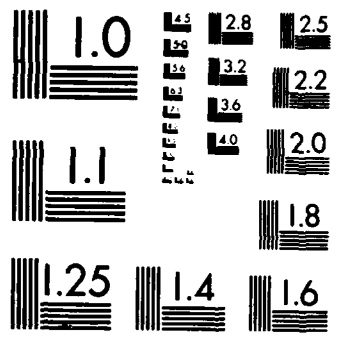
AD-A174 335   DEVELOPING SOFTWARE SIZE ESTIMATING RELATIONSHIPS BASED   1/2
ON FUNCTIONAL DES (U) AIR FORCE INST OF TECH
WRIGHT-PATTERSON AFB OH SCHOOL OF SYST   M J WHETSTONE
UNCLASSIFIED   SEP 86 AFIT/GSM/LSY/86S-24   F/G 9/2   NL


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0

2

DEVELOPING SOFTWARE SIZE ESTIMATING RELATIONSHIPS BASED ON FUNCTIONAL DESCRIPTIONS OF THE SOFTWARE

THESIS

Mark J. Whetstone
Captain, USAF

AFIT/GSM/LSY/86S-24

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY
AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT/GSM/LSY/86S-24

DEVELOPING SOFTWARE SIZE ESTIMATING RELATIONSHIPS BASED ON FUNCTIONAL DESCRIPTIONS OF THE SOFTWARE

THESIS

Mark J. Whetstone
Captain, USAF

AFIT/GSM/LSY/86S-24

The contents of the document are technically accurate, and no sensitive items, detrimental ideas, or deleterious information is contained therein. Furthermore, the views expressed in the document are those of the author and do not necessarily reflect the views of the School of Systems and Logistics, the Air University, the United States Air Force, or the Department of Defense.

AFIT/GSM/LSY/86S-24

DEVELOPING SOFTWARE SIZE ESTIMATING RELATIONSHIPS BASED ON FUNCTIONAL DESCRIPTIONS OF THE SOFTWARE

THESIS

Presented to the Faculty of the School of Systems and Logistics

of the Air Force Institute of Technology

Air University

In Partial Fulfillment of the

Requirements for the Degree of

Master of Science in Systems Management

Mark J. Whetstone, M.B.A.
Captain, USAF

September 1986

Acknowledgements

I would first like to thank my advisor, Mr. Daniel V. Ferens, and my reader, Major William F. Bowlin, for their help, advice, and especially their patience. Without their dedicated efforts and many long hours reading my rough drafts, I would have never been able to accomplish this thesis.

I am also indebted to my typist, Mrs. Joyce Burnette. Her excellent typing skills and ability to read my writing made this thesis possible and has given it a very professional appearance.

Finally, the two people who I will never be able to thank enough are my wife Susan and daughter Crystal. Crystal, born just three weeks before I started AFIT, is truly an "AFIT baby." My wife's love and understanding of why I spent so many extra hours studying or on the computer instead of with her and our daughter made these last 15 months easier to endure. From my daughter's loving smiles of recognition of me as a newborn to her ability to say "da da" now, gave me the drive and encouragement to finish this thesis and degree. This thesis is dedicated to them.

Mark J. Whetstone

Table of Contents

List of Figures

List of Tables

AFIT/GSM/LSY/86S-23

Abstract

This thesis researched the ability to develop regression models to predict the number of source lines of code (LOC) based on functional descriptions of the software. LOC, a major cost driver in currently available software cost estimating models, has been consistently underestimated, thus lowering not only the software cost estimate but also the total cost estimate of the weapon system. Six software sizing data bases containing various functional variables were used. The variables included complexity, reliability, experience level of programmers, etc. For each data base, regression analysis was performed to derive the optimal model to predict LOC. Of the five data bases containing complexity, it was statistically significant in three. The best developed model was for Armament Division's airborne computer programs. The correlation coefficient R^2 was .6583 for the two variables in the model. These were: (1) the system for which the program was developed and (2) the reliability needed in the program. The initial research has been accomplished, but more data and further research is needed.

DEVELOPING SOFTWARE SIZE ESTIMATING RELATIONSHIPS BASED ON FUNCTIONAL DESCRIPTIONS OF THE SOFTWARE

I. Introduction

This chapter provides an overview of the thesis. First, the general issue of the thesis is described including why the issue discussed is a problem and why the problem is important. Second, the scope of this thesis is described. Next, definitions of terms central to the understanding of the thesis are explained. Finally, the research questions which this thesis answers are listed.

General Issue

Air Force and other Department of Defense (DoD) weapon systems are rapidly increasing in complexity. This complexity, in turn, is partially a function of the sophisticated computer programs needed to run the state-of-the-art internal subsystems of these new weapon systems.

