local circumstances. The suggestion that these models may be taken "off the shelf" and yield accurate results is repudiated by the vast majority of workers in the field (e.g. [7, 9, 11]. Even with calibration accuracy can be quite mixed.

Last, is estimation by analogy. In some respects this method is a systematic form of expert judgement since experts often search for analogous situations so as to inform their opinion. The technique involves characterising the project for which an estimate is required. This characterisation then forms the basis for finding similar or analogous projects which have been completed for which effort is known. These effort values are then used, possibly with adjustment, to generate the predicted value. Difficulties with this method include finding the analogies and assessing the degree of similarity.

The remainder of this paper presents our technique for estimating by analogy and then goes onto describe an automated tool that supports data collection and analogy by estimation. The approach is evaluated using six different datasets and compared with results generated by algorithmic models. The paper concludes with an analysis of the strengths and weaknesses of estimation by analogy and suggestions for further work.

2. Estimation by Analogy

As has already been stated, the basis of our technique for estimation by analogy is to describe (in terms of a number of variables) the project for which the estimate is to be made and then to use this description to find other similar projects that have already been completed. The known effort values for these completed projects can then be utilised to construct an estimate for the new project.

Although the concept of estimating by analogy is relatively straightforward, there are a number of problems that must be addressed. First, we have to determine how best to describe projects. Possibilities include the type of application domain, the number of inputs, the number of distinct entities referenced, the number of screens and so forth. The choice of variables must be restricted to information that is available at the point that the prediction is required. For this reason LOC is generally unsatisfactory as it must be estimated. Note that the choice of variables is flexible, although one will wish to choose variables to characterise the project as accurately as possible. It is also important to choose at least one variable that acts as a size driver, for instance number of inputs or screens or classes.

The second problem, is even once we have characterised projects, how do we determine similarity and how much confidence can we place in the analogies? Related, is the problem of knowing how many analogies to search for; too few might lead to maverick projects being used; too many might lead to the dilution of the effect of the closest analogies.

The third, and final, problem is how do we use the known effort values from the analogous projects to derive an estimate for the new project? Possibilities include means and weighted means (giving more influence to the closer analogies).

Our approach is flexible in terms of the variables available to characterise software projects. In general, more variables are better than fewer, however, in practice one is constrained to use the data that is available. Analogies are found by measuring Euclidean distance in n-dimensional space where each dimension corresponds to a variable. Values are standardised so that each dimension contributes equal weight to the process of finding analogies. The only limitation we impose is that we cannot handle categorical data other than binary valued variables (e.g. small-large or realtime-information system). This may be less of a restriction than it first appears since such information may better used to partition large datasets into smaller more homogenous ones. The variables selected for the datasets we studied were imposed by the fact that the data had already been collected. However, all datasets had at least one variable that was in some sense size related. Clearly, this is a requirement for all effort estimation approaches.

We also offer some flexibility in the number of analogies that we search for, ranging between one and three. However, our experience from an industrial dataset tends to suggests that two[1] is generally the most effective [3]. It is likely, though, that different datasets will exhibit different characteristics and as the subsequent empirical analysis shows, we use a range of techniques to obtain optimum results. Having found the analogous projects, however, we in all cases use an unweighted mean of the known effort values in order to determine the predicted value.

3. The ANGEL Software Tool

From experience finding analogies using the approach described in the previous section can be both time consuming and error prone, particularly if there are many projects or many variables. For this reason we decided to automate the process and provide an environment where data can be stored, analogies found and estimates generated. A prototype has been developed using Visual Basic to run under Windows on a PC, and is christened ANaloGy softwarE tooL (ANGEL). It is a prototype in that it is presently limited in the number of variables it can handle for determining the optimum combination of variables for finding analogies due to the fact that it

[1] A possible explanation as to why two analogies is effective is the tendency of the analogies to straddle (under and over estimate) the actual value of the project for which we wish to predict effort. This coupled with using a mean of the two effort values leads to a more accurate prediction.

employs a brute force algorithm which is comparatively inefficient[2]. In all other respects ANGEL is fully operational.

