Journal of Statistics Education

Evaluating Aptness of a Regression Model

Jack E. Matson, Department of Decision Sciences and Management, Tennessee Technological University

Brian R. Huguenard, Department of Decision Sciences and Management, Tennessee Technological University

Journal of Statistics Education Volume 15, Number 2 (2007),
http://www.amstat.org/publications/jse/v15n2/matson.html

Key Words: data transformation; residual analysis; linear model assumptions; linear regression

Abstract

The data for 104 software projects is used to develop a linear regression model that uses function points (a measure of software project size) to predict development effort. The data set is particularly interesting in that it violates several of the assumptions required of a linear model; but when the data are transformed, the data set satisfies those assumptions. In addition to graphical techniques for evaluating model aptness, specific tests for normality of the error terms and for slope are demonstrated. The data set makes for an excellent case problem for demonstrating the development and evaluation of a linear regression model.

1. Introduction

For any organization involved with the creation of computer software, the ability to predict development effort plays a key role in the effective management of the software development process. Regression models based on a software metric called *function points* are an important tool used in the estimation of software development effort. Through these regression models a manager can compare estimated development effort across multiple proposed projects and make intelligent decisions concerning scheduling and priority of the projects. In this paper we develop and evaluate a linear regression model that predicts software development work hours based on a function point measure of software size.

1.1 Function Point Analysis

Function points are a standard metric used for estimating the size of software development projects (International Function Point Users Group, 2005). Function point analysis is a structured method of estimating the size and complexity of a software system. This estimation process is based on the breaking down of a system into smaller components called function points, which measure different types of business functionality delivered by the system to the end user. Function points provide a means of measuring the system functionality perceived by the end user, and are independent of the technology (computer language, operating system, etc.) used to implement the system. Once a count of the function points for a proposed system has been developed, the count can be compared to historical function point counts for completed systems. Using the known development times of the completed systems, an estimate of the development effort required for the proposed system can be generated.

When a new software project is being planned, the number and types of function points for the project can be estimated from the design specifications, thus making it possible to estimate development effort during the early phases of project planning. In addition, since the function point count is derived from the design specifications, any changes in the specifications (which occur frequently during software development) can be easily accounted for in the estimate of development effort.

There are five basic types of function points: external inputs (data coming from the user or some other system), external outputs (reports or messages going out to the user or some other system), external inquiries (queries coming from outside the system which result in a report being sent to the requestor), internal logical files (data files that reside within the boundaries of the system), and external interface files (data files that reside outside the boundaries of the system). Standardized criteria have been developed to allow the consistent identification and categorization of function points from the design specifications of a proposed system, or from the actual features of an existing system (International Function Point Users Group, 2005). Once an initial count of function points has been generated, it is adjusted to allow for the overall complexity of the system, using a standardized system of weights that account for 14 different system factors (for a more detailed account of the adjustment process, see Function Point Counting Practices Manual, 2001). The final adjusted function point measure (FP) is then complete, and serves as an objective measure of the system's size and complexity.

1.2 The Data Set

The data used in this paper are from 104 software projects completed at AT&T from 1986 through 1991. For each project five values are recorded: the adjusted function point count, the actual work hours devoted to completing the project, the operating system used, the database management system used, and the programming language used. The adjusted function point count is the only predictor variable discussed in detail in this paper. One unique aspect of this data set is the fact that the projects represent a total of 7,981 man-months or 665 man-years of effort. This is a very large set of software projects. The project data represent both new project

development and project enhancements, and the data are not ordered by time or any other variable. Figures 1 and 2 show the distribution of function points and work hours.

