Factors systematically associated with errors in subjective estimates of software development effort: the stability of expert judgment

Andrew R. Gray, Stephen G. MacDonell, and †Martin J. Shepperd
Department of Information Science, University of Otago
†Department of Computing, Bournemouth University
{agray}{stevemac}@infoscience.otago.ac.nz †mshepper@bournemouth.ac.uk

Abstract

Software metric-based estimation of project development effort is most often performed by expert judgment rather than by using an empirically derived model (although such may be used by the expert to assist their decision). One question that can be asked about these estimates is how stable are they with respect to characteristics of the development process and product? This stability can be assessed in relation to the degree to which the project has advanced over time, the type of module for which the estimate is being made, and the characteristics of that module. In this paper we examine a set of expert-derived estimates for the effort required to develop a collection of modules from a large health-care system. Statistical tests are used to identify relationships between the type (screen or report) and characteristics of modules and the likelihood of the associated development effort being under-estimated, approximately correct, or over-estimated. Distinct relationships are found that suggest that the estimation process being examined was not unbiased to such characteristics.

1 Introduction

The ability to accurately predict the development effort required for software systems and their constituent modules, as early as possible in the development process, is of considerable importance to any organization carrying out software development. Accuracy for software metric models may of course be defined in many different ways depending on the cost function for errors. Commonly used measurements of software metric model goodness include absolute and relative errors as well as threshold measures such as $pred()$. The optimal measurement of model goodness depends on both project and organizational characteristics which can vary widely.

In this analysis a simplification of the widely used $pred()$ measure is used, where a successful estimation is within 35% of the actual, and under- and over-estimates are defined accordingly. This allows for the use of classification models to predict types of estimate errors. Development effort predictions play an important role in both project management and higher-level management decisions. These decisions include acceptance/rejection of projects, resource allocation, and planning, monitoring, and controlling development.

Predictions of development effort can be made in terms of person-hours (or days, months, or years depending on the scope of the project) using system/module characteristics commonly referred to as software metrics. These may include assessments of module size and complexity, each of which can be measured using many different software metrics, including those available from the design phase. The lower the level of detail of the required information for a predictive model (i.e. the earlier the model can be used with an acceptable degree of accuracy), the more valuable the model will be since it is in the early stages of development that such estimates are most crucial.

When estimates of effort are made, using whatever technique (such as expert opinion, regression models, case-based reasoning, neural network models, fuzzy logic models, and so on), some subjectivity is usually involved – either in making the estimates themselves (since many estimates are in fact simply *guestimates* based on subjective opinion) or in calibrating some inputs into the model such as complexity factors in function point analysis (FPA) [1].

FPA can be subjective in two main ways. First, a count must be made of the various functions, which are external inputs, external outputs, internal logical files, external logical files, and external inquiries. These functions are classified as either simple, average, or complex. These counts are then used to construct a weighted count, with the weightings obviously dependent on complexity (the weights depend on the function type and range from 3 to 15). Second, the total count for a system is then multiplied by a subjective processing complexity score constructed from rating 14 factors on a 0 to 5 scale (from 0 being not present or no influence, through to 5 being strong influence throughout).

Given the scarcity of data for calibrating such models some subjectivity is inevitable, and probably desirable. If the estimation process was limited to only empirical models then the small data sets, coupled with a high proportion of outliers, would be unlikely to result in generalisable models. The inclusion of subjective elements in such models allows for a great reduction in the number of variables, as well as the accounting for factors that are difficult to measure. Expert opinion is often difficult to quantify but can be an effective estimating tool, in its own right or as an adjusting factor for algorithmic models [13].

This realization has recently prompted researchers to revisit their approach to the issue of estimation. Since the 1970s work has largely been concentrated on developing algorithmic estimation models, under the assumption that this would (by its very nature) result in improvements in estimation accuracy. Whilst some models have indeed proved to be useful in specific cases, their general applicability remains uncertain. In a study of the causes of estimation error [9] found that, whilst subjective guesses indeed tended to result in inaccurate estimates, accuracy was not improved with the use of algorithmic models. There are other reasons why some practitioners choose not to use algorithmic models, as illustrated by the following quote taken from the estimation methodology of a major software house:

> The current [estimation] methodology is far too complex and difficult to use. This is demonstrated by the fact that it is not used. In fact, each person undertaking estimation tends to utilize their own heuristic approach.

Such sentiments are also partly behind the demand for simplified versions of techniques such as function point analysis, as these are seen by some to be unnecessarily complex and/or detailed (e.g. see [10]). For instance, another major software house states:

> Because we can write programs and develop systems very quickly ... it is important that any estimating techniques we use are quick and easy to apply, otherwise we could end up taking as long to estimate as we take to write the programs.

