Fraunhofer Institut
Experimentelles
Software Engineering

Using Multiple Adaptive Regression Splines to Support Decision Making in Code Inspections

Authors:
Lionel C. Briand
Bernd Freimut
Ferdinand Vollei

IESE-Report No. 063.00/E
Version 1.0
January 1, 2001

A publication by Fraunhofer IESE

Abstract

Inspections have been shown to be an effective means of detecting defects early on in the software development life cycle. However, they are not always successful or beneficial as they are affected by a number of technical and managerial factors. To make inspections successful, one important aspect is to understand what are the factors that affect inspection effectiveness (the rate of detected defects) in a given environment, based on project data. In this paper we look at how management factors, such as the effort assigned and the inspection rate, affect inspection effectiveness. We collected data on a number of code inspections and performed a multivariate statistical analysis. Because the functional form of effectiveness models is a priori unknown, we use a novel exploratory analysis technique: Multiple Adaptive Regression Splines (MARS). We compare the MARS model with more classical regression models and show how it can help understand the complex trends and interactions in the data, without requiring the analyst to rely on strong assumptions. Results are reported and discussed in light of existing studies.

Keywords: Software Inspection, Effectiveness, Multivariate Analysis, MARS

Table of Contents

1 Introduction

Inspections have been shown to be an important defect detection technology [12]. However, when one is faced with planning inspections, a number of decisions have to be made. For example, the following questions are considered relevant as they are deemed to have an impact on inspection effectiveness, that is the capacity of inspections to uncover defects:

– What overall effort to devote to the inspection?
– What should be the inspection rate?
– How many participants to involve?
– How should the material to be inspected be broken down?

In order to answer such questions, which will be discussed in further details below, we need to develop models that relate defect detection effectiveness to variables such as effort, number of participants, or the amount of code inspected. To build such effectiveness models, data on inspections need to be collected and multivariate statistical analysis techniques are required to exploit such data and capture the complexity of the phenomena under study.

Although we rely on such multivariate models to help predict and understand the relationships between defect detection and the variables mentioned above, there is, however, a problem that we typically face when building such models. Common and mature approaches, such as multivariate regression analysis, require that we specify *beforehand* the functional form of the relationships among model variables. Because there is little knowledge and theory about inspection effectiveness factors [7][3], this is difficult to do without taking the risk to fit an inadequate model to the data.

We are therefore in a typical situation where we need to perform some exploratory analysis in a multivariate context. Not only we are interested in modeling relationships, e.g., between defect detection and effort, but we would like to find out about interactions between variables, that is the way they affect each other's impact on effectiveness, e.g., effort impact on defect detection may depend on the inspection rate, that is the pace followed while inspecting documents.

This paper will first contribute by using a novel exploratory, multivariate analysis technique (MARS [8]), which has not been used before for building software engineering models. MARS is used to help build a defect detection prediction model that is as accurate as possible. For the purpose of evaluating the gain of using MARS compared to conventional approaches, we will also build ordinary least squares regression models following classical variable section procedures [6]. We will compare the two types of models in terms of their goodness of fit, predictive power, and their capacity to help us understand the phenomena under study. We will then analyze the MARS multivariate models to gain some understanding regarding a number of common hypotheses regarding inspection defect detection effectiveness and its relationship to various factors such as effort, participants, or inspection rate. From all these results, we then provide general recommendations regarding the construction of such models in other inspection environments.

The paper is organized as follows. We first summarize in Section 2 the current state of knowledge based on our review of the literature. Then, in Section 3, we describe the motivations, the environment in which our study was performed, and the data collection performed. Section 4 introduces the main modeling technique used: Multivariate Adaptive Regression Splines (MARS). Regression analysis results are then reported in Section 5, followed by MARS results in Section 6. The latter section also performs comparisons with results in Section 5. Section 7 concludes the paper by summarizing findings and lessons learned.

2 Managing and Improving Inspections

In order to improve and control inspections, it is first necessary to identify the factors impacting inspection effectiveness, that is the number[1] of defects detected. Knowingand understanding these factors will enable us to control them when planning and conducting an inspection, so that a maximum defect detection effectiveness can be achieved.

In this section, we summarize existing empirical results so that we can compare our results and discuss them in light of reported data. In the literature, several factors have been hypothesized and/or shown to affect the effectiveness of inspections. For example, several studies showed that the effort spent on inspecting an artifact has a major impact on the inspection effectiveness [12]. Christenson et al. [3] reported the preparation effort of the inspectors to be correlated with the density of defects found. Ebenau [7] identified the examination rate[2] and the preparation rate[3] as major drivers of inspection effectiveness. In a context where defects are searched during meetings (i.e., examination), spending more effort on preparation (i.e., reading a document) yields a higher understanding of the document to be inspected and hence results in more detected defects during inspection meetings. Spending more effort on examining the document simply increases the thoroughness of the inspection and increases the chances of detecting defects.

Characteristics of the inspected product can have an impact on the effectiveness of inspections as well. Some studies [3] [7] reported the size of the inspected document to impact inspection effectiveness as a larger document usually contain a larger number of defects. Additionally, the "complexity" of a product [3] and its initial quality [9] can have an effect on inspection effectiveness as these factors relate to the defect content of the inspected product.

The characteristics of the inspection team can also show some effect on inspection effectiveness. Porter et al. [10] suggest that an inspection team composed of several inspectors can enable the detection of a wider variety of defects since each inspector is likely to rely on a different expertise. The larger and the more varied the team, the better the coverage of the document, thus resulting in an increased inspection effectiveness. Additionally, the qualification of the inspection participants can impact the effectiveness. Inspectors well

[1] Using the proportion of defects found would be equivalent as this is the number of detected defects divided by a constant: the total number of defects.