New weapon systems are very costly. Almost daily, reports of cost overruns on a new DoD weapon system can be read in the evening paper. These reports usually state that DoD's cost estimates were initially low, which is now causing the current cost overruns. The

DoD has recognized this problem. The 24 May 1984 Comptroller General's Report to the Congress states:

> DoD's cost estimates for weapon system programs are of major concern when the Congress is deciding to allocate billions of dollars to defense programs. The accuracy, completeness, and timeliness of DoD's cost estimates need to be improved to give Congress more reliable data for its decision process [13:i].

Since our new weapon systems rely on complex computer software, these software costs make up a major portion of the overall cost estimate for these weapon systems. The major cost driver for most software cost estimating models currently available is the size or the number of lines of code (LOC) for the software. However, the size of the software has been consistently underestimated, thus lowering the software cost estimate and, consequently, the overall cost estimate for the weapon system. In a research report on software sizing methods conducted for the DoD by the ARINC Research Corporation the authors state,

> For several years the DoD has experienced problems in estimating and controlling software costs, including those for all phases of software development and life-cycle support. Many examples exist where cost and schedule overruns in software acquisitions have led to unexpected cost growth for the overall system in which the software is embedded [1:v].

An even more recent article on the problem of software sizing states,

> Software size estimates almost always grow over the life-cycle. The amount of underestimation varies depending on many factors; but on the average, it is in the range of 70 to 100 percent from contract award to project completion. For this reason, it is imperative that greater

> efforts are applied towards obtaining more accurate size estimates earlier in the software life-cycle [14:17].

Still another report prepared for the DoD states,

> The increasing contribution of software development and maintenance costs to the overall life-cycle cost of DoD weapon systems has been well documented in recent years. In particular, software life-cycle costs are predicted to be in excess of 50 percent of the total system costs by the end of the decade [12:1-1].

The significance of the software size estimating problem for the Air Force is stated by the HQ Air Force Systems Command's (AFSC's) Cost Method Improvement Group (CMIG) report:

> The size of non-space-constrained software is regularly underestimated in the early development phases. Yet most software cost models are based on size as an input variable. Low cost estimates result. Since software is becoming a more and more significant part of Full Scale Development (FSD) cost (and schedule), unbiased software size estimating techniques must be developed [6:65].

Problem Statement

Because of the recent Gramm-Rudman budget cuts currently affecting acquisition programs, and because of possible future budget cuts, it is imperative that our future cost estimates be as accurate as possible. As discussed above, an accurate software size estimate is a key factor in the overall cost estimate of a weapon system. According to the ARINC Research Corporation, who investigated techniques on software size estimations, there are four major software sizing models: measurement technique, quantitative functional relationships, qualitative functional relationships, and PERT sizing. Their subjective analysis concluded that the best method is measurement technique, which

> . . . assumes that software size can be reliably estimated through rapid software prototyping in which critical functions of a complex software development are initially developed to demonstrate feasible performance [1:2-5].

Their analysis also states that measurement techniques are the most difficult to develop and are very limited in their generalizability, but do give moderate to high accuracy potential (1:2-6). Quantitative functional relationships are the second best method because they can provide an accuracy potential up to a moderate level, whereas PERT sizing and qualitative relationships can only provide a low level accuracy (1:2-5, 2-6). As pointed out above, quantitative techniques are what the CMIG report says the Air Force should use to estimate software size. Unfortunately, the ARINC report also states that at the present time no reliable quantitative functional relationship for software sizing is available (1:2-5). Therefore, the specific objective of this research study is to develop an equation, or a set of equations, based on multiple regression analysis that relates software size (the dependent variable) to functional characteristics of the software (the independent variables).

Scope

This thesis attempts to develop a regression model for estimating software size (the number of source lines of code) based on functional descriptions of the software for each of the four data bases used. The software size estimate generated from each of the successful models can then be used in a cost estimating relationship that uses software size as an input. The estimating equations developed are based only on multiple regression analysis and use several of the many possible significant functional descriptions of software that could be related to size. Because of the limited number of software sizing data bases available and the content of some of the data bases that were obtained, only four data bases are used.

Definitions

The following definitions are critical to an understanding of this research study:

Computer Software Configuration Item (CSCI). "Hardware or software, or an aggregation of both, which is designed by the contracting agency for configuration management" (4:8).

Size Estimating Relationship (SER).

> A size estimating relationship assumes that software size can be reliably estimated through the development and use of empirical equations that relate size to certain functional characteristics of the software. It is necessary to develop or obtain those equations that have characteristics similar to the software function

under evaluation. The equations are developed by statistical analysis of empirical data from actual software programs [1:2-5].

Multiple Regression. "Multiple regression analysis is a statistical tool that utilizes the relation between two or more quantitative variables so that one variable can be predicted from the others" (7:23).

Computer Program. "A series of instructions or statements in a form acceptable to an electronic computer designed to cause the computer to execute an operation or a series of operations" (12:B-39).

Computer Software. "A combination of associated programs and computer program data definitions required to enable the computer hardware to perform computational or control functions" (12:B-39).

Size Driver. Analagous to a cost driver as an independent variable except that it is a description of the computer software that may be statistically related to size.