In Figure 1 we see part of a template for recording project data. Templates are an important part of our approach because they can be configured to suit the individual data collection environment of an organisation; we do not prescribe any particular set of variables in order that the approach can take optimum advantage of the data available at each data collection site. All variables and variable names are user determined other than project number, status and actual effort which are mandatory. Project number merely provides a mechanism for uniquely identifying each project. Status indicates whether a project has been completed or not and, therefore, whether it can be used as a source of analogy. Actual effort is required for all completed projects in order to provide a basis for prediction. In Figure 1, the projects displayed are all completed, and we see the different values stored for each characteristic coupled with actual effort utilised. Note that minus one signifies that the value for a variable is unavailable.

The next step (Figure 2) is to select the variables that will be used to find analogies. The reason for this is that not all variables that we collect will be helpful in finding good analogies; some variables may create noise. The chosen variables can be all, or just a subset, of the variables stored in the project template. Because the problem of determining the optimal subset of variables by hand is very hard, ANGEL can also automatically determine the best combination of variables to be used for finding analogies for a particular dataset. At present this relies upon a brute force or exhaustive search of all possible permutations, hence this is rather slow for a large number of variables. In addition, the user can select between various estimation methods including the average of the two closest analogies. As a result, the particular method for finding and using analogies can be customised to best suit individual environments.

Figure 3 shows the third step in using ANGEL which is determining the confidence we can have in using analogies drawn from a given dataset. Naturally where we only have a few projects or where the projects differ very widely in nature we will not obtain such accurate results. ANGEL has a facility to find the mean magnitude of relative error (MMRE) and the Pred (25) for any dataset by means of jack knifing. This involves successively removing each project from the dataset and using the remaining projects to derive an estimate. The estimate is then compared with the actual value and the absolute percentage error computed. The mean of all absolute percentage errors is known as MMRE [5] which is defined as:

$$100/n \sum_{i=1}^{i=n} \left(\frac{(|E_{pred} - E_{act}|)}{E_{act}} \right)_i$$

where E_{act} is actual effort and E_{est} is estimated effort and n is the number of projects in the dataset. ANGEL will also compute the percentage of projects that lie within 25% of the actual value (Pred (25)) are indicators of the likely accuracy we can obtain from using the dataset for future projects.

The final step (Figure 4) involves predicting effort for a selected project, in this case Project 20, using the completed projects from the dataset. Here we see a predicted value of 586.25 person days. The confidence that we can have in the estimate is automatically provided in the form of the MMRE and Pred(25) values as described for Figure 3.

4. Empirical Results

In this section we describe the results that have been obtained through using ANGEL to predict project effort using the analogy method described in previous sections. These results are compared with two algorithmic methods, namely linear regression and stepwise multiple regression. Prediction accuracy is assessed using the MMRE statistic. It is chosen since it is widely used when assessing predictive capability and is not limited to regression based methods as is the case with the coefficient of determination R^2.

We use a total of six datasets for our empirical analysis. These are summarised in Table 1.

Note that for the Mermaid dataset two data points were discarded since they contained incomplete information. Likewise four datapoints are discarded from the Desharnais dataset and two datapoints from the Finnish dataset. In all cases there are *a priori* reasons for eliminating these datapoints because of missing values.

It can be seen that the datasets in Table 1 are quite varied both in terms of application and between new and maintenance projects. This enables us to compare the algorithmic and analogy based effort estimation techniques in a wide range of circumstances.

Table 2 shows the MMRE values for the predictions derived from analogy based estimates (using the ANGEL tool) and two algorithmic approaches. Linear regression based predictions utilise a simple univariate model chosen from the independent variable displaying the highest correlation with actual effort. Stepwise regression builds a prediction model based upon one or more independent variables where variables are successively entered into the model until no further significant contribution can be made. Note that for the Mermaid-N dataset no

[2]For example, 20 variables for 21 projects run on a Pentium 90 takes approximately 42 hours to analyse. Fortunately, 12 variables for 21 projects (a more common situation for our datasets) takes a mere 10 minutes as the complexity is $m2^n$-1 where m is the number of projects and n the number of variables.

statistically significant model could be found using this method. The linear regression based model for the same dataset is not significant at the 5% confidence level, however, we provide prediction accuracy figure for the purposes of comparison.

