# Effort Estimation of Web Based Applications

A THESIS SUBMITTED IN PARTIAL FULFILMENT
OF THE REQUIREMENTS FOR THE DEGREE OF

**Bachelor of Technology**
**In**
**Computer Science and Engineering**

**By**
**Abhijit Ghosh**
**Roll No: 10306006**


Department of Computer Science and Engineering
National Institute of Technology, Rourkela

**May, 2007**

# Effort Estimation of Web Based Applications

A THESIS SUBMITTED IN PARTIAL FULFILMENT
OF THE REQUIREMENTS FOR THE DEGREE OF

**Bachelor of Technology**

**In**

**Computer Science and Engineering**

By

**Abhijit Ghosh**

**Roll No: 10306005**

**Under the guidance of**

**Prof. B.D.Sahoo**


Department of Computer Science and Engineering

National Institute of Technology

Rourkela

**May, 2007**


**National Institute of Technology**
**Rourkela**

# CERTIFICATE

This is to certify that the thesis entitled, "Effort Estimation Of Web Based Applications" submitted by Abhijit Ghosh in partial fulfillment of the requirements for the award of Bachelor of Technology Degree in Computer Science and Engineering at the National Institute of Technology, Rourkela (Deemed University) is an authentic work carried out by him under my supervision and guidance.

To the best of my knowledge, the matter embodied in the thesis has not been submitted to any other university / institute for the award of any Degree or Diploma.

Date: 10 May, 2007

Prof. B.D.Sahoo
Dept. of Computer Science and Engineering
National Institute of Technology, Rourkela
Rourkela - 769008

# ACKNOWLEDGEMENT

I am thankful to **Prof. B.D. Sahoo,** Professor in the department of Computer Science and Engineering, NIT Rourkela for giving me the opportunity to work under him and extending every support at each stage of this project work.

I would also like to convey my sincerest gratitude and indebtedness to all other faculty members and staff of Department of Computer Science and Engineering, NIT Rourkela, who bestowed their great effort and guidance at appropriate times without which it would have been very difficult on my part to finish the project work.

Date: May 10, 2007 Abhijit Ghosh

# CONTENTS


**A. ABSTARCT**

**B. LIST OF FIGURES**

**C. CHAPTERS**

**1. INTRODUCTION**

1.1 Introduction----------2

1.2 Difficulties in Software Effort Estimation----------3

1.3 Objective of the Estimate----------3-4

1.4 Benefits of Software Effort Estimation----------4

1.5 Effort Estimation For Web Based Hypermedia Applications ----------5

1.6 Conclusion----------6

**2. Regression Analysis : An Overview**

2.1 Introduction---------- 8-9

2.2 History of regression---------- 9

2.3 Definitions and notation used in regression----------9

2.4 Types of regression

2.4.1 Linear regression----------10

2.4.2 Nonlinear regression models----------11

2.4.3 Non-continuous variables----------11

2.4.4 Other models----------11

2.4.5 Nonparametric regression----------11

**3. Methodology----------12-14**

**4. Analysis and Results**

4.1 Multiple Linear Regression Analysis----------16--18

4.2 Stepwise Multiple Linear Regression Analysis---------- 18-19

4.3Polynomial Regression Analysis---------- 20-22

**5. Conclusion and Future Work---------- 23-24**

**D. REFERENCES**

# ABSTRACT

Countless organizations around the world have developed commercial and educational applications for the World Wide Web, the best known example of a hypermedia system. But developing good Web applications is expensive, mostly in terms of time and degree of difficulty for the authors. Our study tries to predict the effort needed for the development of web pages of a particular category, here we have restricted our self to the domain of news web sites.

We try to forecast the average effort required to code a page of a new site belonging to the same category based on the analysis of the data available from the existing sites. Here we have considered the data from Top Ten News Sites. The number of pages in the site , Average Number of Lines per Page, Average Number of Scripts per Page, Link Density, and Media Density are taken into account while predicting the effort required to code a page of the site. We consider the effort required to be directly proportional to the number of lines of code. It can be expressed to be in man hours only when we have information about the actual effort in man-hours required to code the site.

We use Multiple Linear Regression, Stepwise Regression and Polynomial Regression to analyze the data and obtain the graphs which a be used to predict the approximate effort required to code a web page of a news site.

Finally we devised a method to estimate the effort required or the number of lines of code required for a webpage of a news site. These results can be used to devise a standard for the coding of newer news web sites. If they are incorporated in to a web authoring software it would help the author to stick to the guidelines and the standards automatically.