[2] Effort spent examining the document in the inspection meeting per unit of document size (e.g., LOC)

[3] Effort spent reading the document during preparation per unit of document size (e.g., LOC)

versed in the application domain can already know about potential defects in the inspected product [10] [14] [12].

Finally, the organization of the inspection process and its infrastructure can have an impact on the effectiveness as well. Porter et al. [10] identify the number of inspection sessions as another factor influencing inspection effectiveness. Additionally, the defect detection technique chosen for an inspection may have an impact on effectiveness as well. For example, more systematic techniques may help inexperienced inspectors [11].

3 Case Study Setting and Data Collection

The work described in this paper took place in a business unit of Siemens AG, Germany, which is developing products and services for mobile communication and intelligent networks. In this particular business unit, inspections are performed throughout the entire life cycle to ensure the quality of all software artifacts. Thus, inspections are performed after each of the development phases: analysis, design, and coding. Due to the substantial investment in software quality through systematic inspections, the quality assurance team's objective is to continuously improve and control the defect detection effectiveness of these inspections.

Because software quality is a major objective in this environment, data is systematically collected regarding the factors that have been shown to affect inspections' effectiveness in the published literature: the number of inspection participants, the type and size of the work product, the size of the change from the last version of the work product, the inspection effort, the number of defects found (classed into major and minor defects, where major defects are those that would lead to a fault or failure in subsequent phases), and the estimated rework effort.

Depending on the artifact to be inspected, three different kinds of inspection methods are applied. First, with the so-called "comment technique", the artifact is distributed to many inspectors who simply read the document and send their comments to the author. There is no formal, precisely defined inspection procedure. The second one is an inspection approach, similar to the one described in [14], where inspectors use checklists to identify defects during preparation and where an inspection meeting is held to collect the individual inspectors' defects. Third, there are "intensive" inspections, which enhance the second inspection approach in two ways. First, during the inspection meeting, a reader reads parts of the document, which is then discussed by the participants. In this discussion inspectors also collect the defects they detected during preparation. Thus, with intensive inspection, there is more interaction between authors and inspectors on the content of the inspected document. Second, a discussion takes place on how to prevent defects in the future.

The data collected for code inspections in this environment and used throughout the analysis is listed in Table 1. In addition, qualitative data regarding the type of inspection performed (as discussed above) and the type of document inspected was collected. The definition of these categorical variables is specific to the environment under study and only meaningful in that context.

Variable	Description
Defects	sum of major and minor defects detected in the inspection
Particip	number of participants taking part in the inspection (either in preparation or in the meeting)
Effort	total effort spent by the participants for the inspection (including preparation effort and meeting effort) in person-minutes
Sessions	number of meetings that were performed to completely inspect a document (used when a document is too large to be inspected in one meeting or when multiple meetings, each with a specific focus, are performed)
Dloc	size of the change compared to the last version
Loc	total size of the inspected document.
Iloc	size of document's part actually inspected.

Table 1: Inspection Measurement

In addition, based on the data collected, two composite measures are computed: effort per participant (Effpart), inspection rate (Totrate and Rate as Loc and Iloc per effort unit, respectively).

In the analysis that follows, we only considered inspections in which at least one defect was found. The rationale for this selection was that, based on our discussion with the quality management team, we suspected that some of these zero-defect inspections might not have been thoroughly performed. In particular the ones performed according to the rather loose "comment technique" represented 77% of all zero-defect inspections and were particularly suspect. Since we could not collect information regarding the inspection process conformance or the initial quality of inspected documents, we decided it would be more prudent to eliminate these observations from the analysis. To perform such quality checks and decide on the validity of the data for the analysis at hand is usual when collecting data in industrial settings. Our goal here is to make sure we use relatively clean, valid data to identify significant inspection effectiveness factors. The heuristic we used is rough but appeared to be effective at getting cleaner relationships. These inspections detecting zero defects should be carefully investigated, as they may be the symptom of a problem.

After filtering out zero defect inspections, our data set was composed of 237 observations (code inspections). 27% of the code inspected was in assembler and the remainder in the programming language CHILL. The proportions for "comment", "intensive", and default (standard checklists) inspections were 43%, 40%, and 17%, respectively. The inspection type data, however, was deemed unreliable by the quality management team and, as a result, did not turn out to be useful for building defect prediction models.

4 Multivariate Adaptive Regression Splines

When analyzing and modeling the relationship between fault detection and inspection effort, as well as other potential effectiveness factors mentioned earlier, one of the main issues is that relationships between these variables are expected to be complex (non-linear) and to involve interaction effects. Because we currently know little about what to expect and because such relationships are also expected to vary from one organization to another, analyzing inspection data in order to understand what affects inspections' effectiveness is usually a rather complex, exploratory process.

When using conventional regression techniques, the risk to fit the data with models that may be simplistic is rather difficult to avoid. For example, we typically resort to log-linear models to handle non-linear relationships [6]. But this comes with a number of drawbacks such as forcing the model to have a null intercept or making the analysis of interactions impossible (as the whole log-linear model is a multiplicative expression). An alternative to model such complex relationships is artificial neural networks. However, such models are difficult to interpret [5] as it is nearly impossible to assess the impact of individual independent variables on the dependent variables and their interactions. Interpretation is key in our context, as the models we build are not just used for prediction purposes but are also used to support decision-making and, from a more general perspective, gain a better understanding of software engineering processes.