Although we have six basic datasets we have also partitioned the Desharnais and Mermaid sets according to the development environment for Desharnais and according to project type (enhancement or new) for Mermaid. It is generally accepted that algorithmic models perform better on smaller more homogenous datasets and this is indeed borne out by our results. We see that the overall MMRE figure of 66% for Desharnais dataset is improved to 39%, 29% and 36% by producing separate models for each of the three environments for which the data has been collected. Similarly, the Mermaid dataset demonstrates an improvement from 252% to 62% and 226% although this has the side effect of reducing size of the Mermaid-N dataset to 8 observations, consequently, it was not possible to find statistically significant regression models.

The most striking feature of Table 2 is that in all cases the analogy method outperforms or equals the best algorithmic method. This would suggest that at least for the datasets under examination analogy based effort prediction is a superior technique to the algorithmic methods. A slightly surprising feature of Table 2 is that stepwise regression is not consistently better than linear regression. The reason for this is rather arcane. Regression analysis is based upon minimising the sum of the squares of the residuals (the difference between actual and predicted) whereas MMRE is based upon the average of the sum of the residuals. Effectively this means that poor predictions have far more influence when trying to fit a regression equation than they do when assessing the overall predictive performance of the method. This can lead to small anomalies in the relative performance of linear regression and stepwise regression models.

The final point to derive from this analysis is that the analogy based approach is able to succeed where statistical inference fails. Recall that no statistically significant relationships could be found between any of the dependent variables and actual effort for the Mermaid-N dataset. The consequence is that the algorithmic approach does not yield satisfactory results (the MMRE is 226%). By contrast, the analogy approach does not look for such relationships and is therefore a more robust and widely applicable technique.

Name	n	Description	Source
Albrecht	24	IBM DP Services projects	[2]
Atkinson	21	Builds to a large telecomms product	[3]
Desharnais	77	Canadian software house - commercial projects	[14]
Finnish	38	Data collected by the TIEKE organisation from IS projects from 9 different Finnish companies.	[15]
Kemerer	15	Large business applications	[7]
Mermaid	28	New and enhancement projects	[16]

Table 1: Datasets used to compare effort prediction methods

Dataset	Analogy	Linear Regression	Stepwise Regression
Albrecht	62%	85%	90%
Atkinson	39%	57%	45%
Desharnais	64%	66%	66%
Desharnais-1	37%	39%	41%
Desharnais-2	29%	29%	29%
Desharnais-3	26%	36%	36%
Finnish	62%	133%	101%
Kemerer	62%	107%	107%
Mermaid	78%	252%	252%
Mermaid-E	53%	62%	62%
Mermaid-N	60%	226%	-

Table 2: MMRE Values for Effort Estimation

5. Discussion

From the foregoing analysis it would seem that estimation by analogy is a superior technique to estimation via algorithmic model in at least some circumstances, however, additional evidence would be desirable. If anything the results are presented in an unfavourable light for analogy because the algorithmic results are not obtained through jack knifing and the datasets were not collected with analogy in mind and therefore the variables are not particularly well suited to characterising projects. Nevertheless, the technique would seem to offer some advantages over the other methods. It can succeed even where no statistical relationships can be found. It does not require calibration or for that matter recalibration. Lastly, it is a more intuitive method so it is easier to understand the reasoning behind a particular prediction. This in turn can increase the level of confidence in a prediction (or conversely allow a user to discount the prediction if it appears to be based upon flawed reasoning).

Analogy as a means of predicting, does have some disadvantages. As with algorithmic models, it is not clear for instance about the effect of old datapoints. As an organisation develops and successively introduces new technology the older datapoints will be increasingly misleading. When should such projects be removed from supply of candidate analogies? Nor is clear why different sets of variables and methods for generating the prediction are more or less successful with different datasets. Clearly there is a need for more investigation in this area.

Another area for further work is to study the effect of individual variables in finding analogies. Presently, we can automatically determine the optimal subset of variables but not study the individual contribution of each variable. Some mechanism akin to stepwise regression would enable investigators to better understand the behaviour of their dataset and environment.