# List of Figures

Figure 1: Steps Involved In Software Prediction

Figure 2: Residual Case Order Plot

Figure 3: Tool used for Stepwise Regression Analysis

Figure 4: Shows how the lines of code vary as a polynomial of degree 2 with the no of pages in the website

Figure 5: Shows how the lines of code vary as a polynomial of degree 4 with the no of links per page

Figure 6: Shows how the lines of code vary as a polynomial of degree 5 with the no of scripts per page

Figure 7: Shows how the lines of code vary as a polynomial of degree 5 with the no of images per page

# Chapter 1

## INTRODUCTION

1.1 Introduction

1.2 Difficulties in Software Effort Estimation

1.3 Objective of the Estimate

1.4 Benefits of Software Effort Estimation

1.5 Effort Estimation For Web Based Hypermedia Applications

1.6 Conclusion

## 1.1 Introduction

Effective software project estimation is one of the most challenging and important activities in software development. Let us first define what software is. Software is (1) instruction that when executed provide desired function and performance, (2) data structures that enable the programs to adequately manipulate information, and (3) documents that describe the operation and use of programs. The technological and managerial discipline concerned with systematic production and maintenance of software products that are developed and modified on time and with in the cost estimates is known as software engineering. The primary goals of software engineering are to improve the quality of software products and to increase the productivity and job satisfaction for software engineers .

Software cost estimation is the process of predicting the effort required to develop a software system. Estimating the cost of a software product is one of the most difficult and error–prone tasks in software engineering. It is difficult to make an accurate cost estimation during the planning phase of software development because of the large number of unknown factors at that time, yet contradicting practice often require a firm cost commitment as part of the feasibility study.

Accurate software cost estimate in project is necessary to develop a reliable software system. Underestimating a project leads to

- under-staffing it (resulting in staff burnout),
- under-scoping the quality assurance effort (running the risk of low quality deliverables), and
- setting too short a schedule (resulting in loss of credibility as deadlines are missed).

Over-estimating a project is likely to

- cost more than it should (a negative impact on the bottom line),
- take longer to deliver than necessary (resulting in lost opportunities), and
- delay the use of your resources on the next project.

## 1.2 Difficulties in Software Effort Estimation

Software effort estimating has been growing in importance up to today. When the computer era began back in the 1940's, there were few computers in use and applications were mostly small, one person projects. As time moved on, computers became widespread. Applications grew in number, size and importance; costs to develop software grew as well. As a result of that growth, the consequences of errors in software cost estimation became more severe too. Still today, a lot of cost estimates of software projects are not very accurate, mostly too low. This is not a surprising fact if we look at the various difficulties we have to face when estimating software costs. The by far greatest amount of the total costs of a project arises from the salaries of the personnel. Other costs, as license fees or new equipment for example, occur only once and are not too hard to estimate. The costs for the human workers on the other hand are highly correlated to the effort we need to perform the project. Therefore we have to get an accurate enough estimate of the total effort in order to make a reasonable estimate of the costs. The effort is estimated based on the size and complexity of the project, which both derive from the specification. Because the requirements of the software are likely to change, we have to take this into account too when estimating the effort. The big difference in productivity of software developers is one of the hardest problems to solve during the estimation process. An experienced developer will produce far more than a beginner. But because each project is unique, uses it's own tools and languages, the experience level of the development team is hard to judge. Another problem appears when humans ere estimating.

We all tend to underestimate immaterial things like software. This is also a reason why software is considered to be expensive by most people, although there is nothing to compare its costs with. Today's world would not be the same if there was no software.

## 1.3 Objective of the Estimate

Software cost estimating performs more than the name indicates. It’s not just the total costs that are of interest but a lot more. Quite a few things other than costs are being estimated during the process. Of main interest are the following :

- Size off all deliverables
- Staff needed
- Schedule
- Effort
- Cost to develop
- Cost for maintenance/enhancement
- Quality
- Reliability

But in this paper we are mainly concerned with the software effort in person-month, project duration in calendar time and cost to develop in dollar (or local currency).

## 1.4 Benefits of Software Effort Estimation

The major benefits of a good software cost estimation model that it provides a clear and consistent universe of discourse within which to address a good many of the software engineering issues which arise throughout the software life cycle .

A well-defined software cost estimation model can help avoid the frequent misinterpretations, underestimates, over expectations, and outright buy-ins which still plague the software field. In a good cost estimation model, there is no way of reducing the estimated cost without changing some objectively verifiable property of the software project. This does not make it impossible to create an unachievable buy-in, but it significantly raises the threshold of credibility.

A related benefit of software cost estimation technology is that it provides a powerful set of insights on how a software organization can improve its productivity. Many of a software cost model's cost-drivers attributes are management control labels: Use of software of tools and modern programming practices, personnel capability and experience and available computer speed, memory and turn around time, software reuse. The cost model helps us determine how to adjust these management controllable to increase productivity, and further provides and estimated

of how much a productivity increase we are likely to achieve with a given level of investment.