Software Sizing Data Base. "A collection of data points consisting of software size versus software functions collected at the subsystem and component level which are suitably correlated" (9:3).

Research Questions

The primary research questions addressed in this study are:

1. What are several of the significant "size drivers" for each of the software sizing data bases used?

2. Given these size drivers, can a multiple regression model be developed to predict software size for each data base?

The research subquestion is: How "generalizable" are the mathematical equations developed? In other words, can one or more of them be applied to other software sizing data bases with the original size drivers still being statistically significant to predict software size using this new data base?

Research Development

The development of the research proposed in this paper will follow the general chapter descriptions as outlined below.

As described in this section of the proposal, chapter one is the introduction to the research problem. It covers the general issue of the topic of software sizing, the problem statement, the scope of the research, definitions to help explain important terms that will be essential in understanding the research, and the research question to be investigated.

Chapter two contains the literature review on software sizing. The literature reveals the various viewpoints held in the area.

Chapter three of the thesis explains the specific methodology to be used in the research.

Chapter four of the thesis contains the results of the data analysis and the statistical testing. The results of the regression is analyzed with respect to the methodology described in chapter three.

Chapter five of the thesis discusses the conclusions drawn from the research. It also describes any ideas and needs for further research that may be generated in this study.

II. Background

Chapter Overview

This chapter contains a review of the literature concerning the software sizing concept as it applies to the DoD cost analyst. First, to see where the software sizing estimate fits into the overall software cost estimate, a short explanation of current software trends and the software cost estimating process will be covered. Second, the software sizing concept will be explained in more detail. Third, an explanation of a software sizing data base and how it relates to software size estimation will be discussed. Lastly, a few of the commercial software size estimating models available will be described. The literature review concludes with the author's comments on the literature.

Software Trends

"In 1981 the annual cost of software in the United States was forty billion dollars or two percent of Gross National Product (GNP)" (10:2). If this current growth trend continues, software costs will become a much greater percentage of GNP in the future. Boehm had estimated that by 1985 almost 40% of the American labor force would be using computers in their jobs (2:19). Within the total growth area of computer costs, an ever-increasing percent of the total costs will be made up of software costs (2:18). For DoD the rapidly rising costs of software has hit the critical level.

> For several years, the DoD has experienced problems in estimating and controlling software costs, including those for all phases of software development and life-cycle support. Many examples exist where cost and schedule overruns in software acquisitions have led to unexpected cost growth for the overall system in which the software is embedded [1:v].

Another report states that for the DoD, "Software life-cycle costs are predicted to be in excess of 80% of total computer hardware/software system life-cycle and in excess of 50% of the total system costs by the end of the decade" (12:1-1).

To understand how important software size estimation is, it is necessary to understand the overall software cost estimating process.

Software Cost Estimating

In his book, Software Engineering Economics, Barry W. Boehm describes the importance of software cost estimating.

> The reason for this strong emphasis on software cost estimation is that it provides the vital link between the general concepts and techniques of economic analysis and the particular world of software engineering. There is no good way to perform a software cost/benefit analysis without some reasonably accurate method of estimating software costs, and their sensitivity to various product, project, and environmental factors. Software cost estimation techniques are also important because they provide an essential part of the foundation for good software management [2:30].

Software cost estimating, therefore, is the key link between developing software and determining whether it is economical. However, there are major problems in software cost estimating that must be kept in mind by the cost analyst. In an Air Force Institute of Technology (AFIT) report on software costing, the authors; Steig, Stewe, and Ward; summarize six key problems in software costing. (See Barry Boehm's book for further details; bibliography reference number two.)

> First, source instructions are not uniform from project to project, nor do they capture the essence of the desired product. Second, software engineering requires creativity and the cooperation of human beings whose individual and group behavior is hard to predict. Often the user does not know what is available and does not know the organization's idiosyncrasies that he must know if he is to design a product compatible with the organization. Third, the software engineering process has a much smaller base of relevent quantitative historical experience from which to draw than do other developmental/engineering efforts. Additionally, it is extremely difficult and costly to add to this base by performing controlled experiments. Fourth, the outputs of software engineering efforts are so diverse that it is nearly impossible to rely on comparative techniques to extrapolate meaningful cost factors. Fifth, software engineers are often over-optimistic and tend to forget previous experiences which adversely affected other similar software development efforts. Lastly, in the middle of a software development effort it is often extremely difficult to determine how the total program completion rate compares with the total program expenditure rate [10:6-7].

Because of these factors, the cost analyst will constantly be faced with a challenge to accurately estimate the costs of software.

The software development process involves a highly integrated set of requirements and resources to produce a software program. There are many cost drivers that affect the outcome of this development process. According to a report by The Analytic Sciences Corporation, there are ten major categories of cost drivers affecting the costs of the software development process:

1. Functional requirements
2. Development methods
3. Programming language
4. Development environment
5. Personnel quality
6. Hardware constraints
7. Documentation
8. Operational environment
9. Schedule requirements
10. Code size [12:2-1].