MARS is a novel statistical method presented in [8] and supported by a recent tool developed by Salford Systems[4]. At a high level, MARS attempts to approximate complex relationships by a series of linear regressions on different *intervals* of the independent variable ranges (i.e., subregions of the independent variable space). It is very flexible as it can adapt any functional form and is thus suitable to exploratory data analysis. One challenge though is to find the appropriate intervals on which to run independent linear regressions, for each independent variable, and identify interactions while avoiding overfitting the data. This is the purpose of the search algorithms proposed by the MARS methodology. Though these algorithms are complex and out of the scope of this paper, MARS is based on a number of simple principles. They are introduced below in order for the reader to understand the results presented in later sections. It is also interesting to note that the results in [5] show that for datasets of sizes comparable to what we use in this study, MARS models are more likely to be accurate than artificial neural networks.

4 www.salford-systems.com

Figure 1 illustrates a simple example of how MARS would attempt to fit data, in a two dimension space (where Y and X are the dependent and independent variables, respectively), with piece-wise linear regression splines. A key concept is the notion of *knots*, which are the points that mark the end of region of data where a distinct linear regression is run, i.e., where the behavior of the modeled function changes. Figure 1 shows two knots: x_1 and x_2. They delimit three intervals where different linear relationships are identified. MARS search algorithms identify appropriate knots in an automated way, though a number of search parameters have to be set by the user. Of course, in a case with higher dimensions and interactions between independent variables, the search becomes much more complex but the fundamental principles remain the same. The reader is referred to [8] for further details. In order to model the concept of knots and piece-wise linear regression splines, MARS uses the concept of basis function. These are functions of the form:

max(0, X-c), or

max(0, c-X)

where *X* is an independent variable and *c* a constant.

Such basis functions re-express an independent variable *X* by mapping it to new variables, which are of the form described above. For max(0, X-c), *X* is set to 0 for all values of *X* up to some threshold value *c* and is equal to *X* for all values of *X* greater than *c*. By mixing the two types of basis functions presented above and providing adequate values for *c*, it is possible to approximate any functional shape.

Determining the right knots (threshold values *c*) is a key challenge addressed by MARS search algorithms. In short, basis functions are used as the new independent variables of our regression estimation models. MARS also looks for interaction terms among basis functions, thus leading to the modeling of the interactions among independent variables.

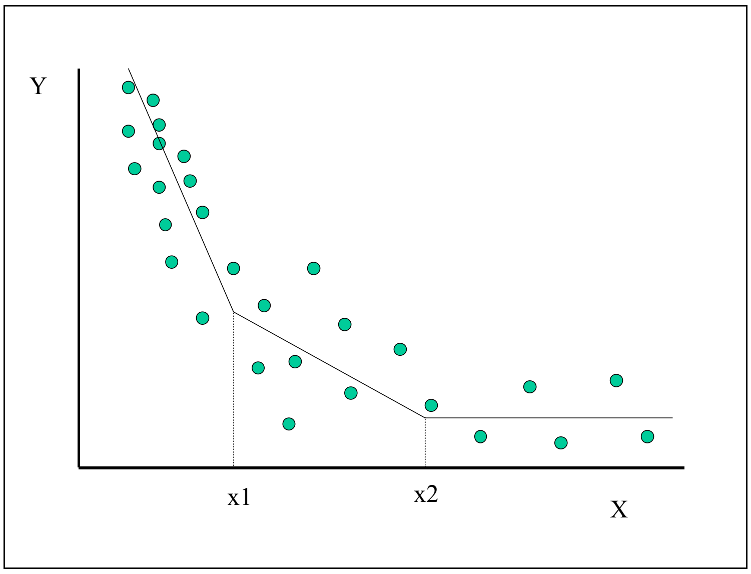

Figure 1: Example Knots in MARS

In addition, MARS provides some insight regarding the importance of variables as predictors of defect detection effectiveness, the dependent variable. MARS refits the model after removing all terms involving the variable to be assessed and calculates the reduction in goodness of fit. All variables are then ranked according to their impact on goodness of fit. An optimal MARS model, in terms of goodness of fit, is the one with the lowest generalized cross-validation (GCV) measure.

The function $\hat{f}$ is the MARS prediction model based on basis functions. *Y* is the dependent variable–the number of defects detected in our case–and there are *N* observations in the dataset. *C(M)* is the cost-complexity measure of a model containing *M* basis functions.

$$GCV(M) = \frac{1}{N}\sum_{i=1}^{N}\left[y_i - \hat{f}(x_i)\right]^2 \Big/ \left[1 - \frac{C(M)}{N}\right]^2$$

Besides the usual computation of the squared prediction error, there is a cost incurred per basis function included in the model so as to avoid overfitting, much like adjusted R^2 in least-squares regression. In other words, *C(M)* is used to penalize model complexity, prevent the overfitting of data, and promote the parsimony of models. This is usually defined as

C(M) = M

in linear least-squares regression and this is what we use in this paper. The loss in *GCV* associated with removing all the basis functions in which a variable is involved is the measure used to assess its importance in a MARS model.

Other measures of goodness of fit can be used to assess regression models from a practical standpoint. In particular, we will use four of them in this paper.

– Absolute Relative Error (ARE):

$$|\text{actual} - \text{estimated}|$$

– Magnitude of Relative Error relative to the actual value (MRE):

$$|\text{actual} - \text{estimated}| \,/\, \text{actual}$$

– Magnitude of Relative Error relative to the estimated value based on a regression model (MRE'):

$$|\text{actual} - \text{estimated}| \,/\, \text{estimated}$$

– Coefficient of determination of the regression model (R^2 between actual and predicted defects)

Looking at the above measures is relevant, especially when the models are to be used for prediction. However, as discussed in [4], they cannot really be used to compare the plausibility of non-nested models, i.e., determine which model fits the closest to reality. Therefore, such goodness-of-fit measurements should be used and interpreted with care.