## 1.5 Effort Estimation Of Web Based Hypermedia Applications

By using measurement principles to evaluate the quality and development of existing Web applications, we can obtain feedback that will help us understand, control, improve, and make predictions about these products and their development processes. Prediction is a necessary part of an effective software process, whether it be authoring, design, testing, or Web development as a whole. As with any software project, having realistic estimates of the required effort early in a Web application's life cycle lets project managers and development organizations manage resources effectively. As shown in Figure 1, the prediction process involves capturing data about past projects or past development phases within the same project, identifying size metrics and cost drivers and formulating theories about their relationship with effort, generating prediction models to apply to the project, and assessing the effectiveness of the prediction models. This article focuses on effort prediction for the design and authoring processes. Design covers the methods used for generating the structure and functionality of the application, and typically doesn't include aspects such as hypermedia application requirements analysis, feasibility consideration, and applications maintenance. In addition, our design phase also incorporates the application's conceptual design, reflected in map diagrams showing documents and links. The data we used to generate our prediction models came from a quantitative case study evaluation, in which we measured a set of suggested size metrics and cost drivers for effort prediction. Some of our complexity metrics are adaptations from the literature of software engineering and multimedia. The metrics we propose characterize Web application size from two different perspectives—length and complexity. Our prediction models derive from statistical techniques—specifically linear regression and stepwise multiple regression. We also use Polynomial regression to study the effect of the predictors individually.

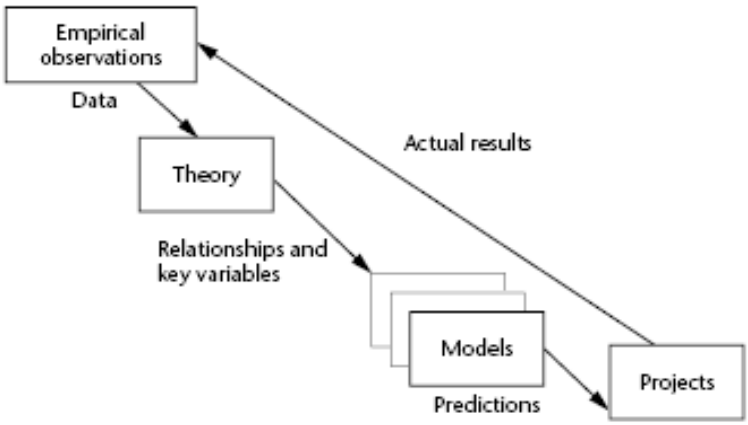


Figure 1

## 1.6 Conclusion

Software cost estimation and particularly that of web based applications is a very complex process. There are a lot of factors that have an influence on the costs of a project. Therefore estimation tools are necessary to produce reliable estimates. Those tools still require some subjective inputs and they also make sure, that no important factor is omitted in the estimating process. Because every project is unique and there is a best method to estimate software costs, is to use several estimation methods. With various regression techniques as tools for predictions we try to estimate the effort required in terms of lines code required per page.

# Chapter 2

## Regression Analysis : An Overview

2.1 Introduction

2.2 History of regression

2.3 Definitions and notation used in regression

2.4 Types of regression

2.4.1 Linear regression

2.4.2 Nonlinear regression models

2.4.3 Non-continuous variables

2.4.4 Other models

2.4.5 Nonparametric regression

## Regression analysis

In statistics, **regression analysis** examines the dependence of a random variable, called a dependent variable (response variable, regressor), on other random or deterministic variables, called independent variables (predictors). The mathematical model of their relationship is the *regression equation*. Well-known types of regression equations are linear regression, the logistic regression for discrete responses (both generalize in the generalized linear model), and nonlinear regression.

Besides the dependent and independent variables, the regression equations usually contain one or more unknown *regression parameters* (constants), which are estimated from given data.

Applications of regression include curve fitting, forecasting of time series, modeling of causal relationships, and testing scientific hypotheses about relationships between variables.

* In real-world applications, data could come from any combination of public or private sources.

## *2.1 Introduction*

Regression analysis estimates the strength of a modeled relationship between one or more response variables (also called dependent variables, explained variables, predicted variables, or regressands) (usually named *Y*), and the predictors (also called independent variables, explanatory variables, control variables, or regressors, usually named $X_1, \ldots, X_p$). These strengths of the relationships given that the model is correct are parameters of the model, which are estimated from a sample. Other parameters which are sometimes specified include error variances and covariances of the variables. The theoretical population parameters are commonly designated by Greek letters (e.g. β), their estimated values by a "hatted" Greek letter (e.g. $\hat{\beta}$), and the sample coefficients by a Latin letter (e.g. *b*). This stresses the fact that the sample coefficients are not the same as the population parameters, but the distribution of those parameters in the population can be inferred from the estimates and the sample

size. This allows researchers to test for the statistical significance of estimated parameters and to measure goodness of fit of the model.

Still more generally, regression may be viewed as a special case of density estimation. The joint distribution of the response and explanatory variables can be constructed from the conditional distribution of the response variable and the marginal distribution of the explanatory variables. In some problems, it is convenient to work in the other direction: from the joint distribution, the conditional distribution of the response variable can be derived. Regression lines can be extrapolated, where the line is extended to fit the model for values of the explanatory variables outside their original range. However extrapolation may be very inaccurate and can only be used reliably in certain instances.