The authors of the previously cited AFIT report have also compiled a list of cost drivers from different sources and have separated them into different attributes with the major determinants of cost under each attribute:

Size:
1. The number of 'preplanned' lines of code.
2. The complexity of the system.

Program:
1. The amount of projected software maintenance.
2. The performance and reliability specifications.
3. Whether or not data dictionaries are used.

Computer:
1. Whether or not the software has to make up for hardware deficiencies.
2. Whether the software is designed before or after the hardware.
3. The amount of memory space available to the engineer.

Personnel:
1. Whether or not the same expert personnel remain on the job from start to finish.
2. Whether or not personnel on the project have

experience on similar projects.

3. Often adding personnel to help a late job catch up only makes it later.

Project:

1. The amount of up front detailed design.
2. How much requirements change.
3. How familiar the user personnel are with the capabilities of the system being developed [10:13-14].

Knowing all these possible problems in software cost estimating and the many possible cost drivers, the cost analyst must then develop a cost model to estimate the cost of the software.

Software Size--The Key Variable

Even after having developed a software cost model, there are still problems the analyst must be aware of. Boehm states it very well.

> Having a good software cost model available does not guarantee good software cost estimates. As with other computer-based models, a software cost estimation model is a 'garbage in-garbage out' device: if you put poor sizing and attribute-rating data in on one side, you will receive poor cost estimates out the other side [2:308].

In recent years, the DoD has found out just how true this statement is. As summarized in a report from the Resource Cost Analysis Office of The Aerospace Corporation, the need for Air Force and other DoD program managers to have complete and accurate information about future software costs for the new state-of-the-art weapon systems has become increasingly critical. This is particularly important in software cost estimating. Since both cost and schedule estimates are usually based on the size of the new weapon system, software size is a key parameter (11:i). As stated in the introduction of this proposal, HQ AFSC's CMIG report emphasized that the size of the software program for the new weapon systems is regularly underestimated, and because most software models rely on size as an input variable into the model, low cost estimates result (6:65).

Boehm also emphasizes the importance of the software size variable. "The software undersizing problem is our most critical road block to accurate software cost estimation" (2:320). He also lists the three main causes of underestimating software size.

> 1. People are basically optimistic and desire to please. Everybody would like the software to be small and easy. High estimates lead to confrontation situations, which people generally prefer to avoid.
> 2. People tend to have incomplete recall of previous experience. In terms of the distribution of source code by function, for example, people tend to have a strong recollection of the primary application software functions to be developed--the 2 to 3 percent of the product devoted to model calculations--and a much weaker recollection of the large amount of user-interface and housekeeping software that must also be developed.
> 3. People are generally not familiar with the entire software job. This factor tends to interact with the incomplete-recall factor to produce underestimates of the more obscure software products to be developed. A major example is a strong tendency to underestimate the size of support software [2:320-321].

In still another report, The Analytic Sciences Corporation stated that even though alternatives to the use of lines of code have been proposed over the years, it is fairly certain that lines of code will remain the standard for software cost estimation models. Many of these alternatives are highly correlated to the lines of code parameter, such as, number of modules (12:2-4). The report concludes, "In general, for software cost estimation, lines of code is the most promising variable when used in combination with qualitative information" (12:2-4).

From the literature surveyed, software size seems to be considered the key parameter in the software cost estimation process.

Software Data Bases

In order to use most types of software size estimating models, the analyst must have historical data to input into the model to come up with an estimate of the number of lines of code for the new system. The data for a particular project, in this case software sizing, is collected, analyzed, sorted, and placed into a data base. This "data" is then inputted into a storage/retrieval system, usually of a computer system. This is the "base" or supporting foundation of all information requests.

According to a software data base report conducted by The Analytic Sciences Corporation, at the present time software data bases have been developed into two categories: those that contain summary data at the system level and those that contain detail data at the lowest level to which software can be logically divided (12:2-2). Unfortunately, the amount of detail is determined not by the requirements of the user, but by the availability of the data (12:2-2). The most detailed data bases exist within the development organization which have a direct influence on the type of data collected for the day-to-day management of a development effort (12:2-2). Therefore, government software development organizations, such as the NASA Software Engineering Laboratory, and defense contractors historically have the best data available (12:2-2). Unfortunately, data availability at government program officies is limited by the existing data items used for reporting software technical and resource utilization data (12:2-2).

Generally, software data bases are divided into six distinct categories:

1. System description and characteristics
2. Development schedule data
3. Hardware characteristics and constraints
4. Development resources and constraints
5. Software size and characteristics
6. Resource expenditure data

> The data within each of these categories consist of the elements required to classify the system, to define the development environment, and to derive the software development cost drivers and input parameters for software cost and sizing models [12:2-6].

For this research, the most important of these six categories is the software size and characteristic category. The type of software size data included in the data base is driven by two requirements.