Sometimes the choice of method is imposed – [5] describe the situation where an organization makes a significant change to its software process, rendering algorithmic models developed and calibrated from historical data virtually worthless. Under these circumstances expert estimation may be the only alternative.

As a result we have begun to see a resurgence in the use of informed subjective estimation as a realistic alternative. In a comparative study of case-based reasoning, function point analysis COCOMO [2], and expert estimation, [11] found that the expert out-performed all the other techniques in terms of accuracy and consistency. Moreover, the case-based reasoning method, which attempts to mimic the expert estimation process, was the next best performed. Their study also reported prior evidence of the extensive use of expert estimation in industry. The results of a recent survey of software houses has provided further verification of the widespread use of expert judgment as either the only estimation method used or as a back-up to other models [3]. [5] provide further evidence of the effectiveness of expert judgment as an estimation method in a small experiment based on the personal software process [7]. [6] also reports the results of a case study of estimation practices, ultimately suggesting that research should be directed towards *supporting* expert judgment rather than abandoning it as a viable technique.

This rather pragmatic approach may be even more feasible if more formally combined with the "estimation by analogy" method. Experts provide their estimates based on what they recall of previous similar (or analogous) modules or projects, perhaps with adjustments to cope with any differences between the previously built objects and the object to be developed. [4] found that these two approaches were far more widely employed in industry than algorithmic models, a finding also supported by [8]. As a result, the analogy approach has also seen a greater degree of exposure recently (e.g. see [12]).

2 Misestimation of effort

Despite the importance of accurate effort estimates, acceptable levels of accuracy for effort estimation are surprisingly low with a common measure for a *good* model being the attainment of estimates that are within 25% of the actual figure at least 75% of the time (this can be written as $pred(0.25) \geq 0.75$). Whilst this level of accuracy may not appear ambitious it is seldom met in practice, or even in post-hoc academic studies. Thus it can be seen that misestimation is more the norm for software metric models than an occasional and unfortunate occurrence.

It might be supposed that the misestimation of development effort required for modules within a system (and also for systems themselves within an organization) is not entirely random when these estimates include subjective human input. Some associations, such as between the probability of significant misestimation and the sizes or types of the modules/systems, may be found to exist.

Further evidence of the plausibility of such effects may be taken from the following comment from one of the software houses referred to above: “We still get it wrong. We under estimate constantly. Not so much on the small projects, but quite significantly on the larger ones.”

These associations between characteristics of projects and the likelihood of misestimation may exist due to a large number of causes including:

1. changes in technology that are not fully understood in terms of their effect on effort(such as newer tools that makes some types of module easier to develop),
2. levels of personnel experience and skills (such factors are often difficult to assess, not to mention personal biases that project managers may have),
3. a lack of understanding of the module/system characteristics (for example, some features may be seen as simple by a particular project manager when in fact they require substantial development effort),
4. other influences due to the estimator’s background, or
5. any number of other related reasons (including political and motivational goals – the “price-to-win” approach, whereby a low estimate is provided simply in order to win a contract in a competitive tendering situation, is an example of how these goals may circumvent the estimation process).

In short, the existence of such consistent biases in subjective (or partially subjective) estimates seems plausible and could even be argued as being inevitable with any estimation procedure involving such subjectivity.

The question would seem to be less that of whether or not such biases exist and more concerned with the *significance* of these biases. The question of significance is harder to answer since it depends on organizational characteristics (such as available excess capability, project cost margins, project portfolio construction, and so on), and here only the issue of identifying statistically significant associations between module characteristics and estimation performance will be pursued.

3 Uses of models of systematic biases in effort estimation

Knowing the direction and magnitude of any systematic biasing effect could be seen as significant to project management for at least three reasons. First, since much effort estimation is performed using expert opinion or subjectively calibrated models then any information that enables bias correction in these estimates would be invaluable for project management purposes. Errors that are systematically made may be preventable by educating project managers that they are making such errors or by automatically correcting estimates in such circumstances. A manager who *consistently* underestimates the effort required for database modules, and underestimates the effort for larger modules is just as valuable in project management as an equally consistent correct estimator providing that the necessary adjustments are known. In some sense such adjustments may also be technology independent, or at least more so than algorithmic estimating models themselves. This would possibly enable their use for corrections on new forms of development where less empirical data is available.

Second, any such associations would indicate that models that treat the difference between estimates and actual effort as independent and identically distributed random variables would be flawed.