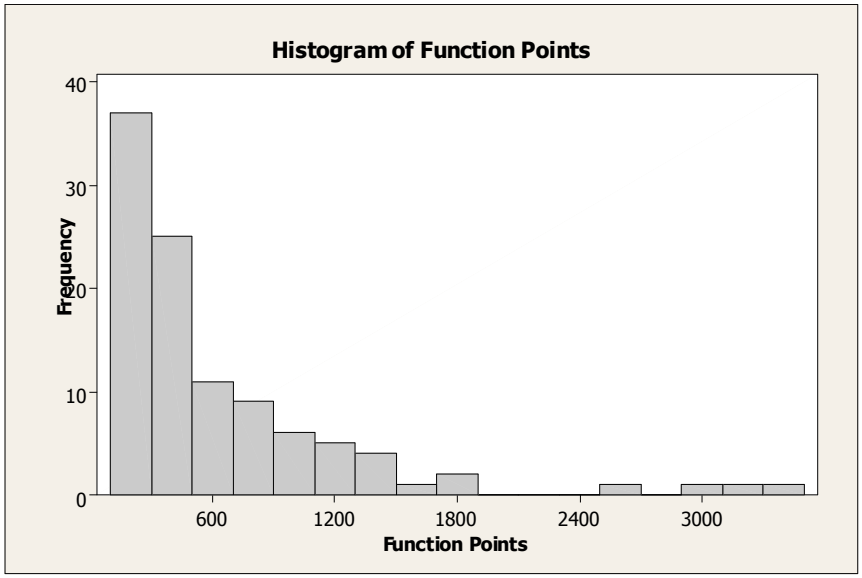

Figure 1. Distribution of Function Points

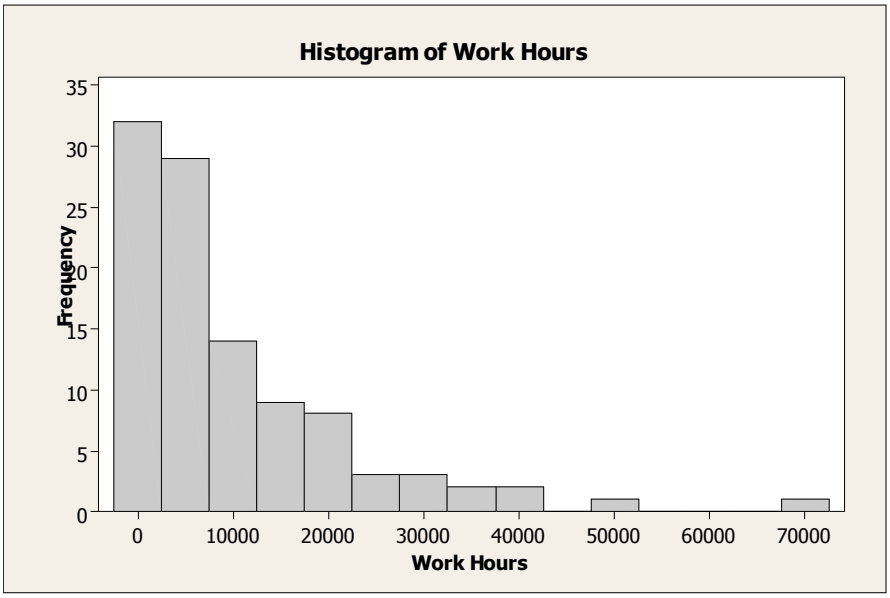

Figure 2. Distribution of Work Hours

1.3 The Objective

This paper presents a methodology for the development of a linear regression model for estimating software development effort using historical function point data. In developing a useful regression model, a number of concerns must be addressed. The first is model adequacy, or explanatory power of the independent variable in accounting for the variability of the dependent variable. This is typically measured by the coefficient of determination, R^2. A large value of R^2 is a good indication of how well the model fits the data. However, it is not the only measure of a good model when the model is to be used to make inferences. Linear regression models are tied to certain assumptions about the distribution of the error terms. If these are seriously violated, then the model is not useful for making inferences. Therefore, it is important to consider the aptness of the model for the data before further analysis based on that model is undertaken.

Model aptness refers to the conformity of the behavior of the residuals to the underlying assumptions for the error values in the model. When a regression model is built from a set of data, it must be shown that the model meets the statistical assumptions of a linear model in order to conduct inference. Residual analysis is an effective means of examining the assumptions. This method is used to check the following statistical assumptions for a simple linear regression model:

1. the regression function is linear in the parameters,
2. the error terms have constant variance,
3. the error terms are normally distributed, and
4. the error terms are independent.

If any of the statistical assumptions of the model are not met, then the model is not appropriate for the data. The fourth assumption (independence of error terms) is relevant when the data constitute a time series. Since the data in this paper is not time series data, we do not test for independence of the error terms.

Residual analysis uses some simple graphic methods for studying the aptness of a model, as well as some formal statistical tests for doing so. In addition, when a model does not satisfy these assumptions, certain transformations of the data might be done so that these assumptions are reasonably satisfied for the transformed model.