A few other technical issues need to be considered when using MARS. In [5], simulations and case studies show that MARS is sensitive to outliers (i.e., observations in empty parts of the sample space, which are more difficult to detect in multidimensional settings) and strong collinearities among independent variables[5]. In the analysis below, we will attempt first to remove outlying, overinfluential observations in the sample space before building any model. However, to retain the objectivity of the analysis results, outliers will be kept during the validation stage of the models (see cross-validation below). These outliers will be identified using the Jackknife Mahalanobis distance (distance from the sample space multivariate mean or centroid) [6]. We will verify whether observations showing a very large Mahalanobis distance have an overinfluential effect on the multivariate models that we build. If this is the case, they should be

[5] Note that these problems also affect the reliability of variable selection procedures used to build least squares regression models. MARS may, however, be more sensitive to it because of the automated computations of optimal knots.

removed for model building. The main motivation here is to make sure that no one observation will distort the models being built. In the case study presented below, one observation was clearly outlying and hence removed for model building purposes. Because of space constraints, the detail of this analysis cannot be shown here. Regarding collinearity, we will use Principal Component Analysis (PCA) [6] to identify strongly collinear variables belonging to the same principal component. One variable from each principal component will then be allowed to enter the MARS models. These procedures aim at preventing, to the best extent possible, the computation of spurious results by MARS search procedures. This would otherwise prevent us from building stable, accurate models and understand the inspection processes.

5 Multivariate Regression Analysis

We first used conventional procedures for investigating the inspection data by applying multivariate, ordinary least squares regression [6]. Our goal is to use the least-squares regression results as a comparison baseline to assess the benefits of using MARS. As discussed above, we identified and eliminated one outlier. Then in order to prevent the use of independent variables being strongly collinear, we run a Principal Component Analysis (PCA) on all the variables in Table 1 and the composite measures. We ended up with 6 principal components that explained 97% of the variance in the data set (see [1] for details). The variables Effort and Effpart as well as Rate and Iloc, respectively, were part of the same principal component. In other words, each of the variable pairs capture an orthogonal dimension in the dataset. In order to avoid strongly correlated covariates, as discussed in Section 4, we decided to only allow Effort and Rate to enter the model and leave Effpart and Iloc out [1]. A multivariate regression analysis was then run, using a backward variable selection procedure [6]. The obtained (log-linear) regression model for code inspections has the following form

$$\ln(defects) = a_o + a_1 \ln(effort) + a_2 \ln(rate)$$

Because of space constraints, we refer the reader to [1] for complete details on the regression procedures we followed. Estimation statistics for the estimated coefficients a_i are shown in Table 2. For each coefficient, we provide: its estimate, the standard error of the estimate, the t-ratio of the coefficient, the statistical significance of the coefficient (i.e., the probability that the coefficient is equal to zero), and the standardized beta coefficient.

Coefficient	**Estimate**	**Std Error**	**t Ratio**	**Prob>|t|**	**Std Beta**
a_0	-3.1360	0.3112	-10.07	<.0001	0
a_1	0.7522	0.0470	15.99	<.0001	0.7759
a_2	0.2445	0.0332	7.37	<.0001	0.3573

Table 2: Regression Estimation Statistics

The fit of the model can be characterized by the adjusted R^2 (which accounts for the increased number of independent variables in a multivariate regression model [6]) for the multivariate model described above. We can also compute the adjusted R^2 in the *normal* domain, that is considering *Defects* instead of *ln(Defects)*. The adjusted R^2 in the log and normal domains are 0.52 and 0.56, respectively. The results therefore tell us that our regression model explains little more than 50% of the variance in number of defects. This means that important effectiveness factors are still not captured by our data collection. The mean relative error of the model is 0.51, 0.51, and 2.72, for MRE, MRE', and ARE, respectively. This tells us that 50% of the predictions show a relative error of 50% or above and an absolute error of approximately 3 defects or more.

In this model, the inspection effort and rate of the document show a significant impact on the number of defects detected. Estimated regression coefficients are significantly lower than one and thereby confirm that the relationships are not linear and show diseconomies of scale for effort. A straightforward interpretation is that more effort or a higher rate spent on inspections allows the inspectors to obtain a more thorough understanding of the document, thus resulting in the detection of more defects. The impact of the inspected document size can simply be explained by the fact that, if more of the document is inspected, assuming a somewhat constant defect density, more defects are to be detected. It is important to note that the log-linear model suggests that the number of defects detected does not grow proportionally to effort, rate or inspected Locs. There are several possible interpretations for this. For example, as reported in [16], when inspected document size grows there are fatigue effects resulting in lower effectiveness to find defects.

6 MARS Analysis

We present below the results obtained when performing a MARS analysis to our inspection data.

We will allow the same predictor variables as in the log-linear models to enter the MARS models, making sure no strong collinearities are present among variables that can enter the MARS model. We will first provide the models built by the MARS procedures and their validation results. Then, we will provide an interpretation of the modeling results.

6.1 Model Building and Validation

The basis functions identified by MARS search algorithms are described in Table 3. We can right away identify a number of interaction effects and, in particular, interactions between effort (as captured by basis function BF1) and a number of other variables. Such interactions can be seen in the table when basis functions are part of the definition of other basis functions, e.g., BF1 in BF4. Because of many such interactions, Table 3 suggests that the model is far from being additive and that interactions will play an important role in building an accurate model for code inspections. Models and interactions will be further discussed below.