To summarise, there is a pressing need for effective project effort prediction at an early stage in the development. In this paper we have shown that analogy is a viable estimation method for prediction, particularly when given the necessary tool support. Apart from its superior accuracy for the six datasets we studied, the analogy based approach has the benefit of being self calibrating whereas other research [11] has shown calibration to be essential for the algorithmic methods.

We do not, however, wish to create the impression that analogy based prediction should replace algorithmic methods. Different data sets will have different characteristics suggesting that a range of techniques should be considered. In addition by using more than one technique it is possible to assess the degree of risk associated with the prediction with widely divergent predictions indicating a high level of risk and need to obtain more information.

Acknowledgements:

The authors are grateful to the Finish TIEKE organisation for granting leave to use the Finnish dataset.

References:

[1] Abran, A. 'Function point models: empirical conditions for reliability and ease of use', in *Proc. European Software Cost Modelling Conference.* Ivrea, Italy: 1994.

[2] Albrecht, A.J. and J.R. Gaffney, 'Software function, source lines of code, and development effort prediction: a software science validation', *IEEE Trans. on Softw. Eng.*, 9(6), pp639-648, 1983.

[3] Atkinson, K. and M.J. Shepperd. 'The use of function points to find cost analogies', in *Proc. European Software Cost Modelling Conference.* Ivrea, Italy: 1994.

[4] Boehm, B.W., *Software Engineering Economics.* Prentice-Hall: Englewood Cliffs, N.J., 1981.

[5] Conte, S., H. Dunsmore, and V.Y. Shen, *Software Engineering Metrics and Models.* Benjamin Cummings: Menlo Park, CA, 1986.

[6] IIT Research Institute, A descriptive evaluation of software sizing models. Data Analysis Center for Software, RADC/COED, Griffiths AFB, NY 13441, USA, 1987.

[7] Kemerer, C.F., 'An empirical validation of software cost estimation models', *CACM*, 30(5), pp416-429, 1987.

[8] Kitchenham, B.A., 'Empirical studies of assumptions that underlie software cost estimation models', *Information & Softw. Technol.*, 34(4), pp211-218, 1992.

[9] Kitchenham, B.A. and N.R. Taylor, 'Software cost models', *ICL Tech. J.*, 4(3), pp73-102, 1984.

[10] Kuntzmann-Combelles, A., *Metrics for Management,* in *Software Reliability and Metrics,* B. Littlewood and N.E. Fenton, Editor, Elsevier Applied Science: 1991.

[11] Low, G.C. and D.R. Jeffery, 'Calibrating estimation tools for software development', *Softw. Eng. J.*, 5(4), pp215-221, 1990.

[12] Quantitative Software Management, Reference Notes for the DOD SLIM Software Cost Estimating Course. No. , Quantitative Software Management, McClean, VA, 1983.

[13] Symons, C.R., *Software sizing and estimating. Mk II FPA.* John Wiley: Chichester, 1991.

[14] Desharnias, J.M. *Analyse statisique de la productivitie des projects de development en informatique apartir de la techniques des points de function de fonction.* Masters thesis. Universite du Quebec, Montreal, 1988.

[15] Finnish Dataset: Dataset made available to the ESPRIT MERMAID project by the Finish TIEKE organisation.

[16] MM2 Dataset: Dataset made available to the ESPRIT MERMAID project anonymously.