2. Methodology

The following procedure was used to develop and evaluate the regression model:

1. Plot the dependent variable against the (various) predictor variable(s).
2. Hypothesize a model.
3. Check if the statistical assumptions for the regression model are reasonably satisfied. If so, an appropriate model has been identified. If not, repeat steps (2) and (3).

2.1 Straight Line Model

The scatter plot shown in Figure 3 indicates that a simple linear regression model might be appropriate for our project data. In particular, the fitted regression model is

$$E_{est} = 585.7 + 15.124 \times FP \qquad \text{(Model 1)}$$

where E_{est} is the estimated development hours and FP is the size in function points. The coefficient of determination (R^2) for this model is 0.655. The Minitab results for the simple linear regression model are shown in Table 1.

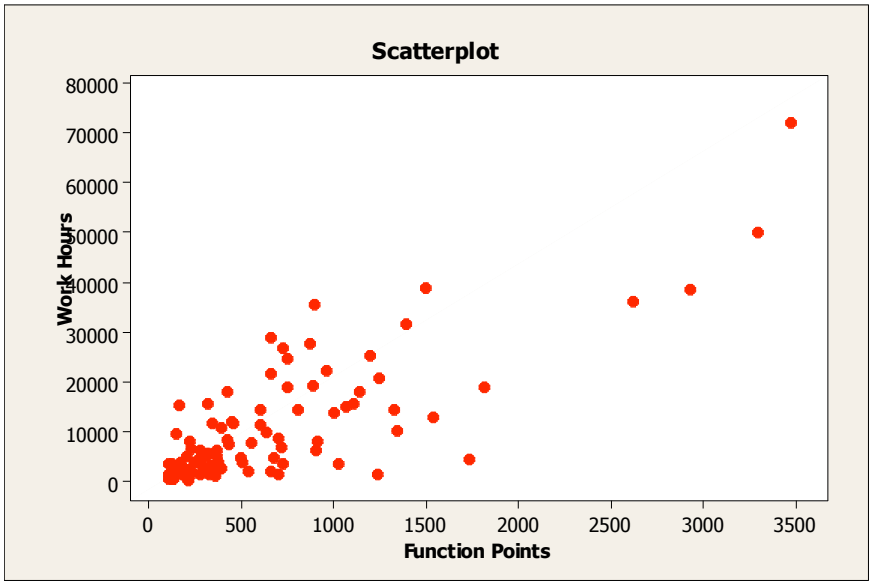

Figure 3: Work Hours vs. Function Points

Predictor	Coef	SE Coef	T	P
Constant	585.7	965.5	0.61	0.545
Function Points	15.124	1.086	13.93	0.000

Table 1: Minitab Results for Model 1

To determine the adequacy of the model, residual analysis should be performed. Figure 4 shows the plot of residuals against the independent variable (function points).

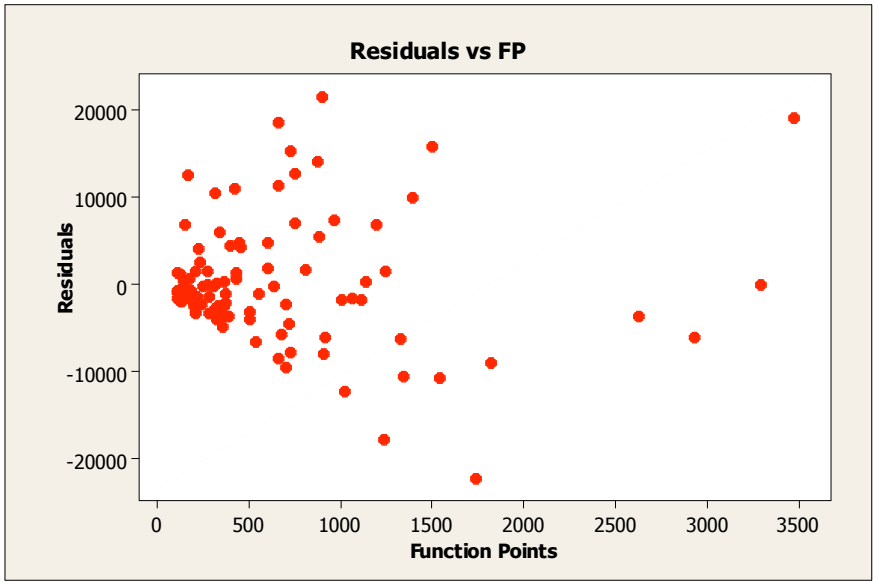

Figure 4: Residuals vs. Function Points for Linear Fit

The spread of residuals around zero increases as function points increase for this plot. Ideally, the residuals should fluctuate in a more or less uniform band around zero. The residuals shown in Figure 4 get larger as software size in function points increases, an indication that the error variance is not constant. The project data violate the equal variance assumption.