> First, is the need for size data at the computer program configuration item (CPCI) level with allocations to various functional characteristics, processing modes, and languages to support the specific requirements of several cost and sizing models. Second, the need for size decomposition to the lowest level available with functional categorization and language identification to support sizing by analogy requirements [12:2-10].

Because of the recent emphasis on software sizing models to produce a reliable estimate on the number of lines of code, software sizing data bases have been developed. However, because of the cost, time, and effort needed to develop a software sizing data base, there are only a limited number of them within DoD. One of the newest (5 June 1985) sizing data bases has been developed by the Space Systems Cost Analysis Group, Software Subgroup, of The Aerospace Corporation. They used two surveys to collect data needed to develop the data base. In their report attached to the sizing data base they state:

> The purpose of both surveys was to build a software sizing data base for use in predicting software module size (lines of code) for new software development. The parameter 'lines of code' is critical to the accurate estimating of software development effort using available cost estimating models. A specific goal of the survey effort and subsequent analysis was to statistically correlate software module size with software module function. The current data base has been successfully used by Space Division and by the Aerospace Corporation as a guide ('look up table') for estimating software size as related to function [9:2].

A second extensive software sizing data base has been developed by the Armament Division (AD). It consists of programs written for missile, range, and munition systems. Each of the programs are described by various functional characteristics: the number of source lines of code, the number of development months, the programming language, the degree of system specification, the reliability requirement needed in the program, and the relative complexity of the function.

A third comprehensive software sizing data base has been constructed by the Electronic Systems Division (ESD). This data base consists of varying electronic software programs. Each program is also described by various functional characteristics similar to the AD data base.

A fourth software sizing data base has been developed by the Simulator System Program Office at the Aeronautical Systems Division (ASD). This data base also describes each program by different functional characteristics. However, in this data base, the number of lines of code are in machine language.

Finally, the Ballistic Missile Office (BMO) has developed a software sizing data base. Again, each program is described by several different functional characteristics. In this data base, the number of lines of code are in source lines of code.

Software Sizing Models

Given that a software sizing data base is available and the appropriate inputs for the model being used are known, then the analyst can develop an estimate of the number of lines of code. Because the purpose of this research is to develop a new elementary software sizing model, it is appropriate now to describe some of the software sizing models currently in use.

According to a technical report on software sizing and cost estimation conducted by the ARINC Research Corporation, there are four general software sizing methods used by the DoD and software development companies. The four general methods are (1) Program Evaluation and Review Technique (PERT) sizing; (2) qualitative functional relationships; (3) quantitative functional relationships; and (4) measurement. Table I lists these methods together with the corresponding approaches and form of the estimating relationships (1:2-2,2-3).

The four general methods are briefly discussed below.

PERT sizing allows the analyst to estimate software size on the basis of experience and engineering judgment. PERT sizing makes the assumption that software development experts can provide reliable size estimates for new developments by using the knowledge they and others have gained from similar software development projects. However, the quality of the estimate is dependent on the expert's capabilities in remembering the knowledge gained from other efforts and engineering judgment. Therefore, PERT sizing is a common technique for using experience and judgment in estimating software size. PERT sizing is a formal approach used by analyst to estimate the most likely sizes for any given software function, as well as upper and lower limits. The mathematical equation for PERT sizing is

$$\mathrm{SIZE} = a + 4m + b, \qquad (1)$$

TABLE I

Software Sizing Methods

Method	Approach	Form of Estimating Relationship
PERT Sizing	Delphi Polling	Size = $\frac{a + 4m + b}{6}$ $\sigma = \frac{b - a}{6}$
Qualitative Functional Relationships	Analogy	Size = 5 to 10K lines *for* function sonar track processing + 10 to 15K lines for function data management
Quantitative Functional Relationships	Analytical SERs	Size = $\sum_i$ (number modules$_i$) x (average module size$_i$) x (attributes$_i$)
Measurement	Prototyping	Size = Extrapolation from size of rapid prototyping of critical function(s)

REPRINTED FROM: (1,2-3)

and the associated standard deviation (σ) is approximately,

$$\sigma = b - a / 6 \qquad (2)$$

where:

> a = smallest number of lines of code
> m = most likely number of lines of code
> b = largest number of lines of code

PERT sizing should be used in the early phases of a program when little detail about the software is known (1:2-3,2-4).

The second method for estimating software size, qualitative functional relationship, is based on performing a comparison or analogy to a similar system.

Qualitative functional relationships assume that

> software size for a new weapon system can be estimated by analogy, where identified functional requirements of the new system are compared in a qualitative manner with those of existing systems [1:2-4].

Two approaches for the size estimate based on analogy can be used: top-down or bottom-up. In general, the top-down approach involves system-level estimates based on existing systems with similar applications. The bottom-up approach entails function-level estimates based on similarity of software functions. The estimated size of each function determined by analogy are then summed to produce a total system size estimate (1:2-4).