## *2.2 History of regression*

The term "regression" was used in the nineteenth century to describe a biological phenomenon, namely that the progeny of exceptional individuals tend on average to be less exceptional than their parents, and more like their more distant ancestors. Francis Galton studied this phenomenon and applied the slightly misleading term "regression towards mediocrity" to it. For Galton, regression had only this biological meaning, but his work[1] was later extended by Udny Yule and Karl Pearson to a more general statistical context.[2]

## *2.3 Definitions and notation used in regression*

The measured variable, *y*, is conventionally called the "response variable". Other terms include "endogenous variable," "output variable," "criterion variable," and "dependent variable." The controlled or manipulated variables, $\vec{x}$, are called the explanatory variables. Other terms include "exogenous variables," "input variables," "predictor variables" and "independent variables."

## 2.4 Types of regression

Several types of regression analysis can be distinguished; all of these can be seen as special cases of the Generalized Linear Model.

### 2.4.1 Linear regression

Linear regression is a method for determining the parameters of a linear system, that is a system that can be expressed as follows:

$$\sum_{i=1}^{p} \beta_i f_i(\vec{x})$$

where $\beta_i$ is called the parameter and *f* is only a function of $\vec{x}$. This can be rewritten in matrix form as

$$y = \vec{X}\vec{\beta},$$

where $\vec{X}$ is a row vector that contains each of the function, that is, $\vec{X} = \langle f_1(\vec{x}), f_2(\vec{x}), \ldots, f_p(\vec{x}) \rangle$ and $\vec{\beta}$ is a column vector containing the parameters, that is, $\vec{\beta} = \langle \beta_1, \beta_2, \ldots, \beta_p \rangle^T$.

The explanatory and response variables may be scalars or vectors. In the case, where both the explanatory and response variables are scalars, then the resulting regression is called simple linear regression. When there are more than one explanatory variable, then the resulting regression is called multiple linear regression. It should be noted that the general formulae are the same for both cases.

Two common techniques for solving linear regression models is using least squares analysis or robust regression.We use least square regression for all cases.

### 2.4.2 Nonlinear regression models

A number of nonlinear regression techniques may be used to obtain a more accurate regression. It should be noted that an often-used alternative is a transformation of the variables such that the relationship of the transformed variables is again linear.

### 2.4.3 Non-continuous variables

If the variable is not continuous, specific techniques are available. For binary (zero or one) variables, there are the probit and logit model. The multivariate probit model makes it possible to estimate jointly the relationship between several binary dependent variables and some independent variables. For categorical variables with more than two values there is the multinomial logit. For ordinal variables with more than two values, there are the ordered logit and ordered probit models. An alternative to such procedures is linear regression based on polychoric or polyserial correlations between the categorical variables. Such procedures differ in the assumptions made about the distribution of the variables in the population. If the variable is positive with low values and represents the repetition of the occurence of an event, count models like the Poisson regression or the negative binomial model may be appropriate.

### 2.4.4 Other models

Although these three types are the most common, there also exist supervised learning and unit-weighted regression.

### 2.4.5 Nonparametric regression

The models described above are called **parametric** because the researcher must specify the nature of the relationships between the variables in advance. Several non-parametric techniques may be also used to estimate the impact of an explaining variable on a dependent variable. Nonparametric regressions, like kernel regression, require a high number of observations and are computationally intensive.

# Chapter 3

# Methodology

## Methodology

We first downloaded the top ten news sites from the world wide web.We only consider the pages in its own domain for our study.

TOP TEN NEWS SITES

1. www.cnn.com
2. www.bbc.com
3. www.abcnews.com
4. www.boston.com
5. www.xinhua.com
6. www.telegraph.com
7. www.sfgate.com
8. www.reuters.com
9. www.nytimes.com
10. www.iht.com

We used a VBScript to list all downloaded web pages and their directories of each website in a text file.

Then a script in Matlab was run to count the number of pages, Average Number of Lines per Page, Average Number of Scripts per Page, Link Density, and Media Density.

We assume the Number of Lines of Code to be directly proportional to the effort required and is thus taken as the observation(y) and the other four parameters as the regressors(x). This regressor data was represented as a matrix x is a 10 * 4 matrix and the observed data is taken as a 10 * 1 matrix (Y)

X =

| | | | |
|---|---|---|---|
| 21.0000 | 94.4100 | 29.5800 | 45.8710 |
| 10.0000 | 83.1100 | 3.8000 | 34.2000 |
| 16.0000 | 99.1800 | 41.8700 | 56.2500 |
| 4.0000 | 385.2500 | 6.5000 | 63.0000 |
| 1.0000 | 155.0000 | 12.0000 | 59.0000 |
| 5.0000 | 121.2000 | 15.8000 | 80.2000 |
| 1.0000 | 202.0000 | 54.0000 | 30.0000 |

```
   1.0000  387.0000   19.0000   26.0000
   2.0000  204.0000   23.0000   21.5000
  30.0000  108.7930   25.6970   57.6990
```

y =

```
    518.9
   686.9
   613.9
   1375.3
   1643.0
   772.8
   1462.0
   1585.0
   1401.5
   642.9
```