A normal probability plot of the residuals can be used to test the normality of the error terms. The normal probability plot in Figure 5 is not linear, an indication that this assumption is also being violated. A Kolmogorov-Smirnov (K-S) test for normality resulted in a p-value less than .010, a second indication that the error terms are not normally distributed.

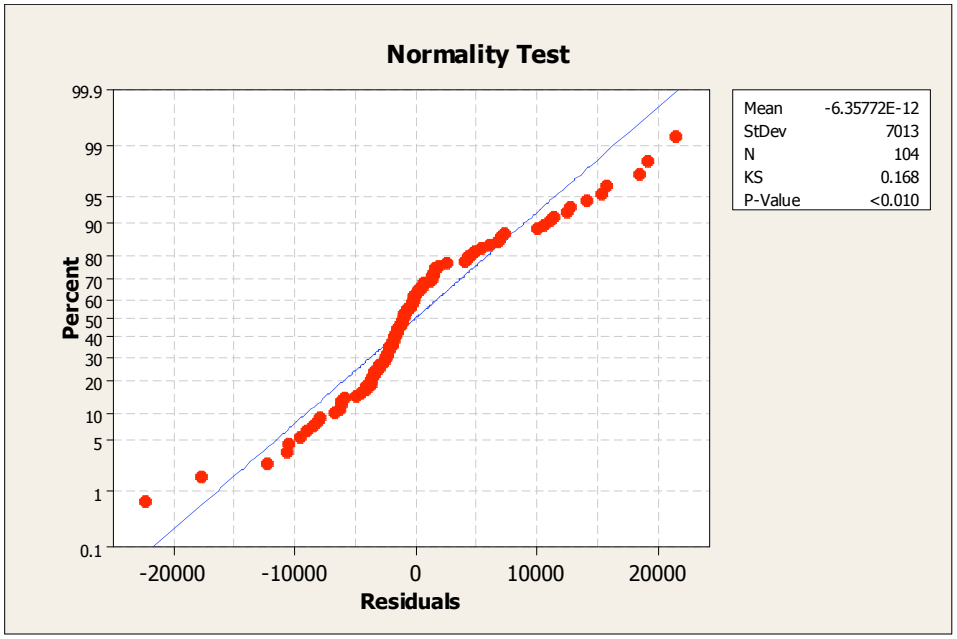

Figure 5: Normal Probability Plot of Residuals When Fitting by a Line

---

Based on the results of the residual analysis, if inference about development effort is to be conducted, then fitting function points by simple linear regression model is inappropriate. The model violates the constant error variance and normality assumptions.

2.2 Regression Model Using Transformed Data

Some sort of transformation is often used to deal with the lack of constant variance. A common way of stabilizing the variance is to apply a logarithmic transformation to the data. Applying this transformation to the dependent and independent variables and plotting the transformed data (Figure 6), a linear relationship can be seen between the natural logarithm of development effort and the natural logarithm of function points.

The regression model built from the transformed project data is as follows:

$$\ln(E_{est}) = 2.5144 + 1.0024 \times \ln(FP) \quad \text{(Model 2)}$$

where $\ln(E_{est})$ is the natural logarithm of estimated development hours and ln(FP) is the natural logarithm of function points. The coefficient of determination (R^2) for this model is 0.534. The Minitab results for this regression are shown in Table 2.