> Regardless of which approach is used, the accuracy of the software size estimate depends on the quality of data available, the extent of the data analysis, and the validity

> of the analogies. Normally, as systems become more complex, the probability that inaccurate comparisons will be made increases and, accordingly, one's level of confidence in the estimate decreases [1:2-5].

According to the same report, there are presently no reliable quantitative functional relationships for software sizing. Nonetheless, those quantitative methods under research assume that software size can be reliably estimated through the development of empirical equations, such as regression equations, that relate size to different functional characteristics of the software. It is necessary, therefore, to develop equations that have characteristics similar to the software function under evaluation. These equations are developed by statistical analysis of data from actual software programs (1:2-5). These data points compose the software sizing data bases discussed earlier.

The last method is the measurement technique.

> This technique assumes that software size can be reliably estimated through rapid software prototyping in which critical functions of a complex software development are initially developed to demonstrate feasible performance. Although the prototype software is not the final product, size measurements at the function level provide fairly refined data that can be extrapolated into software size estimates for the final product with a high level of confidence [1:2-5].

Table II summarizes the four sizing methods discussed above. The entries in the table are the subjective opinions of the authors of the ARINC Research Corporation report where the table comes from. In order to pick the best method, it is necessary to consider the study objectives, what is currently known about the weapon system being

TABLE II

Summary of Sizing Methods

Factor	Sizing Method			
	PERT Sizing	Qualitative Functional Relationships	Quantitative Functional Relationships	Measurement
Approach	Delphi polling	Analogy	Analytical SERs	Prototyping
Development Basis	Engineering judgment	Similar projects	Statistical analysis of historical data	Collected performance data
Development Difficulty	Very low	Moderate	High	Very high
Application Phase	Requirements	Requirements	Requirements and design	Feasibility studies
Generality	Very good	Good	Limited	Very limited
Accuracy Potential	Least accurate	Low	Low to moderate	Moderate to high

REPRINTED FROM: (1,2-6)

considered, and the number of assumptions that can be made and still be able to produce a useful estimate (1:2-6).

Proceeding from these four general categories of software sizing models to more specific sizing models, the Air Force's Space Division and The Aerospace Corporation have developed a "look up table" from the previously mentioned software sizing data base. This "look up table" or guide is used for estimating software size as related to function (9:2). The report under bibliography reference number nine describes the effort in more detail. This is a first step in developing more sophisticated sizing models within DoD.

Private companies have also researched and developed software sizing models, but on a more sophisticated level. These have been developed so the model can be sold for use by those needing software size estimates. The DoD currently uses some of these commercial sizing models in their military format. Two of the more well known models are discussed next.

1. PRICE Systems Division of RCA, Cherry Hills, New Jersey, has developed one of the most recent and state-of-the-art software sizing models to date. Known as the PRICE SZ for PRICE SIZER,

> . . . the PRICE SZ module is a conversational parameter model designed to estimate software program instruction size for commercial or military applications, using RCA's empirical modeling techniques. The PRICE SZ model uses a state-of-the-art approach, . . . using a mix of software design requirements, technical approaches, growth requirements, functional input/outputs, and historical software size behavior [8:1-1].

The size estimates produced from the SZ model can then be used as an input into software cost models. According to the reference manual, SIZER can be used early in the design planning process when only the functions and application of the software are available (8:1-1).

2. A different approach to the sizing problem has been taken by GJB Associates, Software Engineering and Analytical Services of Redwood City, California. They have developed a sizing model called the Software Sizing Model (SSM). They state,

> It has been established, conclusively, that qualitative sizing information available at the proposal stage is significantly more accurate than the corresponding quantitative data. Therefore, the use of qualitative (relative) input is the fundamental principle of SSM [5:1].

This qualitative data determines the relative sizes of the software modules (5:1). "When the modules are ranked in relative magnitude, the actual sizes of only two modules (the lowest and highest possible module sizes) are needed to extrapolate the remaining module sizes" (5:1). The SSM model also interfaces with any software cost model.

Comments and Conclusions From Literature Review

Because of very likely and significant budget cuts faced by DoD caused by the Gramm-Rudman resolution, it will be even more critical that DoD and especially the American taxpayer get more "bang for the buck" for future weapon systems. Since these new weapon systems are becoming more dependent on computer software to control the myriad of different functions built into these systems, software costs will become an even bigger portion of the total costs of these weapon systems. Of all the software cost drivers, it is software size or the number of lines of code that has been singled out as being the most important. This point has been emphasized throughout this literature review.

However, it has only been in the last couple of years that DoD has started to develop software sizing data bases in order to develop software sizing models. As can be seen from this review there are only a few sizing models available and, according to one report, there are no significant quantitative models based on empirical equations that can be used by a program office cost analyst. Therefore, the main problem is to develop a general, statistically accurate sizing model, using regression equations. Finally, the model should be developed using existing software sizing data bases (although limited) which are based on the functional description of the software.