The third reason for such models being useful is that knowing which types of modules or systems are more likely to be misestimated allows project managers to assess the risk of estimates. For example, for modules that are high risk for underestimating it may be necessary to allow for more slack in the process. There has been little work on the risk of software metric models and this could be seen as a useful foundation for such work to be built on.

4 Health project data

In order to examine the possibility and nature of such misestimation patterns a data set of 77 observations was formed from a large health system data set. The original data set contained 85 modules for which all module data was available, less three without effort estimates and another five which included both reports and screens (without providing separate effort measures or estimates). The variables and their associated codings shown in Table 1 were used.

As can be seen the *Modifies*, *Type*, *Entry*, and *Links* variables are all binary, and there are three levels for *Estimate* and *Size*. If *Estimate* is treated as the response variable then there are 48 levels in the independent variables.

This data can be analyzed in terms of counts for each level of each variable. Since there are six variables it is difficult to meaningfully illustrate the frequencies so that relationships can be noted for further investigation, and also to check that the cells have a reasonable count.

Tables of counts for each variable can be constructed individually. All variables show a reasonable split between classes, with only *Entry* class 0 having a proportion less than 25%, in this case 13%. All other classes for all variables contain at least 25% of the observations for that variable, which seems adequate for the purposes of this analysis.

Variable	Code	Description
Modifies	0	The sum of create, modify, and delete transactions is zero.
	1	The sum of data-altering transactions is non-zero.
Type	0	The module is made up entirely of screens.
	1	The module is made up entirely of reports.
Entry	0	Data cannot be entered from this module (for example, report parameters or search criteria as well as standard data entry and editing).
	1	Data can be entered by the user.
Links	0	Only one table was used (all modules accessed at least one table).
	1	More than one table was accessed.
Estimate	-1	The effort was at least 35% underestimated.
	0	The estimated effort was within 35% of the actual.
	1	The effort was at least 35% overestimated.
Size	1	The bottom one-third of the modules as sized by actual size.
	2	The middle one-third of the modules as sized by actual size.
	3	The top one-third of the modules as sized by actual size.

Table 1. The variables in the model

	Entry	*Links*	*Modifies*	*Size*
Entry	-	-	-	-
Links	3.765 (0.052)	-	-	-
Modifies	0.011 (0.915)	6.715 (0.10)	-	-
Size	1.495 (0.473)	49.479 (0.000)	4.412 (0.110)	-
Type	0.135 (0.713)	8.263 (0.004)	54.110 (0.000)	7.852 (0.020)

Table 2. Associations between the predictor variables

The data was then analyzed using contingency table analysis, logistic regression, and log-linear modeling. Contingency analysis is used to initially determine the associations between the *Estimate* type and the other five variables. Logistic regression is used to determine the ability of the independent variables to predict underestimates, errors in estimation, and overestimates (using the 35% threshold in each case). Log-linear modeling is then used to examine the main predictor variables in terms of associations with the three levels of *Estimate*.

5 Contingency table analysis

5.1 Introduction

Contingency table analysis is suited to analyzing the independence or otherwise of the relationships between the variables in terms of the counts in each cell. If the counts follow the marginal expectations then independence can be concluded. Otherwise, an association exists and will need to be further investigated.

5.2 Analysis

There are associations among the predictor variables as shown in Table 2 where the χ^2 statistics and their significance levels are reported. The significant associations are between *Size* and *Links*, *Type* and *Links*, *Type* and *Modifies*, and *Type* and *Size*.

When analyzing the data using χ^2, Phi, Cramer's V, and the Contingency Coefficient to test for associations between each variable (except *Estimate*) and *Estimate* the results in Table 3 were found. The crosstabs are shown in Tables 4 to 8.

From Table 3 it can be seen that the value of *Estimate* is associated with *Size*, *Type*, and *Links*. There is also a potential relationship with *Modifies*. *Entry* does not seem to be associated with the development effort estimate. However, it should be remembered that these variables are intercorrelated, with, for example, smaller modules being more likely to contain no table links. The fact that such correlations exist suggests that not all variables may be needed in a model for *Estimate*.