# Chapter 4

## Analysis and Results

4.1 Multiple Linear Regression Analysis
4.2 Stepwise Multiple Linear Regression Analysis
4.3Polynomial Regression Analysis

# Analysis and Results

## 4.1 Multiple Linear Regression Analysis

Mathematical Foundations of Multiple Linear Regression

The linear model takes its common form

$y = X\beta + \varepsilon$

where:

y is an n-by-1 vector of observations.

X is an n-by-p matrix of regressors.

B is a p-by-1 vector of parameters.

ε is an n-by-1 vector of random disturbances.

The solution to the problem is a vector, b, which estimates the unknown vector of parameters, β . The least squares solution is

.b= $\beta=(X^TX)^{-1}X^Ty$

This equation is useful for developing later statistical formulas, but has poor numeric properties. *regress* uses QR decomposition of X followed by the backslash operator to compute b. The QR decomposition is not necessary for computing b, but the matrix R is useful for computing confidence intervals. You can plug b back into the model formula to get the predicted y values at the data points.

$Y = Xb = Hy$

$X= (X^TX)^{-1}X^T$

The residuals are the difference between the observed and predicted y values.

$r = y - \boldsymbol{y} = (I-H)\, y$

The residuals are useful for detecting failures in the model assumptions, since they correspond to the errors, ε, in the model equation.

The formal definition of confidence interval is a range of values for a variable of interest constructed so that this range has a specified probability of including the true value of the variable the specified probability is called confidence level and the endpoints are called confidence limits.

[b,bint,r,rint,stats] = regress(y,X)

```
stats =   1   4    0    82024

rint =
 -799.6814  568.9189
 -617.6864  234.0638
 -797.5063  475.6233
 -549.1791  402.6039
  132.9485  777.8940
 -675.1909  243.5107
 -415.0570  507.8610
 -619.1346  506.1871
 -605.0391  713.1949
 -129.1232  644.7917

r =
 -115.3812
 -191.8113
 -160.9415
  -73.2876
  455.4213
 -215.8401
   46.4020
  -56.4737
   54.0779
  257.8343
b =
  1.0e+003 *

    1082.2
   -26.3
     1.6
     2.1
    -2.3
```

rcoplot(r,rint)

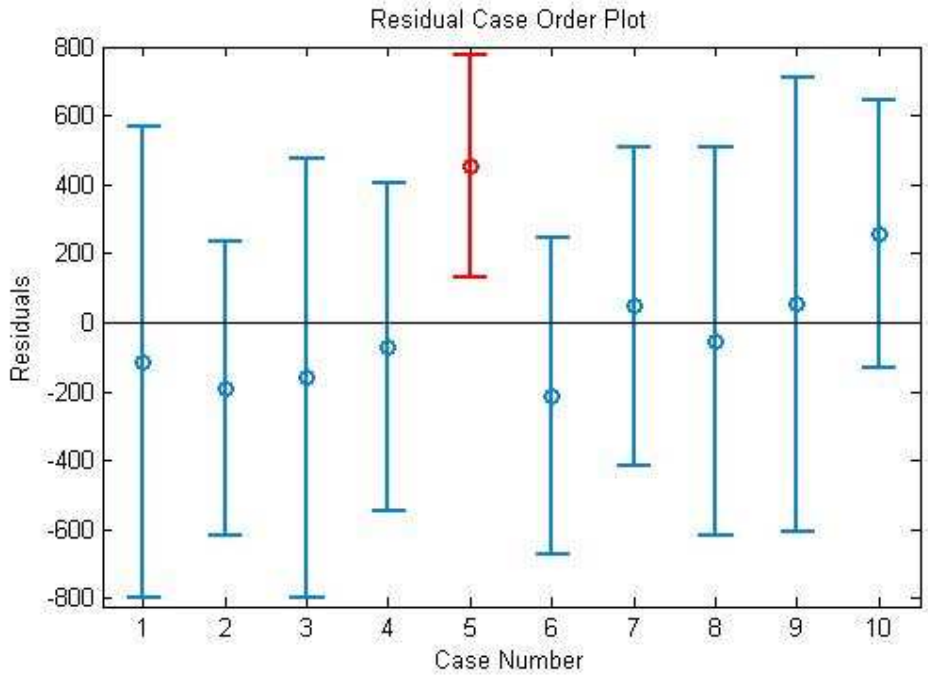


Figure 2

Figure 2 shows a plot of the residuals with error bars showing 95% confidence intervals on the residuals. The fifth error bar doesn't pass through the zero line, indicating that it is an outlier in the data.