III. Methodology

Chapter Overview

The methods used to conduct this research will be discussed in this chapter. The initial work consisted of personal interviews, phone interviews, and a review of the literature that helped determine the availability of software sizing data bases and, in general, the current state-of-the-art of software sizing models. Secondly, the size drivers were determined for each sizing data base used. Finally, multiple regression analysis was used to develop analysis of variance (ANOVA) tables and other regression statistics to determine the significant size drivers and the resulting regression equations.

Data Collection

The personal interviews and phone interviews resulted in four major software sizing data bases being acquired for the research. (As will be noted later, two more data bases were formed from these four major data bases.) The four sizing data bases are all different in terms of number of data points and functional description of the software.

The smallest data base was from HQ Air Force Systems Command and consisted of seven data points for ballistic missile software programs. These programs are all used on missile systems developed at the Ballistic Missile Office (BMO). Each program in the data base had been broken into eight different functional descriptions. They were: the number of source lines of code (LOC), the environment (either ground support programs or airborne programs), the computer language used, the number of interfaces between the program and the user or other programs, the number of inputs needed by the program, the number of outputs generated by the program, the experience of the programmers in months, and the number of months it took to develop the program.

The second data base was from the Electronic Systems Division (ESD). This data base consisted of 26 data points of various electronic software programs. Each of the programs were described by eight different functional descriptions. They were: the LOC, the environment (either ground or airborne), development hours in man-hours, the computer language used, the reliability requirement needed in the program (low to high), a rating of low to extra high representing the relative complexity of the function, the experience of the programmers in months, and the quality of the specification or, in other words, the degree of system definition (low to high).

A third software sizing data base was from the Armament Division (AD). This data base contained 25 useable data points. Each program was described by seven functional descriptions. They were: the LOC, the type of system the program supported (missile, range, or munition), the development months, the programming language used, the degree of system specification (low to high), the reliability requirement needed in the program (low to high), and the relative complexity of the function (low to high).

The fourth data base was a combination of the first three major data bases described above and another data base obtained from Space Division (SD). Because the SD data base contained only two logically possible size drivers (complexity of the function and the programming language) that could relate to the number of LOC, it was decided to combine all four data bases in order to separate each program by language and then use this "new" data base to develop a model.

As will be described in the next section, the methodology used on each data base was the same. Regression models estimating the number of LOC were developed from six distinct data bases. These data bases were: ballistic missile support programs, general electronic system programs, armament system programs which were subdivided into ground support programs and airborne support programs, and last, programs subdivided into the two most prevalent programming languages found in the four major data bases, Fortran and Assembly.

As can be inferred from the description of the four major sizing data bases above, the assembly of the sizing data bases themselves by the various organizations determined what possible size drivers would be tested for each data base. Therefore, the first research question of what are some of the possible significant size drivers for each data base was answered.

At this point, it should be noted that the "size" of the software or number of LOC was estimated in source code. All the LOC in each of the data bases used were in source code. This decision was based on the opinion of Wheaton in her article:

> The use of machine language instructions (MLI) for estimating the size of software is not recommended, because it is best to consider lines of code as units of effort which comprise the total software development effort. This is not possible with MLI as they are a function of the language and compiler efficiency, and not directly related to effort. Using MLI does not provide a consistent basis for measuring effort, since the same source program may generate different numbers of object instructions depending on the compiler [13:17].

Finally, the nonquantitative variables were quantified. The variables of complexity, reliability, and quality of specification were rated as shown in Table III. These ratings were already assigned for the above three variables for each data base by the organization which assembled the data bases. It has therefore been assumed that the software personnel in each of these organizations were knowledgeable about their own data and have assigned the correct rating to each variable. Unfortunately, no references were stated as to why each variable was given its particular rating. The harder-to-quantify variable

TABLE III

Quantification of the Nonquantitative Variables

Complexity/Reliability/ Quality of Specification	Languages	Environment	Function (For AD Data Base)
Very Low = 1	Fortran = 1	Ground = 1	Missile = 1
Low = 2	Pascal = 2	Airborn = 2	Range Equipment = 2
Nominal = 3	Assembly = 3	Space = 3	Munition = 3
High = 4	Event Driven Language = 4		
Very High = 5	Jovial = 5		
Extra High = 6	CMS = 6		
	PLM-86 = 7		
	Basic = 8		
	CPL = 9		
	PL1 = 10		

of programming language was quantified by assigning numerical values to each different language used in the four data bases, per the method used by the ARINC Research Corporation (see 1:4-16). The assignment of values are shown in Table III. The environment that the software operated in was quantified as shown in Table III. Finally, for the AD data base, the function of the system the software operated in was quantified as depicted in Table III.

Statistical Testing and Multiple Regression

As stated in chapter one, the second primary research question asks, can a multiple regression model be developed, using the previously defined size drivers, to accurately predict software size for each data base? In order to answer this question, multiple regression was used to identify the significant size drivers and develop a model to predict the LOC for each data base. The regression analysis was done using the Statistical Analysis System (SAS) package. For each data base the following general steps and analysis techniques were used to develop a regression model or at least identify any significant size driver(s).