Variable	Statistic	Value	sig.
Size	χ^2	37.079	0.000
	Likelihood Ratio	41.004	0.000
	Phi	0.694	0.000
	Cramer's V	0.491	0.000
	Contingency coefficient	0.570	0.000
Type	χ^2	8.886	0.012
	Likelihood Ratio	10.874	0.004
	Phi	0.340	0.012
	Cramer's V	0.340	0.012
	Contingency coefficient	0.322	0.012
Entry	χ^2	2.854	0.240
	Likelihood Ratio	3.315	0.191
	Phi	0.193	0.240
	Cramer's V	0.193	0.240
	Contingency coefficient	0.189	0.240
Links	χ^2	38.047	0.000
	Likelihood Ratio	41.699	0.000
	Phi	0.703	0.000
	Cramer's V	0.703	0.000
	Contingency coefficient	0.575	0.000
Modifies	χ^2	5.019	0.081
	Likelihood Ratio	5.549	0.062
	Phi	0.255	0.081
	Cramer's V	0.255	0.081
	Contingency coefficient	0.247	0.081

Table 3. Associations with *Estimate*

	Estimate			
Size	-1	0	1	Total
1	1	6	19	26
2	12	9	4	25
3	19	5	2	26
Total	32	20	25	77

Table 4. *Size* **and** *Estimate*

	Estimate			
Type	-1	0	1	Total
1	20	14	24	58
2	12	6	1	19
Total	32	20	25	77

Table 5. *Type* **and** *Estimate*

	Estimate			
Entry	-1	0	1	Total
0	5	4	1	10
1	27	16	24	67
Total	32	20	25	77

Table 6. *Entry* **and** *Estimate*

	Estimate			
Links	-1	0	1	Total
0	0	2	17	19
1	32	18	8	58
Total	32	20	25	77

Table 7. *Links* **and** *Estimate*

	Estimate			
Modifies	-1	0	1	Total
0	12	7	3	22
1	20	13	22	55
Total	32	20	25	77

Table 8. *Modifies* **and** *Estimate*

Estimate	Model 1	Model 2	Model 3
-1	1	1	0
0	0	0	0
1	0	1	1

Table 9. The three models

5.3 Conclusions from contingency table analysis

For *Size* larger modules are more likely to be underestimated in terms of effort, while smaller modules are more likely to be overestimated. The latter aspect is predictable since small modules will generally not take less than a minimal amount of time that reflects the general overhead associated with developing a module irrespective of its size or other characteristics. However the tendency for large modules to be underestimated is more pronounced than might have been expected.

The *Type* of the module seems to affect effort estimates in the manner that screens are much more likely to be overestimated, whereas reports are much more likely to be underestimated. This indicates a definite bias on the part of the estimator which would be interesting to investigate further on subsequent projects to see if it is a general bias or one specific to this project.

Modules without any *Links* are more likely to be overestimated, while those with *Links* are more likely to be underestimated. This is consistent with the *Size* effect mentioned above since larger modules are more likely to have several tables involved. While this variable may be interesting, the correlation between *Size* and itself may mitigate against its usefulness in modeling.

6 Logistic regression

6.1 Introduction

Logistic regression is a suitable technique for the problem of classifying the modules into two classes. The output of the resulting equation expresses the probability of membership to the second group (with one minus this probability being that of the observation belonging to the first group).

The predictions of probabilities can be turned into classification by using a threshold, usually 0.5 (corresponding to 50%, i.e. the most likely class) although the actual value used depends on the cost of incorrect decisions. For example, if incorrect classifications to the first group were more expensive than to the second group the cut-off point may be lowered to move some of the more marginal decisions into the second class. In the absence of any guidelines here the default value of 0.5 is used.

When developing the logistic regression models the method used was to enter all five predictor variables (*Links*, *Size*, *Type*, *Modifies*, and *Entry*) and then pick out the one or two most significant for further investigation. Assuming that two variables were reasonably related to *Estimate*, these variables were then used individually, in combination, and then in combination with an interaction.

Finally the model with the one or two variables (but without the interaction) was tested against the full model to see if any of the other variables could be influential. If this test was not significant, best indicated by the fact that the difference in deviance would be insignificant even with one degree of freedom, then the analysis ended there and the best model was selected. If the effect of the remaining variables was significant or almost significant then each of the remaining variables were tested to see if any one could reduce the deviance significantly. The process was repeated as necessary to see which variables were useful.

Thus the process may be summarized as:

1. run the analysis using all five predictor variables with *Estimate* as the dependent variable
2. select all variables that have significance levels of less than 5%
3. add variables that are close to being significant
4. develop models using
   (a) each of the significant or nearly significant variables separately
   (b) all of these variables included
   (c) all of these variables plus their interaction(s)
5. test the significance of adding the remaining variables
6. if these variables provide a significant or close to significant result then
   (a) test adding each separately
   (b) add the significant variable(s) and repeat the process
7. repeat until no further variables can be added.

As well as considering the significance of changes in the deviance statistics, Akaike's Information Criterion (AIC) was also used to encourage parsimonious models. However, where small differences in AIC existed the deviance statistic was used to select between the simpler and more complex model.