Basis Functions
BF1 = max(0, EFFORT - 30.000);
BF4 = max(0, PARTICIP - 10.000) * BF1;
BF9 = max(0, DLOC - 3000.000) * BF1;
BF11 = max(0, SESSIONS - 4.000) * BF1;
BF14 = max(0, 0.250 - RATE) * BF1;
BF19 = max(0, EFFORT - 3600.000);
BF21 = max(0, EFFORT - 2100.000);
BF24 = max(0, 4200.000 - EFFORT);
BF25 = max(0, PARTICIP - 9.000) * BF24;
BF27 = max(0, LOC - 29.999) * BF19;
BF28 = max(0, DLOC - 4.000) * BF21;
BF30 = max(0, 0.433 - RATE) * BF21;

Table 3: MARS Basis Functions

Table 4 provides a ranking of the variables by order of importance. Variables with no impact at all are not shown. As described above, this is computed based on the reduction in goodness of fit when the variable is removed (i.e., all the basis functions involving the variable are removed). The loss in GVC is denoted as "-gvc" in Table 4. The column "Importance" shows the relative importance (percentage) of variables as compared to the best one (i.e., effort here). Confirming the regression analysis results, we can see that Effort is by far the most important variable determining defect detection effectiveness. To a lesser extent the change delta in terms of lines of code (Dloc), the number of lines of code of the artifact inspected (Loc), the rate at which inspections are taking place (Rate), and the number of sessions (Sessions) have also a significant effect on defects detected. These results are rather intuitive as the larger the amount of code modified and inspected, the larger the number of defects detected. The impact of inspection rate has been mentioned in a number of articles [1] [7] [3] and is confirmed by our analysis. We will get back to these issues later in this section.

Variable	**Importance**	**- gcv**
EFFORT	100.000	312.942
DLOC	62.006	178.804
LOC	45.308	139.755
PARTICIP	39.235	128.567
RATE	38.770	127.775
SESSIONS	12.632	98.497

Table 4: Relative Variable Importance

Table 4 shows that the two significant defect detection predictors (Effort, Rate) were already selected in the log-linear regression model. The MARS model is essentially a richer model in the sense that it models additional effects (Dloc, Loc, Participants, Sessions) and automatically identifies relevant interactions. This is what we would expect from such a data mining procedure: to uncover additional information from the data as no restrictive assumptions are made regarding the model's functional form or interactions. We will see below how the goodness of fit and predictive capability improved as a result from using additional covariates and a different functional form for the regression equation.

Variable	**Estimate**	**Std Error**	**t ratio**	**Prob>|t|**
Constant	2.218	0.807	2.748	0.006
BF 1	0.008	.71E-03	11.453	.<.0001
BF 4	0.003	.29E-03	10.435	<.0001
BF 9	.34E-05	.22E-06	15.282	<.0001
BF 11	-0.005	0.001	-4.400	<.0001
BF 14	-0.037	0.007	-5.441	<.0001
BF 25	-0.003	49E-03	-5.375	<.0001
BF 27	.91E-06	.82E-07	11.051	<.0001
BF 28	-.46E-05	.42E-06	-10.786	<.0001
BF 30	-0.033	0.006	-5.549	<.0001

Table 5: MARS Regression Statistics

Though the regression model presented in Table 5 is more complex than the log-linear regression model, the number of covariates (9) is still very reasonable as compared to the number of observations (236). We want to ensure that the model generated is stable and will be accurate over other datasets. A typical rule of thumb is to have a minimum of 10 data points for each covariate in a regression model. MARS parameters have to be set to avoid overfitting with a too large number of covariates, i.e., basis functions. As mentioned in [5], this is relatively easy with the recent MARS tool but is outside the scope of this paper.

Table 6 summarizes the comparison of goodness of fit of the two models. Several measures are presented as all provide valid insights into goodness of fit. R^2 is informative about the percentage of defect variance explained by the regression models. We can see from Table 6 that the coefficient of determination R^2 (or rather the adjusted R^2, which is adjusted for the number of covariates) of the MARS regression (0.785) significantly outperforms the log-linear regression model (0.559). Table 6 also shows the results regarding the relative error of the models.

A log-linear model gives more weight to observations with smaller actuals (due to the log transformation), i.e., they weigh more on the estimation of regression coefficients. Since smaller actuals tend to yield higher MREs, lower MREs resulting from log-linear models are in fact more of a mathematical artifact than an evidence of better goodness of fit. Furthermore, regression models optimize R^2, not relative error, and they should be compared on that basis. But it is also a well-known fact that comparing non-nested regression models with R^2 may be misleading [4]. A better way of comparing the plausibility of non-nested regression models is the J-test (See [1] for relevant details and [4] for a complete description). This test confirms very clearly that the MARS model is more plausible than the log-linear model and is therefore more appropriate for interpretation purposes.

We also evaluated the predictive power of the log-linear model and the MARS model using cross-validation [5]. We randomly divided up the dataset (including the outlier as discussed before) into 10 subsets. Each subset was in turn used as a test set and the complementary set was used to refit both models. Thus, each observation in each subset was predicted using a model that was built on the other subsets. The goal is to obtain a more realistic picture of the predictive power of the models as the goodness of fit tends to give optimistic results.