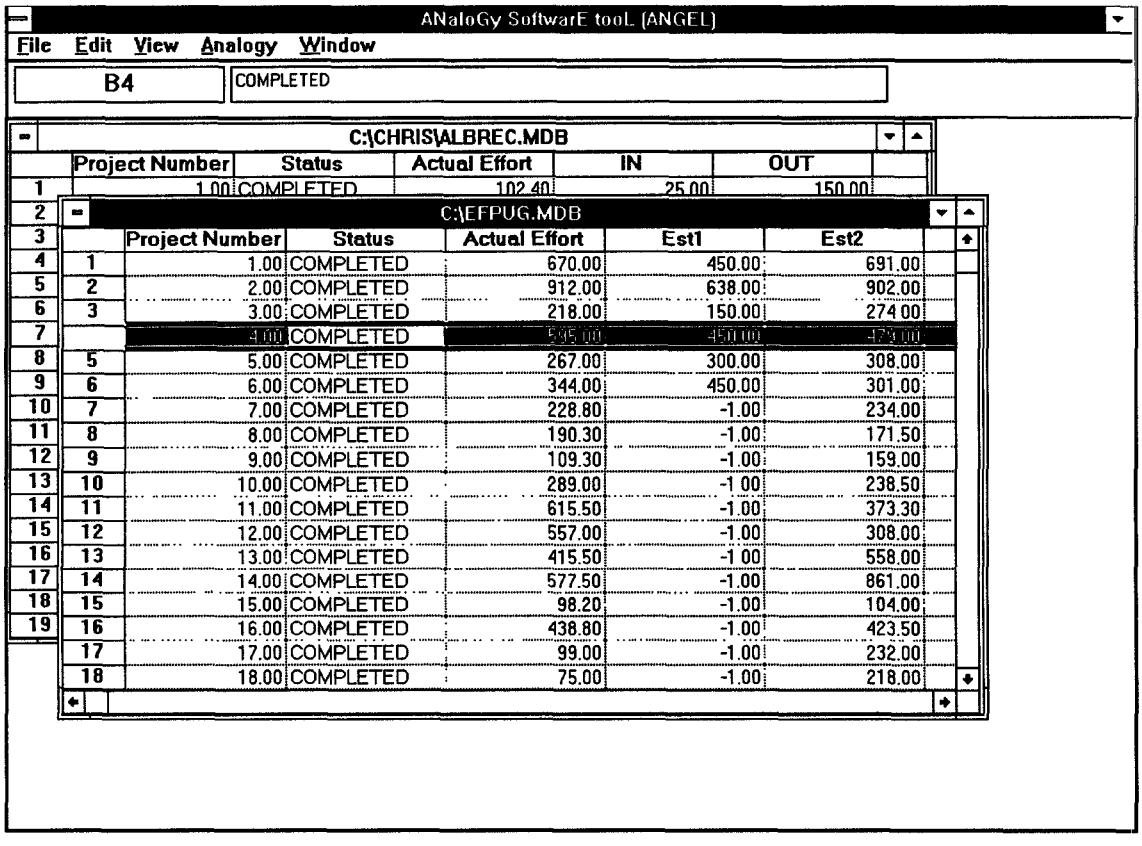

Figure 1: Data entry templates in ANGEL

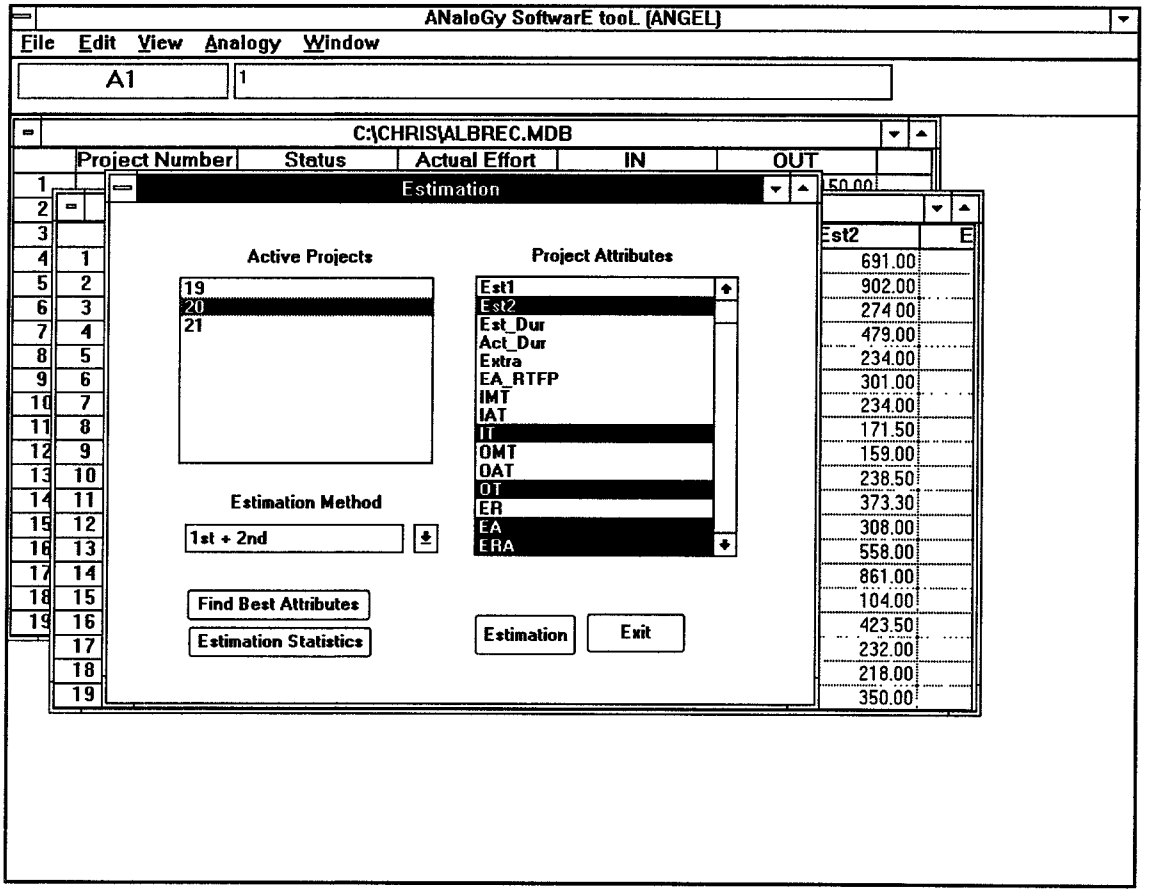