Since the problem as it stands deals with three classes, and ordinary logistic regression is only suitable for binary classification a modification of the goal is required (nominal and ordinal logistic regression are other options here). Logistic regression models were used to separately predict three variables, defined as shown in Table 9.

Model	Deviance	df	Parameters	AIC	χ^2	df	p-value
Constant	104.539	76	1	106.539			
Size	73.384	74	3	79.384	31.155	2	0.0000
Type	99.734	75	2	103.734	4.806	1	0.0284
Size + Type	69.641	73	4	77.641	34.898	3	0.0000
Size + Type + Interaction(Size, Type)	66.622	71	6	78.622	37.917	5	0.0000
Size + Type + Entry + Links + Modifies	67.399	70	7	81.399	37.140	6	0.0000

Table 10. The logistic regression models for identifying underestimates

Term	Deviance Change	df	p-value
Size	31.155	2	0.0000
Type	4.805	1	0.0284
Size | Type	30.093	2	0.0000
Type | Size	3.743	1	0.0530
Interaction(Size, Type)|Size and Type	3.019	2	0.2210
Entry, Links, Modifies | Size and Type	2.242	3	0.5237

Table 11. The analysis of deviance for the logistic regression models for identifying underestimates

	Predicted 0	Predicted 1	Percentage Correct
Observed 0	34	11	75.56
Observed 1	6	26	81.25

Table 12. The classification table for the underestimates model

6.2 Model 1 – underestimates

The first model for predicting underestimates produced the results shown in Table 10. From this table it is possible to construct Table 11 which shows the change in deviance for each term.

From Tables 10 and 11 it can be seen that the best model is *Size* and *Type*. Adding *Size* to *Type* is certainly significant and adding the variables in the reverse order still results in a p-value of 0.0530 for *Type*. This is close enough to use, especially since AIC is also minimized for this model (77.641). Adding an interaction to this model, or the remaining three variables does not improve the model sufficiently. The performance of the model is shown in Table 12 with a cut-off point of 0.5.

$$\begin{aligned} logit(Under) = -0.4989 + 1.3093\,Type - 4.2847\,Size1 \\ - 1.4696\,Size2 \end{aligned} \tag{1}$$

The model is therefore as shown in Equation 1. This can be interpreted as meaning that for a large (*Size3*) screen-based module the probability of the estimate being an underestimate is about 69% (note that screens are *Type*=1, and reports are *Type*=2). For a report the probability increases (to 89%), while for smaller modules the probability decreases (to 3% and 34% respectively for *Size1* and *Size2* screens, and 10% and 66% for *Size1* and *Size2* reports).

Thus it can be seen that the size and type of the module do appear to share an association with the quality of the estimate in terms of project management making underestimates. More specifically, reports are more likely to be underestimated as are larger modules.

6.3 Model 2 – errors

The second logistic regression model for predicting errors in estimates produced the results shown in Table 13. Using these results Table 14 can be obtained showing the change in deviance for each term. The lack of data prevented the development of a model including the interaction between *Links* and *Size* even if *Size* was considered sufficiently significant to warrant further investigation. There are zero links to other tables only for small modules.

From Tables 13 and 14 it can be seen that the best model is *Links*. This does not quite minimize AIC (88.634 versus 88.475 for both *Size* and *Links*) but is very close. Adding other variables to *Links* does not improve the model sufficiently. The performance of the model is shown in Table 15 with a cut-off point of 0.5. Since the model only produced two values (0.6897 and 0.8947) taking any value between these two results in the performance shown in Table 16

Model	Deviance	df	Parameters	AIC	χ^2	df	p-value
Constant	88.209	76	1	90.209			
Size	86.218	74	3	92.218	1.991	2	0.3696
Links	84.634	75	2	88.634	3.574	1	0.0587
Size + Links	80.475	73	4	88.475	7.734	3	0.0518
Size + Type + Entry + Links + Modifies	80.118	70	7	94.118	8.091	6	0.2315

Table 13. The logistic regression models for identifying errors in estimates

Term	Deviance Change	df	p-value
Size	1.991	2	0.3695
Links	3.575	1	0.0284
Size | Links	4.159	2	0.1250
Links | Size	5.743	1	0.0166
Entry, Type, Modifies | Size and Links	0.357	3	0.949

Table 14. The analysis of deviance for the logistic regression models for identifying errors in estimates

	Predicted 0	Predicted 1	Percentage Correct
Observed 0	0	20	0.00
Observed 1	0	57	100.00

Table 15. The classification table for the errors in estimate model

	Predicted 0	Predicted 1	Percentage Correct
Observed 0	18	2	90.00
Observed 1	40	17	29.82

Table 16. The classification table for the errors in estimate model (cutoff 0.74)

$$logit(Error) = 2.1401 - 1.3416 Links \quad (2)$$

The model is therefore as shown in Equation 2. For a module with no links to other tables the probability of an error in estimation of greater than 35% is 89%, and with links this drops to 69%. Thus it can be seen that the incidence of links to other tables from the main table is associated with more accurate estimation. Thus more complex and larger modules seem to have been estimated more accurately, which could suggest that more care was taken of their estimates or that more care was taken with their management to keep them to schedule.