	R^2 adjusted	**Mean ARE**	**Mean MRE**	**Mean MRE'**
MARS model	0.78	5.33	1.05	0.58
Log-Linear model	0.56	6.49	0.78	0.76

Table 6: Comparison of goodness of fit

Cross-validation yields a R^2 of 0.532 and 0.647 between actual and predicted defects, for the log-linear and the MARS models, respectively. It is clear that a better goodness of fit is obtained with MARS, based on the exact same data. However, we can see that the MARS R^2 is, as expected, significantly lower that the goodness-of-fit R^2 (0.785) when running a cross-validation. This is not the case of the log-linear model, probably because it is based on less covariates (2 and 9, for the log-linear and MARS models, respectively) and therefore yields more accurate estimated regression coefficients. In general, we have to expect that MARS models show more covariates and this may be a significant drawback, depending on the data set size.

In addition, looking at the distributions of the difference between actual and predicted defects showed clearly that the log-linear model is biased in the sense that it tends to underpredict the number of defects. But because of size constraints, the reader is referred to [1] for further details on this matter.

Based on the discussions above, the MARS model seems to provide a better basis for interpreting the impact of various factors on code inspection effectiveness. This is discussed below where we focus our attention on interactions between factors, as modeled by the MARS model, and their implications in terms of decision making and gaining a better understanding of the code inspection process.

6.2 Model Interpretation

To help interpreting the models, we visualize 2-way interactions between independent variables.

Figure 2 is a typical example of two-way interactions that can be observed by graphical means from a MARS model. This figure represents the model predicted surface for the dependent variable (i.e., number of defects detected) when only considering the interaction effect of two variables. In other words, the 3-D graph captures only part of the model's effect, i.e., the interaction effect of two variables that contributes to the final model defect prediction. More precisely, in Figure 2, the "contribution" axis is in this case -0.037*BF14 - 0.033*BF30 and the other axes are simply the variables involved in the interaction terms: Rate and Effort. The MARS tool shifts values on the contribution axis so that the minimum value is 0. Color codes (here shown as gray scales) represent different contribution value intervals. Though the absolute values on the contribution axis are not easy to interpret and not of interest here, the shape of the surface modeling the interaction allows us to better understand how the effectiveness factors interplay. Note that the MARS tool only displays the part of the space that is populated by observations and some surfaces may appear truncated.

A grid is also drawn to help the reader get a better sense of the modeled surface in three dimensions. Higher-level interactions may be present in the data, but they are difficult to visualize and interpret. We will not investigate them here.

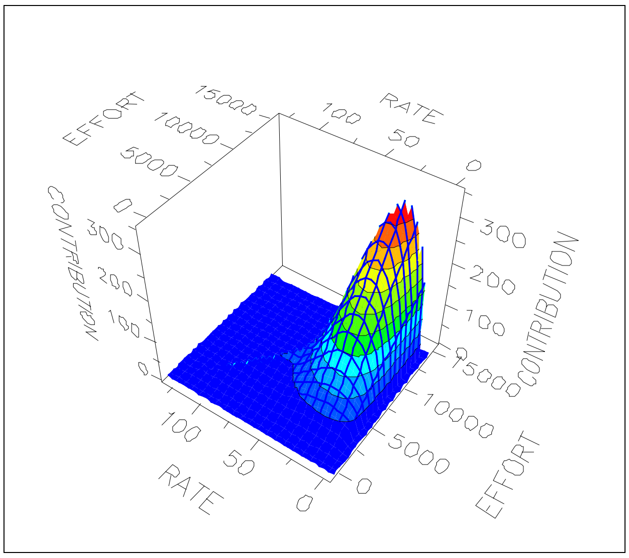

Figure 2: Interaction between Rate and Effort

From Figure 2, we can see that the interaction of Rate and Effort indicates an interval (roughly between 25 and 50 lines per hour) on the inspection rate measurement scale that is optimal, i.e., an increase in effort provides a higher pay-off in that interval. It would therefore be desirable that inspection rates remain in that interval for maximum return on investment. However, in practice, other factors may come into play, like the time and people actually available to perform an inspection on an artifact to be certified before a deadline.

It is now interesting to compare our optimal rates with the existing literature. Ebenau [7] reports an optimal rate of 150 lines per hour. It is however reported that the rates were not planned and the smallest observed rates were slightly above 50 lines per hour. Moreover, he simply used a linear regression between rate and defect density to estimate this “optimal” rate. It is worth noting that the resulting correlation coefficient of this regression was weak and its significance mainly due to two outliers. Another major difference with our inspections is that the programming language inspected was C. Gilb et al [16] found, based on numerous experiences, that effective individual inspections of software development task products usually lies between 0.5 to 1.5 pages an hour, where a page is roughly around 50 lines. They state that, though such slow rates shock people initially, we have to remember that inspections include cross-checking against check-lists and other documents and may involve several passes where inspectors focus on different types of faults. In light of the existing reported rates, and considering their level of uncertainty, the optimal rate determined by the MARS analysis is within plausible range.

Ebenau [7] notes that we should expect optimal rates to vary across organizations and inspections as the type of work products, their complexity and size, and the expertise of the inspectors vary significantly. It is therefore important for organizations not to rely on published rates to plan their inspection but to identify their own, optimal rates. MARS can be of assistance in doing this.

In Figure 3 we can see that for a given effort value, the higher DLOC (i.e., LOC changed, added, or deleted), the more defects, and therefore the higher the impact of effort on defect detection. This is visible by observing how the contours change as DLOC increases. This is a plausible trend: the larger the change, the higher the number of defects introduced, the more defects detected for a given effort value. This result reinforces the evidence that the MARS model is a plausible one. Also, one may predict, based on the amount of change in the code, the number of resulting defects to be expected and derive the resulting correction effort. Decisions to implement changes and plan change effort can be based on impact analysis (to estimate Dloc) and the use of the MARS model.