## 4.2 Stepwise Multiple Regression Analysis

Stepwise Regression Stepwise regression is a technique for choosing the variables, i.e., terms, to include in a multiple regression model. Forward stepwise regression starts with no model terms. At each step it adds the most statistically significant term (the one with the highest F statistic or lowest p-value) until there are none left. Backward stepwise regression starts with all the terms in the model and removes the least significant terms until all the remaining terms are statistically significant. It is also possible to start with a subset of all the terms and then add significant terms or remove insignificant terms. An important assumption behind the method is that some input variables in a multiple regression do not have an important explanatory effect on the response. If this assumption is true, then it is a convenient simplification to keep only the statistically significant terms in the model. One common problem in multiple

regression analysis is multicollinearity of the input variables. The input variables may be as correlated with each other as they are with the response. If this is the case, the presence of one input variable in the model may mask the effect of another input. Stepwise regression might include different variables depending on the choice of starting model and inclusion strategy.

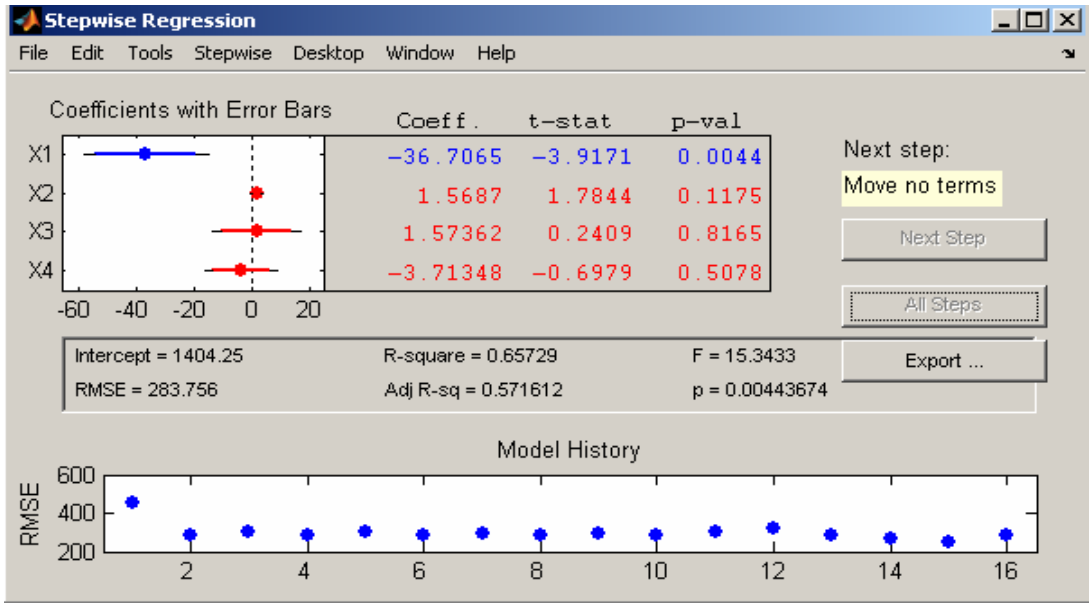


Figure 3. Tool used for Stepwise Regression Analysis

From Stepwise regression analysis it was found that Average Number of Lines per Page, Average Number of Scripts per Page, Link Density, and Media Density were statistically significant where as the Number of Pages per site was not statistically significant in order to determine the number of lines of code required for a web page.

## 4.3 Polynomial Regression Analysis

Polynomial Regression Based on the plot, it is possible that the data can be modeled by a polynomial function

$Y=a\ t + b\ t^1 + ct^2 + \ldots$

The unknown coefficients a0, a1, and a2 can be computed by doing a least squares fit, which minimizes the sum of the squares of the deviations of the data from the model.

If any fit does not perfectly approximate the data. We could either increase the order of the polynomial fit, or explore some other functional form to get a better approximation.

Here we have taken each of the predictors separately and tried to get an optimal polynomial fit at the lowest possible order.