6.4 Model 3 – overestimates

The third logistic regression model for predicting overestimates in effort produced the results shown in Table 17. From here Table 18 can be obtained showing the change in deviance for each term.

From Tables 17 and 18 it can be seen that the best model is *Size*, *Type*, and *Entry*. The addition of *Links* and *Modifies* does not seem to improve estimation. This model also minimizes AIC (63.687). The performance of the model is shown in Table 19 with a cut-off point of 0.5.

$$\begin{aligned} logit(Over) = &-1.7646 + 3.9111 Size1 + 1.1522 Size2 \\ &- 2.9399\, Type + 2.6532 Entry \end{aligned} \quad (3)$$

The actual model is the one shown in Equation 3. In this case the smaller modules are more likely to be overestimated, as are modules for data entry. Reports are less likely to be overestimated than screens.

This model is consistent with that obtained for predicting underestimates where this was associated with larger modules that were screen based. Here however a new variable, *Entry*, has emerged as useful for classification. This variable was only the third in the entry procedure, and only seems to be influential after accounting for *Size* and *Type*.

Model	Deviance	df	Parameters	AIC	χ^2	df	p-value
Constant	97.073	76	1	99.073			
Size	66.375	74	3	72.375	30.698	2	0.0000
Type	86.508	75	2	90.508	10.565	1	0.0012
Size + Type	58.763	73	4	66.763	38.310	3	0.0000
Size + Type + Interaction(Size, Type)	57.868	71	6	69.868	39.205	5	0.0000
Size + Type + Links	54.112	72	5	64.112	42.961	4	0.0000
Size + Type + Modifies	58.754	72	5	68.754	38.319	4	0.0000
Size + Type + Entry	53.687	72	5	63.687	43.386	4	0.0000
Size + Type + Entry + Links + Modifies	52.352	70	7	66.352	44.721	6	0.0000

Table 17. The logistic regression models for identifying overestimates

Term	Deviance Change	df	p-value
Size	30.698	2	0.0000
Type	10.565	1	0.0012
Size | Type	27.745	2	0.0000
Type | Size	7.612	1	0.0058
Interaction(Size, Type)|Size and Type	0.895	2	0.6392
Entry | Size and Type	5.076	1	0.0243
Links | Size and Type	4.651	1	0.0310
Modifies | Size and Type	0.000	1	1.0000
Links, Modifies | Size, Entry, and Type	1.335	2	0.5130

Table 18. The analysis of deviance for the logistic regression models for identifying overestimates

	Predicted 0	Predicted 1	Percentage Correct
Observed 0	38	14	73.08
Observed 1	3	22	88.00

Table 19. The classification table for the overestimates model

6.5 Conclusions from logistic regression

From the above three analyses it appears that the two main factors that are associated with errors in estimation are the size and type of the module. The existence of links and data entry capability also influence the estimate accuracy. The analysis for errors is less significant than for under- and overestimates since the point of the analysis is to be able to make such directional predictions. However, it is interesting that only *Links* was required for modeling errors in prediction. This, along with the emergence of the *Entry* variable for overestimates, suggests the models for under- and overestimation are not entirely symmetrical.

There is definite evidence that the type of the module affects the probability, which is an interesting finding. The other conclusion that size affects estimate accuracy is more predictable since small modules will generally be correct or overestimated, while large modules will more often be underestimated.

7 Log-linear analysis

7.1 Introduction

After the contingency table analysis and logistic regression models, three variables were selected for further analysis. These are *Size*, *Type*, and *Estimate*. The purpose of this analysis using log-linear modeling is to investigate associations between *Estimate* and levels of the other variables. The advantage of log-linear modeling over logistic regression is that here any number of output categories can be used.

While *Entry* and *Links* were also found to be significant in the logistic regression section, they have been discarded here for three reasons each. First, common to both variables, there is a desire for simplicity in this analysis that is greatly aided by restricting the analysis to just these two variables. Parsimony is a useful goal to keep in mind when developing models that must be interpreted and relied upon by managers who are not qualified statisticians.