Another interesting observation is that as effort increases, the impact of DLOC on the number of defects increases up to a point, and then decreases. Though not everything can be explained in such an exploratory data analysis, this might reflect the fact that when a certain amount of effort–which is strongly related to effort per participant in our case–goes below or over a certain threshold (roughly below or above 2000 or 6000 man minutes, respectively), inspections tend to be less effective. In the former case, people may not have the time to really understand the documents they are inspecting and find fewer defects. The latter case is somewhat more complex to understand (e.g., may be due to fatigue effects in reported in [16]), is based on fewer observations, and should be the object of further enquiry. A strange surface shows up for high effort values and low DLOCs, but it very likely spurious as we have relatively few observations in that area. That might be, for example, the result of poor quality data collection or an idiosyncrasy of the MARS method, but it is hard to say. In general, regardless of the method used for exploratory analysis, we cannot expect to explain all observed trends at the smallest level of detail.

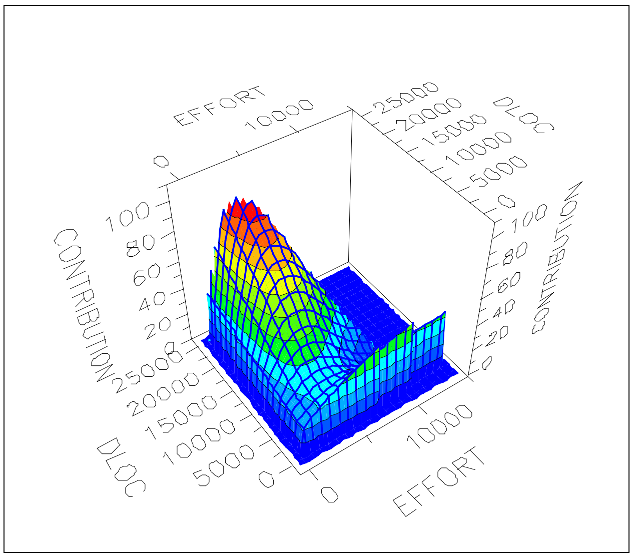

Figure 3: Interaction between DLOC and Effort

From Figure 4, we see that when the number of participants increases, more defects are found at a constant effort level. This is visible from looking at how the contours bend downward as the number of participants increases. On the range we can observe in our dataset (1 to 17), increasing the number of participants seems to increase inspection effectiveness. As mentioned before and reported in [10], this is likely due to the fact that with a larger number of participants, the inspection is more likely to include people with the right expertise. Now, a cost-benefit analysis, including the cost of additional participants in the equation, would be necessary to make useful recommendations.

For looking at the interactions in the figures above, we can first conclude that interactions seems to be key in modeling the effects of typical inspection factors on code inspections. A model that does not account for such interactions, such as log-linear regression models, is likely to be inadequate, as supported by the cross-validation results above. Common sense can also provide us with relatively straightforward explanations for many of these interactions.

We have seen above that many of the interactions modeled by MARS may be extremely useful to gain understanding about the inspection process and provide decision support. From Figures 2, 3, and 4, respectively, we can decide about the inspection rate (Rate), the maximum size of the documents to inspect[6], or the number of participants (Particip). The MARS model is therefore not only useful for planning purposes (e.g., inspection and correction effort planning) but also to help management decisions regarding inspections' effec-

[6] based on their corresponding effort, using the optimal inspection rate, and in order to prevent fatigue effects

tiveness. The results presented here cannot be systematically generalized to other environments. But similar data collection and analysis procedures should help every organization make its own, optimal decisions.

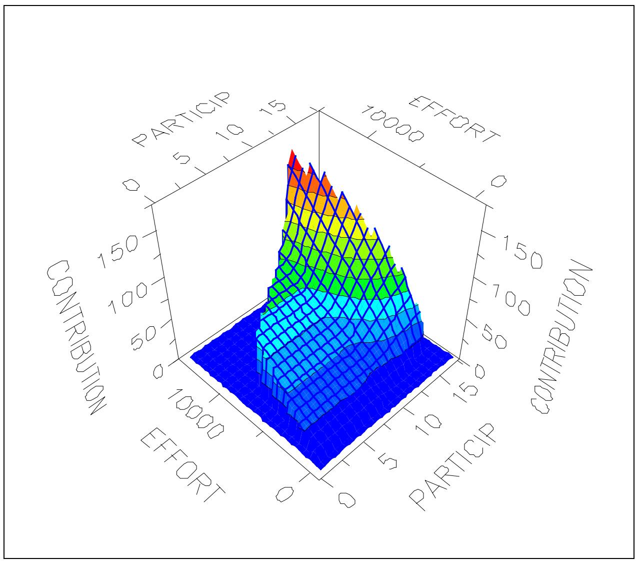

Figure 4: Interaction between Participants and Effort

7 Conclusions

This paper has focused on investigating the impact of a number of inspection factors, such as effort or inspection rate, on inspection effectiveness. A substantial amount of data was collected on code inspections and a novel exploratory modeling technique was used to improve our understanding of the underlying structures in the data: Multivariate Adaptive Regression Splines (MARS).

The results have shown that inspection effort and rates were very important defect detection drivers. Their effect is, however, interacting. That is, the impact of effort on effect detected is optimal within a certain code inspection rate (roughly 25 to 50 lines of code an hour). In addition, we have seen that the size of changes, the size of inspected artifacts, and the number of participants all interact with effort and are also important to predict defects detections.