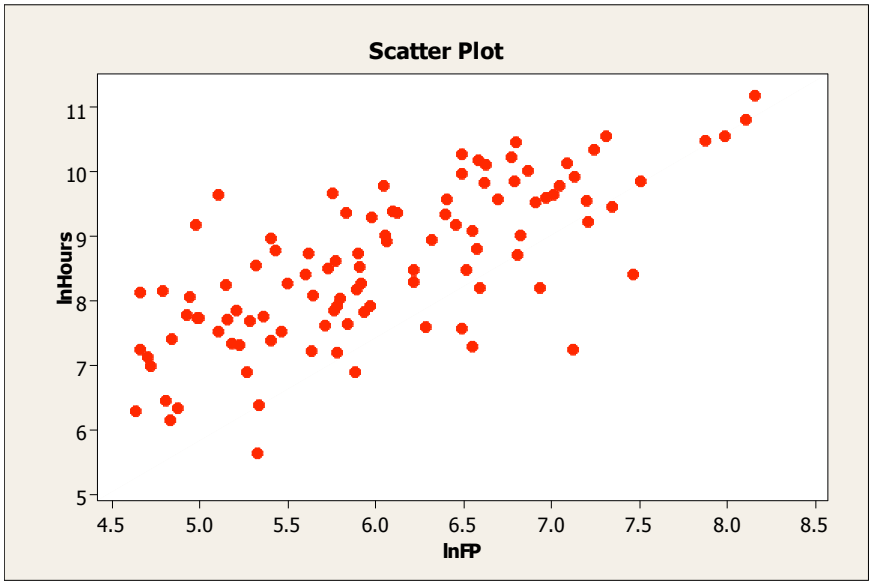

Figure 6: Natural Logarithm of Work Hours vs. Natural Logarithm of Function Points

Predictor	Coef	SE Coef	T	P
Constant	2.5144	0.5660	4.44	0.000
Ln_FP	1.00239	0.09274	10.81	0.000

Table 2: Minitab Results for Model 2

The next step is to check the equal variance and normality assumptions. A scatter plot of the residuals against the independent variable (natural logarithm of function points) is shown in Figure 7. No pattern in the residual data is apparent. The logarithmic transformation resolved the problem with increasing variance of error terms that existed with the project data in its original form.

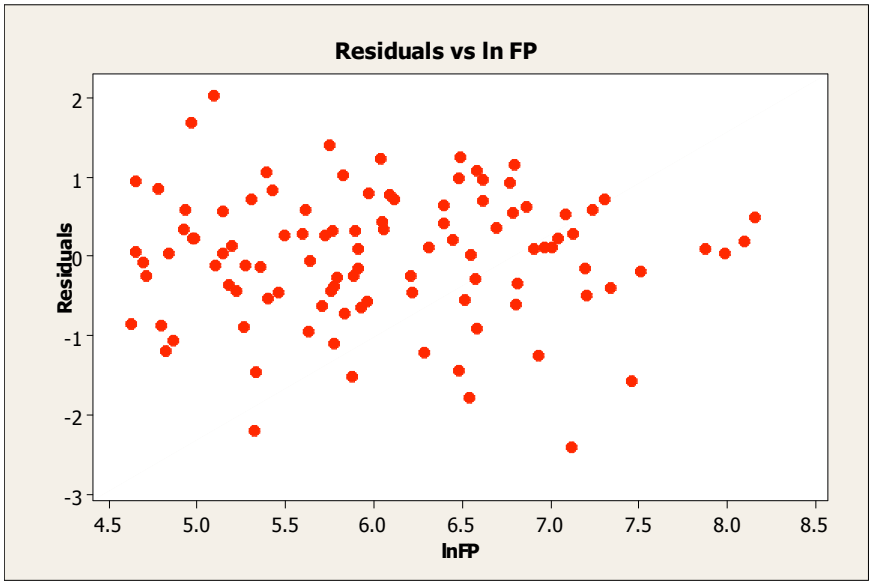

Figure 7: Residuals Versus Natural Logarithm of Function Points

To check the error terms for normality, a histogram of the residuals and a normal probability plot of residuals are shown in Figures 8 and 9, respectively. The normal probability plot is nearly linear, indicating that the error terms are normally distributed. The shape of the histogram supports this conclusion. In addition, a K-S test for normality resulted in a p-value greater than .150. The K-S test provides further support for the error terms being normally distributed.