	Size						
	1		2		3		
	Type		Type		Type		
	1	2	1	2	1	2	Total
Estimate							
-1	0.00	1.00	5.00	7.00	15.00	4.00	32
0	5.00	1.00	5.00	4.00	4.00	1.00	20
1	18.00	1.00	4.00	0.00	2.00	0.00	25
Total	23.00	3.00	14.00	11.00	21.00	5.00	77.00

Table 20. Overall frequency of *Size*, *Type*, *Estimate* variables

	Size						
	1		2		3		
	Type		Type		Type		
	1	2	1	2	1	2	Percentage
Estimate							
-1	0.0%	33.3%	35.7%	63.6%	71.4%	80.0%	41.6%
0	21.7%	33.3%	35.7%	36.4%	19.0%	20.0%	26.0%
1	78.3%	33.3%	28.6%	0.0%	9.5%	0.0%	32.5%
Total	29.9%	3.9%	18.2%	14.3%	27.3%	6.5%	100.0%

Table 21. Proportions of *Size* and *Type* for *Estimate*

Second, again common to both variables, the associations using *Size* and *Type* are more useful from both research and practice perspectives in that they are more generically applicable and acceptable when compared to the more obscure *Entry* and *Links* measures.

Third, *Links* was only significant for predicting errors – not errors of any direction, and *Entry* was only added to the model for overestimates as the third variable, suggesting that the majority of the association is contained in *Size* and *Type*.

The overall frequency for the reduced set of *Size*, *Type*, and *Estimate* is shown in Tables 20 and 21. The full table showing all combinations of all variables is less useful since there are 144 levels of the six variables, with only 77 observations. In addition many of the observations fall in the same cells, leading to large numbers of empty cells.

However, even from this simple presentation it again appears that there is some relationship between *Size* and *Estimate* and between *Type* and *Estimate* even when the two variables are included (as compared to the pairwise comparisons in Section 5 on contingency table analysis). Overestimates are more common for small modules and reports.

7.2 Analysis

Table 22 shows the results of the various models developed. From Table 23 the significance of the interactions between *Type* and *Estimate* as well as between *Size* and *Estimate* can be seen. However the full model with the three way interactions shows that the interaction between *Size*, *Type*, and *Estimate* is not significant.

All parameters are significant except for interactions between *Type*=1 and *Size*=1, *Estimate*=-1 and *Size*=2, and *Estimate*=0 and *Size*=2. The other 11 parameters are all significant. Residuals are normally distributed and appear to be independent.

The expected counts are shown in Table 24. From here the odds of a correct estimate for a small screen are 0.22:1, for a small report they are 1.47:1 giving a log-odds ratio of 6.7 in favor of the reports. For medium modules the corresponding odds are 0.62:1 and 0.49:1 respectively and the ratio is 1.27 in favor of screens. For large modules the odds are 0.27:1 and 0.13:1 respectively with a ratio of 2.08 in favor of screens. For all *Type* and *Size* combinations the misestimates outnumber the correct estimates except for small reports. As the modules get larger the ratio becomes more favorable to screens, suggesting that effort for large screens is easier to estimate than for large reports.

7.3 Conclusions from log-linear analysis

From this basic analysis it can be seen that both *Type* and *Size* offer considerable potential in being able to predict the probability of misestimation. Modules are more likely to be underestimated if they are screens rather than reports, and

Model	Deviance	df	Significance
Type + Size + Estimate	60.5808	12	0.0000
Type + Size + Estimate + Type*Size	52.8846	10	0.0000
Type + Size + Estimate + Type*Size + Type*Estimate	42.0105	8	0.0000
Type + Size + Estimate + Type*Size + Size*Estimate	11.8810	6	0.0647
Type + Size + Estimate + Type*Size + Type*Estimate + Size*Estimate	3.032	4	0.5539
Full model	0	0	-

Table 22. The results of the log-linear analysis using *Size* **and** *Type* **as predictors of** *Estimate*

Terms	Change in Deviance	df	Significance
All interactions	60.5808	12	0.0000
All 2-way interactions	57.548	8	0.0000
Type*Estimate	8.849	2	0.0117
Size*Estimate	38.9785	2	0.0000
Type*Estimate*Size	3.032	4	0.5539

Table 23. The analysis of deviance of the log-linear analysis using *Size* **and** *Type* **as predictors of** *Estimate*

	Size						
	1		2		3		
	Type		Type		Type		
	1	2	1	2	1	2	Total
Estimate							
-1	0.52	0.48	4.86	7.14	14.63	4.37	32
0	4.22	1.78	5.38	3.62	4.40	0.60	20
1	18.27	0.73	3.76	0.24	1.97	0.03	25
Total	23	3	14	11	21	5	77

Table 24. Expected frequency of *Size*, *Type*, *Estimate* **variables**

are larger rather than smaller. The opposite relationships apply for overestimates.