From a more general standpoint, MARS has helped us better understand and uncover the complex relationships and interactions that exist in inspection data. The MARS model turned out to be a richer model, accounting for more factors, than the linear model we developed. In most cases, simple intuitive explanations could be given for interactions. Though not every result can be readily explained, MARS has nevertheless shown to be a useful exploratory data analysis tool in our context. MARS also seems to be a potentially useful technique to obtain more accurate models than with standard linear regression analysis. However, as discussed in [1], the dataset has to be large enough and the underlying structure in the data has to warrant such an analysis: strong non-linearity and interactions must be present.

In terms of applications, the model we have presented here can be used for a number of purposes. First, with respect to planning, the number of defects can be predicted for a project's planned inspections. Change effort can then be derived and planned for. Second, in terms of quality control, management may use MARS results to determine reasonable inspection effort, numbers of participants, and rates, thus maximizing inspection effectiveness.

It is interesting to note that our MARS model for code inspections supports some previous empirical results and hypotheses in the inspection literature. As expected and reported by Christenson et al [3], inspection effort plays also a very important role. Both the document and change size of the code (Loc, Dloc) plays here a significant role and this is similar to what both Christenson et al. and Ebenau [7] reported. Ebenau estimates inspection size by considering new and changed material and the interfaces to other work products. That latter attribute is missing in our dataset and should probably be part of our future data collection. The number of participants and sessions also shows a significant impact as in the work by Porter et al [10]. The inspection rate shows to be a significant factor as reported by Ebenau [7] and Gilb et al [16].

Though some of the results here are common with other reported studies, they cannot be readily generalized to other environments. However, the type of data collection and analysis that was performed in this study can be reused in any environment where inspections need to be better understood and controlled. Our study, performed in a representative development environment, has shown that it was practically feasible to undertake such measurement and obtain useful, interpretable models. Furthermore, from a practical and subjective standpoint, the feedback we received from practitioners and quality engineers was clearly positive.

8 Acknowledgements

We would like to thank Mr. Karl for his crucial support during the case study. Dan Steinberg, from Salford Systems, has been very helpful in answering crucial questions regarding the MARS tool. The graphs in this paper were generated using MARS, Version 1.0, from Salford Systems (www.salford-systems.com). Lionel Briand was partially supported by NSERC, the Canadian National Science and Engineering Research Council.

9 References

[1] Lionel C. Briand, Bernd Freimut, Ferdinand Vollei, Using Multiple Adaptive Regression Splines to Understand Trends in Inspection Data and Identify Optimal Inspection Rates, International Software Engineering Research Network (ISERN) Technical Report ISERN-00- 07, www.iese.fhg.de/network/ISERN/pub/isern.biblio.html, 2000

[2] F. O. Buck, Indicators of Quality Inspection, Tech. Report TR21.802, IBM Corp., 1995

[3] D. A. Christenson, H. T. Steel, and A. J. Lamperez, Statistical Quality Control Applied to Code Inspections, IEEE Journal Selected Areas in Communication, vol. 8, pp. 196-200, Feb. 1990

[4] Davidson, R., McKinnon, J., Several Tests for Model Specification in the Presence of Alternative Hypotheses. *Econometrica, vol. 49, no. 3* (1981), 781-93.

[5] R. D. De Veaux, D. C. Psichogios, L. H. Ungar, A Comparison of two Nonparametric Estimation Schemes: MARS and Neural Networks, Computers Chemical Engineering, Vol 17, N 8, pp. 819-837, 1993

[6] W. Dillon, M. Goldstein, Multivariate Analysis: Methods and Applications, Wiley, 1984

[7] R. G. Ebenau, Predictive Quality Control with Software Inspections, Cross Talk, The Journal of Defense Software Engineering, vol. 7, pp. 9-16, June 1994

[8] J. Friedman, Multivariate Adaptive Regression Splines, The Annals of Statistics, vol. 19, pp. 1-141, 1991

[9] W. H. Humphrey, A Discipline for Software Engineering, Addison-Wesley 1995

[10] A. A. Porter, H. Siy, and L. G. Votta, A Review of Software Inspections, Software Process, Advances in Computers Series, Academic Press, Marvin Zelkowitz Editor, May 1996

[11] O. Laitenberger and J.-M. DeBaud, Perspective-based Reading of Code Documents at Robert Bosch GmbH, Information and Software Technology, vol. 39, pp. 781-791, Mar. 1997

[12] O. Laitenberger, J. DeBaud: An Encompassing Life-Cycle Centric Survey of Software Inspection, Journal of Systems and Software, vol. 50, no. 1, 2000.

[13] A. A. Porter, H. P. Siy, C. A. Toman, and L. G. Votta, An Experiment to Assess the Cost-Benefits of Code Inspections in Large Scale Software Development, IEEE Transactions on Software Engineering, vol. 23, pp. 329-346, June 1997

[14] P. J. Fowler, In-Process Inspections of Workproducts at ATT, ATT Technical Journal, vol. 65, pp. 102-112, mar 1986

[15] N. E. Fenton and S. L. Pfleeger, Software Metrics - A Practical and Rigorous Approach. International Thomson Computer Press, 2nd edition ed., 1996.

[16] T. Gilb, D. Graham, Software Inspection, Addison-Wesley, 1993

Document Information

Title:	Using Multiple Adaptive Regression Splines to Support Decision Making in Code Inspections
Date:	January 1, 2001
Report:	IESE-063.00/E
Status:	Final
Distribution:	Public