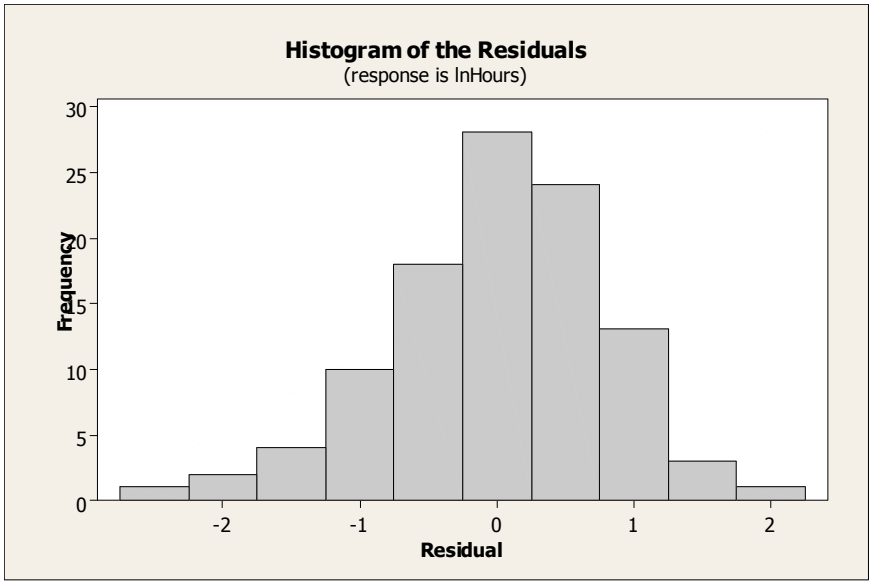

Figure 8: Histogram of Residuals – Transformed Data

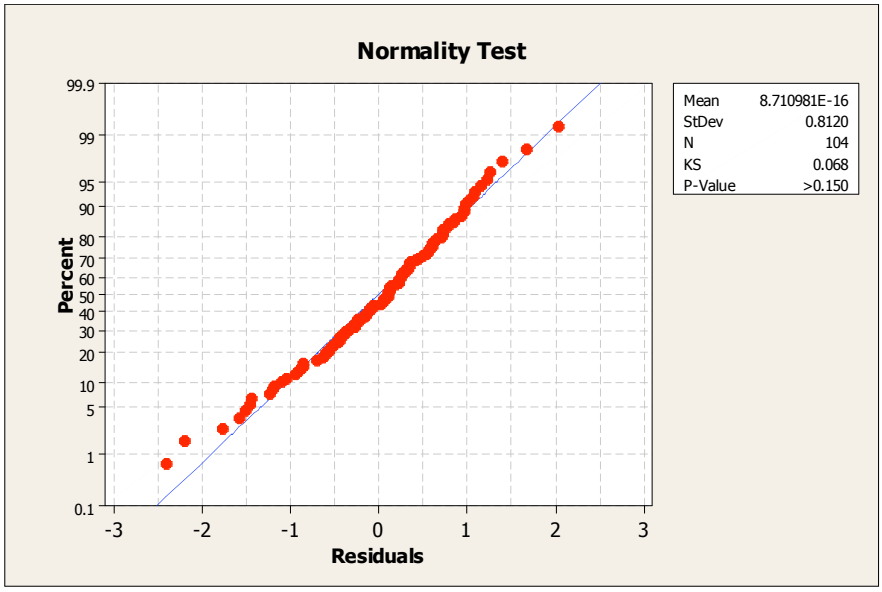

Figure 9: Normal Probability Plot of Residuals When Fitting by a Line

---

Therefore, Model 2 apparently satisfies the assumptions of equal variance and normality. Thus, Model 2 is appropriate for the transformed project data.

Since the assumptions of our regression model are reasonably satisfied, we may now perform inference for our model. Appendix A, for instance, shows the results of a test for the slope $\beta_1 = 0$ versus $\beta_1 \neq 0$ (Neter, Wasserman, and Kutner, 1985). We conclude that $\beta_1 \neq 0$ and a 95% confidence interval for the slope β_1 is $0.818 < \beta_1 < 1.186$.

For an example of applying Model 2 to estimate software development effort, suppose a system has a software size of 381 function points. To estimate development effort, the function points must be transformed: ln(FP) = ln (381) = 5.9428. Using the regression equation for Model 2,

$$\text{Ln}(E_{est}) = 2.5144 + 1.0024 \times (5.9428) = 8.4715.$$

Taking the inverse logarithm of 8.4715, the estimated development effort for the system is 4776.5 hours or 36.7 man-months. One man-month is defined as 130 hours.