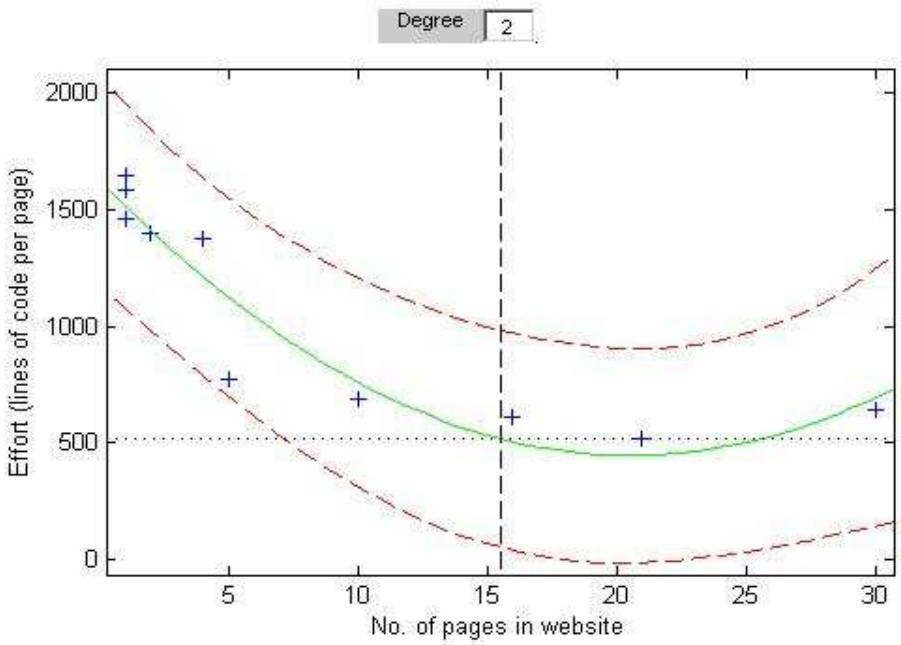


Figure 4.

Figure 4 shows how the lines of code vary as a polynomial of degree 2 with the no of pages in the website.

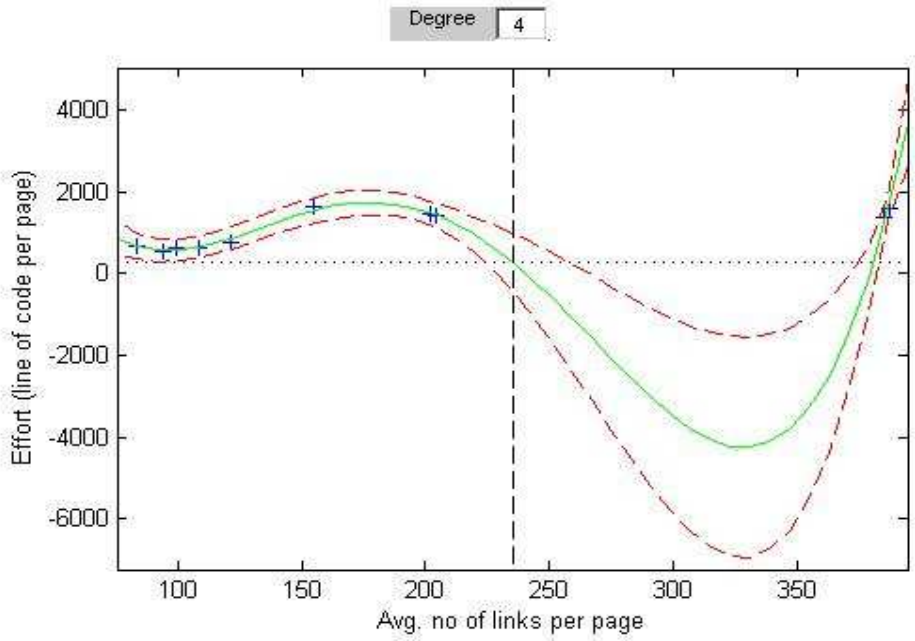


Figure 5

Figure 5 shows how the lines of code vary as a polynomial of degree 4 with the no of links per page.

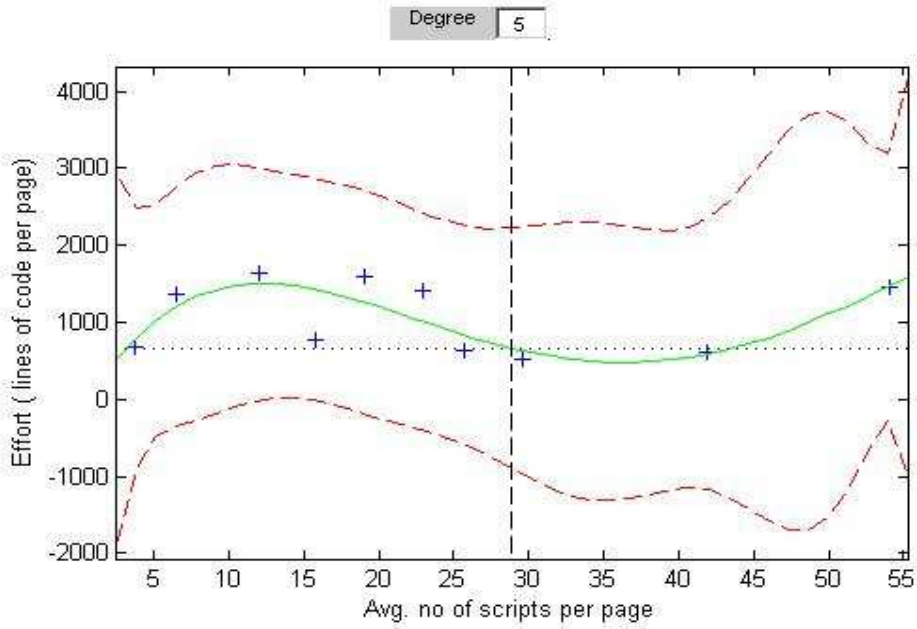


Figure 6

Figure 6 shows how the lines of code vary as a polynomial of degree 5 with the no of scripts per page.