Figure 2: Choosing an analogy method in ANGEL

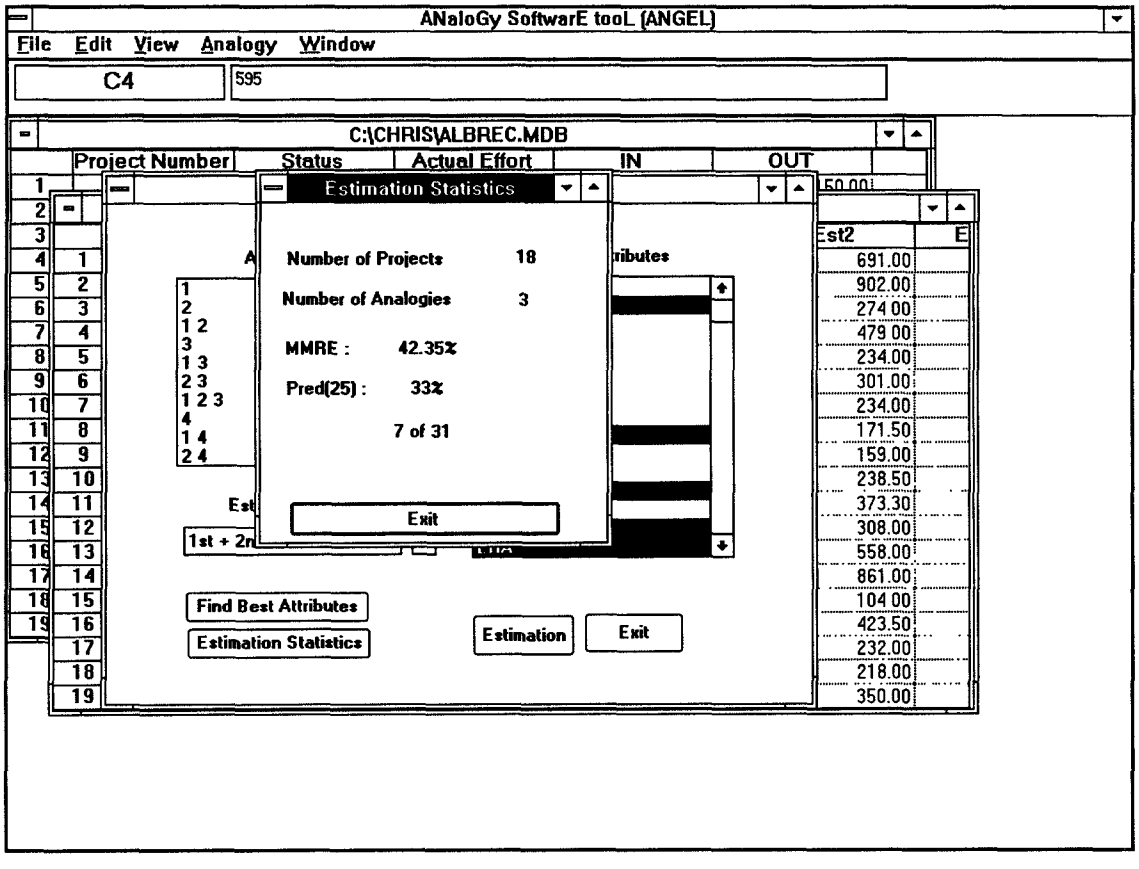