2.3 Using Prediction Intervals

The regression model (Model 2) derived from the logarithmic transformed data satisfies the assumptions of a linear model and is an appropriate simple linear regression model of the transformed project data. However, some problems exist in this software project data that make statistical models difficult to use. Software data variance tends to increase with increasing

project size, and the data typically have a relatively large number of outliers. The project data in this study reflected these characteristics, although the effect of the outliers was lessened by the logarithmic transformation on the project data. The point estimate produced by the cost estimation model is misleading without some measure of the variability that exists in the sample of projects. Specifying an upper and lower bound for the estimate in the form of a prediction interval provides this needed measure of variability.

The coefficient of determination for this model was 0.534, which means that almost half of the variability cannot be explained by the model. Basing an estimate solely on the results of this model would be extremely risky. However, the model is not without significance and usefulness because it leads to a measure of the uncertainty that exists in the point estimate. The primary cost estimate should always be based on a detailed analysis of the work to be performed. The model developed in this study could be used to provide further support of the detailed cost estimate and provide a measure of uncertainty for the effort estimate.

Prediction intervals for Model 2 (Appendix B) are apparently narrower for small projects and get wider as software size in function points increases. For example, the smallest project in the sample of project data is 119 function points. The point estimate of development effort for this project is 11.4 man-months. Considering that the corresponding 90% prediction interval is 2.91 man-months to 44.98 man-months the practical application of this model becomes questionable.

3. Classroom Applications

The data has been used in a sophomore-level engineering statistics class to demonstrate the development of linear regression models. The first assignment used the data to develop a simple linear regression model that could be used to predict development effort using function points. The assignment requires that students plot the data to see if a linear model seems appropriate. The students are then required to determine the regression model, to explain the meaning of the coefficient of determination, and to use an appropriate statistical test for a non-zero slope. The students are asked to perform residual analysis to verify that the assumptions of a linear model are satisfied. They must describe each assumption and explain the results of their analysis. When they discover that some of the assumptions have been violated, they are asked to try a transformation of the data. The text (Devore, 1995) for the class had a table of useful transformations.

The second assignment dealt with developing a multiple regression model. In addition to function points, the data also includes three other explanatory variables: operating system, database management system, and programming language. This requires the use of indicator variables. Inclusion of these variables results in a modest improvement over the simple linear regression model, increasing the R-Squared value from .534 to .694. The Minitab results for a multiple regression model are shown in Table 3. (The variable Unix, for example, is equal to 1 if the Unix operating system was used and is equal to 0 if the Unix operating system was not used.) While this paper does not address multiple regression, the topic is discussed, for example, by Chu (2001).

```
The regression equation is
ln(WH) = 2.48 + 0.866 ln(FP) + 0.155 Unix + 0.990 IDMS + 1.35 IMS
   + 0.343 INFORMIX + 0.225 INGRESS + 0.509 COBOL + 0.121 PL1  - 0.230 C

      Predictor            Coef        StDev        T            P
Constant                    2.479       0.5159       4.81         0.000
ln(FP)                    0.86635       0.0796      10.88         0.000
Unix                       0.1549       0.3123       0.50         0.621
IDMS                       0.9899       0.2015       4.91         0.000
IMS                        1.3473       0.2516       5.35         0.000
INFORMIX                   0.3434       0.2270       1.51         0.134
INGRESS                    0.2252       0.3033       0.74         0.460
COBOL                      0.5094       0.1798       2.83         0.006
PLI                        0.1215       0.2561       0.47         0.636
C                         -0.2296       0.2855      -0.80         0.423

S = 0.6583                          R-SQ = 72.0%          R-Sq(ajd) = 69.4%
```

Table 3: Minitab Results for Multiple Regression Model

4. Summary

Regression analysis was used to determine the relationship between actual software development effort and software size in function points. An initial linear regression model was built, and upon further investigation was found to violate the constant error variance and normality assumptions of a linear model. Techniques for discovering these violations were demonstrated, along with one possible method of rectifying the violations: using a logarithmic transformation of the data. After transforming the data a second regression model was developed and was shown to satisfy the previously violated assumptions. We concluded with the use of prediction intervals to demonstrate the practical limitations of the final model.

5. Getting the data

The data discussed in this article are contained in the .dat file aptness.dat at http://www.amstat.org/publications/jse/datasets/aptness.dat. The file aptness.txt at http://www.amstat.org/publications/jse/datasets/aptness.txt contains a description of the data and, in particular, a listing of the different variables.

Acknowledgements

The authors wish to acknowledge Linda Hughes and Mary Dale of AT&T for their assistance in providing the function point data set.