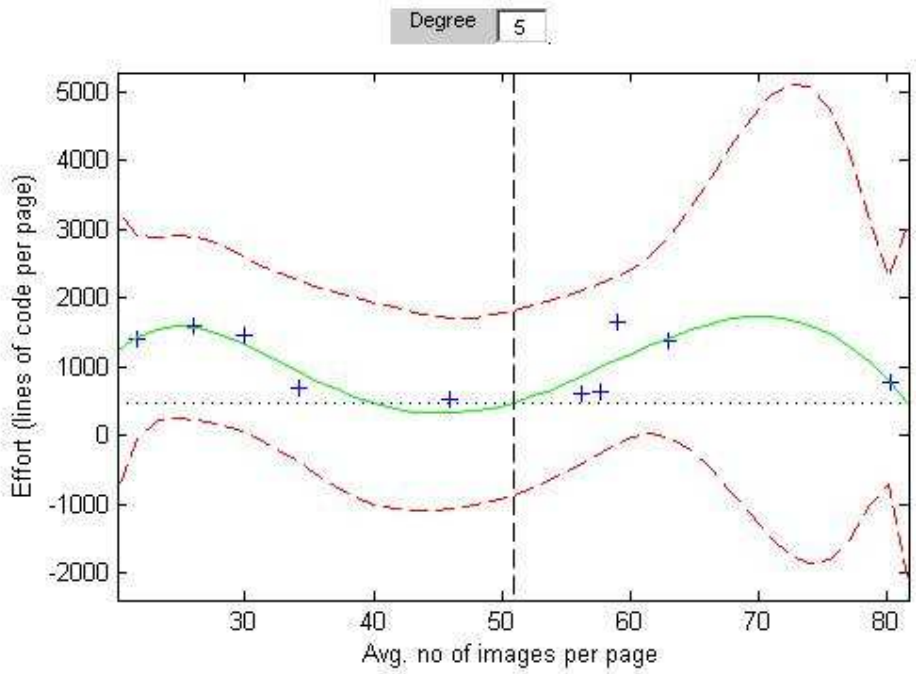


Figure 7

Figure 7 shows how the lines of code vary as a polynomial of degree 5 with the no of images per page.

# Chapter 5

## Conclusion And Future Work

5.1 Conclusion
5.2 Future Work

## 5.1 Conclusion

We were able to predict how the development of a website depends on various parameters and also find out till what extent it depended on them i.e. which parameter affects the observations more and which can be neglected.

In this project work we devised a method to estimate the effort required or the number of lines of code required for a webpage of a news site. These results can be used to devise a standard for the coding of newer news web sites. If they are incorporated in to a web authoring software it would help the author to stick to the guidelines and the standards automatically .

## 5.2 Future Work

Although in this project work we devised a method to estimate the effort required in the number of lines of code required for a webpage of a news site. These results may not be accurate for all cases as it is the actual number of man hours that matters .Total effort should be calculated as:

$Total\text{-}effort = \Sigma\, PAE + \Sigma\, MAE + \Sigma\, PRE$

where PAE is the page authoring effort, MAE the media authoring effort and PRE the program authoring effort. This data can be made available only by the internal sources.

We considered only four predictors to construct this model but there are other measures which should also be included like:

1. Reused Media Count (RMC) - Number of reused/modified media files.
2. Reused Program Count (RPC) - Number of reused/modified programs.
3. Total Page Complexity (TPC) - Average number of different types of media per page.
4. Total Effort (TE) - Effort in person hours to design and author the application

# REFERENCES

[1]Chris Triggs and Ian Watson , A Comparison of Development Effort Estimation Techniques for Web Hypermedia Applications..

[2]Emilia Mendes and , Barbara Kitchenham ,Within-company Effort Estimation Models for Web Applications **.**

[3] M.J. Shepperd, C. Schofield, and B. Kitchenham, "Effort Estimation Using Analogy." Proc. ICSE-18, IEEE Computer Society Press, Berlin, 1996.

[4] R. Gray, S. G. MacDonell, and M. J. Shepperd, "FactorsSystematically associated with errors in subjective estimates of software development effort: the stability of expert judgement", IEEE 6th International Metrics Symposium, Boca Raton, November 5-6, 1999.

[5] Kok, P., B. A. Kitchenham, J. Kirakowski, The MERMAID Approach to software cost estimation, ESPRIT Annual Conference, Brussels, p: 296-314, 1990.

[6] T. DeMarco, Controlling Software Projects: Management,Measurement and Estimation, Yourdon: New York, 1982.

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