Figure 3: Assessing the reliability of a dataset in ANGEL

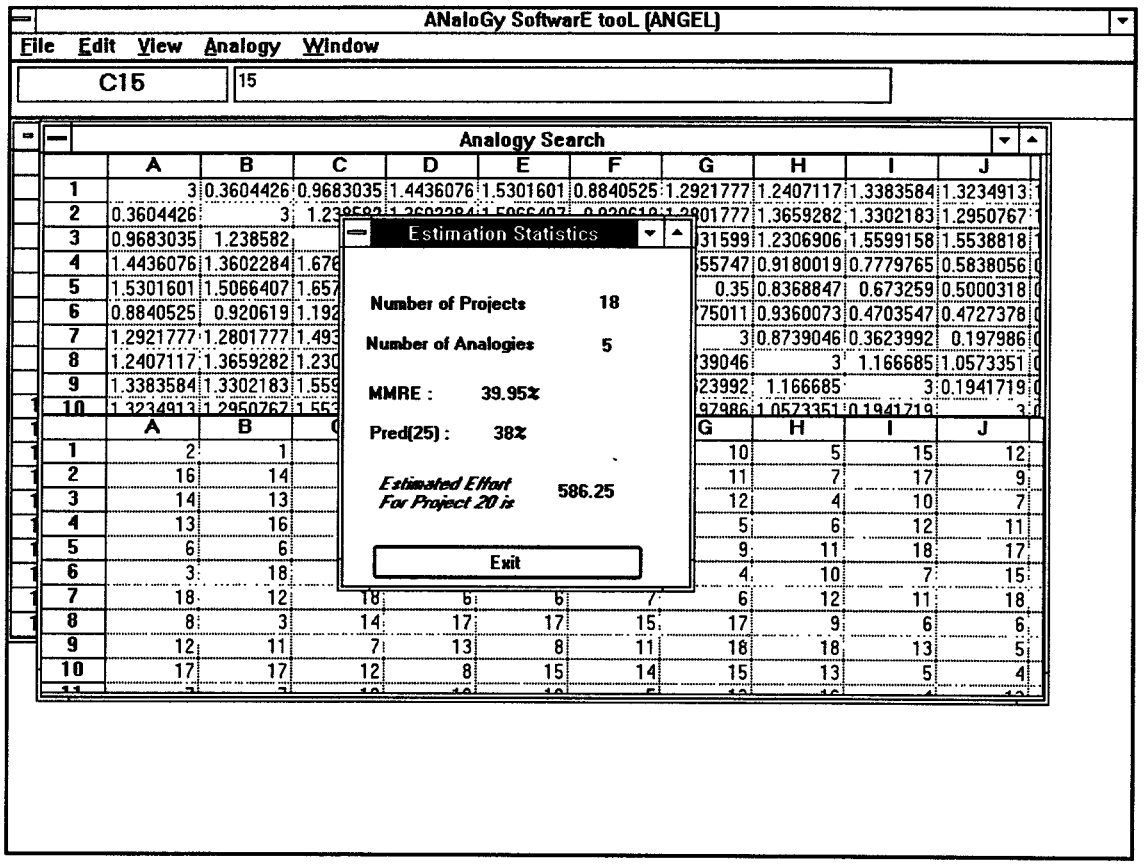

Figure 4: Predicting using ANGEL