Test for non-zero slope for Model 2 using the natural logarithm of function points and the natural logarithm of development hours.

$$Y_i = \beta_0 + \beta_1 X_i + \varepsilon_i$$

where Y_i is the natural logarithm of estimated effort in hours and X_i is the natural logarithm of function points.

Test of Hypothesis:

H_0: $\beta_1 = 0$
H_1: $\beta_1 \neq 0$

Significance Level $\alpha = .05$

Test Statistic: $T = \dfrac{b_1}{S_1}$

where S_1 estimates the variance of the estimated slope b_1 for the project data.

Decision Rule: Reject H_0 if $|T| > t_{\alpha/2,n-2}$

Calculate the test statistic:

$$T = \frac{1.00239}{.09274} = 10.81$$

Conclusion:

$|T| = 10.81$ is greater than $t_{.025,102} = 1.9835$. There is sufficient evidence to reject the null hypothesis and conclude that $\beta_1 \neq 0$.

95% Confidence Interval for β_1:

$$b_1 - t_{.025,102}S_1 < \beta_1 < b_1 + t_{.025,102}S_1$$

$$1.00239 - (1.9835)(.09274) < \beta_1 < 1.00239 + (1.9835)(.09274)$$

$$.818 < \beta_1 < 1.186$$

Appendix A: Test for Non-Zero Slope

A 100(1-α)% prediction interval for Y is given by Y_L and Y_U where

$$Y_L = Y_h - t_{\alpha/2,n-2} MSE \sqrt{1 + \frac{1}{n} + \frac{(x_0 - \bar{x})^2}{(n-1)s_x^2}}$$

and

$$Y_U = Y_h + t_{\alpha/2,n-2} MSE \sqrt{1 + \frac{1}{n} + \frac{(x_0 - \bar{x})^2}{(n-1)s_x^2}}$$

Sample calculations of 90% prediction interval for Model 2 where
n = 104, x_0 = ln(119) = 4.7791, Y_h = 2.5144 + 1.0024 (4.7791) = 7.3050, s_x= .8669, $\bar{x}$ = 6.0423,
MSE = .6657, α = .10, and $t_{\alpha/2,102}$ = 1.6599

$$Y_L = 7.3050 - (1.6599)(.6657)\sqrt{1 + \frac{1}{104} + \frac{(4.7791 - 6.0423)^2}{(103)(.8669)^2}}$$

$$Y_U = 7.3050 + (1.6599)(.6657)\sqrt{1 + \frac{1}{104} + \frac{(4.7791 - 6.0423)^2}{(103)(.8669)^2}}$$

$$5.9363 < Y_h < 8.6737$$

Taking the inverse logarithm, the interval becomes

$$378.54 \text{ hours} < E_{est} < 5846.84 \text{ hours}$$

With 90% confidence, Model 2 predicts that the development of a system with software size of 119 function points will require somewhere between 378.54 and 5846.84 man-hours of effort. Using 130 man-hours per man-month, the prediction interval becomes 2.91 man-months to 44.98 man-months with a point estimate of 11.4 man-months.

Appendix B: Prediction Interval

References

Chu, Singat (2001), "Pricing the C's of Diamond Stones", *Journal of Statistics Education*, Volume 9, Number 2, http://www.amstat.org/publications/jse/v9n2/datasets.chu.html .

Devore, Jay L. (1995), *Probability and Statistics for Engineering and the Sciences*, 4th edition, Belmont, CA: Wadsworth Publishing Company.

International Function Point Users Group (2001), Function Point Counting Practices Manual, release 4.1.1, Princeton Junction, NJ.

International Function Point Users Group (2005), "About Function Point Analysis", http://www.ifpug.org/about/about.htm.

Neter, J., Wasserman, W., and Kutner, M. (1985), *Applied Linear Statistical Models*, Homewood, Illinois: Irwin.

Jack E. Matson
Department of Decision Sciences and Management
Tennessee Technological University
Johnson Hall
1105 North Peachtree Street
Cookeville, TN 38505-0001
U.S.A.
JEMatson@tnech.edu

Brian R. Huguenard
Department of Decision Sciences and Management
Tennessee Technological University
Johnson Hall
1105 North Peachtree Street
Cookeville, TN 38505-0001
U.S.A.
BHuguenard@tntech.edu.