8 Estimate accuracy as a function of time

One final analysis procedure was undertaken in order to determine whether estimate accuracy might change over the period of the project. In terms of this particular system, reasonable accuracy was achieved at the beginning and end of the project, whilst the most important errors (underestimates) occurred in the middle of the project.

This suggests that either effort for the beginning and end tasks was simply easier to predict correctly, or perhaps the estimates were revisited towards the end of the project, meaning that tasks occurring at or after that point were no longer subject to significant effort error.

9 Conclusions

After the analysis of the dataset using contingency table analysis, logistic regression and log-linear modeling it appears that for this particular dataset there are indeed systematic biases in the estimates of effort. These biases exist, at least, for the size and type of the modules and also for the existence of links to multiple tables and data entry capability. These latter two characteristics are however also related to the size and type of module.

It is of course possible to produce models that adjust their estimates for these types of modules while using the full set of available numerical data for effort estimates. Simply by including binary variables in such models the intercept would be changed to reflect differences. Alternatively, for variables such as size modification to the slope parameter may be made.

Also, the results could be used in educating project estimators as to where their estimates are going wrong. This would provide a useful feedback mechanism, possibly accelerating the learning process for project managers.

Similarly, models of risk could be constructed for those modules that are more likely to be misestimated, including some accounting for the likely magnitude (more than the three levels used here could allow for this) and direction (which has been shown here to be associated with some module characteristics).

A number of limitations on this analysis need to be considered when examining the results. First, the obvious limitation of having only the one dataset is that there is no way of knowing if the biases were due to some special characteristics of the project or if the manager makes similar misjudgements on their other projects. Second, there are several other variables that could also have been related to estimate accuracy that were not able to be considered.

Despite these objections, the analysis does at least show that for this particular project manager and system combination there were systematic biases on the effort prediction involving size and type of the constituent modules. These two factors are perhaps the most intuitive and reasonable, providing a foundation for further analysis into other associations for the purposes of developing corrective models, educating and assessing project managers, and making risk estimates.

References

[1] A. Albrecht and J. G. Jr. Software function, source lines of code, and development effort prediction: a software science validation. *IEEE Transactions on Software Engineering*, 9(6):639–648, 1983.

[2] B. Boehm. *Software Engineering Economics*. Prentice-Hall, Englewood Cliffs NJ, USA, 1981.

[3] A. Gray and S. MacDonell. Fuzzy logic for software metric models throughout the development life cycle. In *Proceedings of the 1999 Annual Meeting of the North American Fuzzy Information Processing Society - NAFIPS*, New York NY, USA, 1999. IEEE Computer Society Press. To appear.

[4] F. Heemstra and R. Kusters. Function point analysis: evaluation of a software cost estimation model. *European Journal of Information Systems*, 1(4):229–237, 1991.

[5] M. Host and C. Wohlin. A subjective effort estimation experiment. *Information and Software Technology*, 39:755–762, 1997.

[6] R. Hughes. Expert judgement as an estimating method. *Information and Software Technology*, 38:67–75, 1996.

[7] W. Humphrey. *A Discipline for Software Engineering*. Addison Wesley, 1995.

[8] A. Lederer and J. Prasad. Nine management guidelines for better cost estimating. *Communications of the ACM*, 35(2):51–59, 1992.

[9] A. Lederer and J. Prasad. A causal model for software cost estimating error. *IEEE Transactions on Software Engineering*, 24(2):137–148, 1998.

[10] R. Meli. Early function points: a new estimation method for software projects. In *Procceedings ESCOM97*, Berlin, 1997.

[11] T. Mukhopadhyay, S. Vicinanza, and M. Prietula. Examining the feasibility of a case-based reasoning model for software effort estimation. *MIS Quarterly*, pages 155–171, June 1992.

[12] M. Shepperd and C. Schofield. Estimating software project effort using analogies. *IEEE Transactions on Software Engineering*, 23(12):736–743, 1997.

[13] E. Stensrud and I. Myrtveit. Human performance estimating with analogy and regression models: an empirical validation. In *Proceedings of the Fifth International Software Metrics Symposium (Metrics'98)*, pages 205–213, Los Alamitos, California, 1998. IEEE Computer Society Press.