The first step was the identification of possible size drivers. Only those variables that were included in the data bases and seemed plausible to influence the number of LOC were included in the initial model.

The second step was to run the SAS regression program for each data base using all the independent variables for that data base and then conducting a systematic analysis from that point. The SAS program provided the ANOVA table and other statistics to help evaluate the model that was being tested.

For a review of the regression statistics used in this research, the reader should consult Appendix A. For a more detailed explanation of the ANOVA table and multiple regression the reader should consult reference 7 . In addition to the many regression statistics used in developing the regression models for each data base, problems of multicollinearity, outlying observations and their possible influence on the model and model specification had to be considered. Again, the reader should consult Appendix A for an explanation of each of these conditions.

Analysis of Results

This section describes the specific steps that were used to build and evaluate each software sizing model for the six data bases described earlier in this chapter. They were:

1. Because multicollinearity is usually present in nonexperimental data and can have significant effects on the other regression statistics, all the regression statistics associated with multicollinearity were checked first.

2. Because most of the data bases are small, outliers with respect to X and Y, or both, were looked for next. If any outliers were found, their influence was measured by examining Cook's D and the associated F-distribution (see Appendix A).

3. The R^2, adjusted R^2, and the standard error of the estimate values were examined. These variables are important because the R^2 value measures the amount of variation in Y (=LOC) which is explained by the independent variables in the model. The adjusted R^2 value measures this same variation, but takes into account the number of independent variables in the model. Lastly, the standard error of the estimate is a measure of the reliability of the estimate.

4. An F test and a partial F test, when necessary, were conducted next. The F test is used because it tests whether there is a regression relation between the dependent variable Y and the set of X variables. A partial F test is used to test the indepen[illegible] variables for regression relationships when multicollinearity is present.

5. If the above four steps looked promising, then the model specification was evaluated.

6. If the model specification seemed reasonable based on a priori logic, then the regression model was evaluated further. Confidence interval estimations were made for each of the population parameters. Finally, the prediction limits with a $1-\alpha$ confidence

coefficient for a new observation was calculated. The analysis in the next chapter employed the previously described methodology.

Formulation of Conclusions and Recommendations

The conclusions of this thesis were based on the analysis of the multiple regression results and an intense study of the literature. The recommendations come from the recognition that there is much more to be done in the area of software size estimation. They constitute suggestions for further research which could not be accomplished within the time and scope of this thesis effort.

IV. Analysis of Results

Chapter Overview

This chapter contains the step-by-step results of building a software sizing model for each of the six data bases. Each model is analyzed separately following the methodology discussed in the previous chapter. The statistical results are presented in summary form in the chapter. Also, each of the data bases are described in the same order as in chapter three. Finally, all the data bases are listed in Appendix B.

Developing the Software Sizing Models

The software sizing model development for each data base begins with running all of the variables for each data base in a single model. Because each variable included in each data base seems like a logical software size driver, the first model run and analyzed for a data base contains all of the functional variables. This is done in order to establish a starting point to evaluate each variable in the model to determine whether or not it should be deleted or not. Based on these results, different combinations of the variables were then run and analyzed according to the established methodology described in chapter three. If possible, further models were then developed based on these results. This process continued until no further statistically significant model could be developed.

The BMO Data Base. As described in chapter three, the BMO data base contains the variables of environment (ENV), language (LANG), number of interfaces (INTF), number of inputs (INPT), number of outputs (OUTPT), experience level (EXP), and development months (DM). It includes only seven data points.

When the first regression model was run with all the variables, a singular matrix was formed and the variables EXP and DM are found to be a linear combination of each other. However, in order to have inverse matrix, a matrix cannot have any columns that are linear combinations of each other; they all must be independent. "The rank of a matrix is defined to be the maximum number of linearly independent columns in a matrix" (7:200). Therefore, a matrix with rank less than this maximum, such as the one formed by this regression, is said to be singular, and does not have an inverse. If a matrix does not have an inverse, then regression analysis cannot be performed. Consequently, two separate models were run; one containing EXP/ENV/LANG/INTF/INPT/OUTPT and the other DM/ENV/LANG/INTF/INPT/OUTPT. Unfortunately, both of these models produced singular matrices. In the model containing EXP, it was found that EXP was a linear combination of all the other variables. In the model containing DM, it was found that OUTPT was a linear combination of all the other variables. As stated before, further analysis cannot be performed on these models. Next, a model excluding the variables EXP and DM was run. Table IV describes the summary statistics for these models. This table and the remaining tables in the chapter give the statistics that are needed to evaluate the model that the table corresponds. For example, Table IV lists only those statistics that are necessary to show that the model containing the variables of ENV, LANG, INTF, INPT and OUTPT for the BMO data base had severe multicollinearity and was therefore not useful. As can be seen in Table IV, the severe multicollinearity is first shown by the variance inflation factors for each variable. "A maximum variance inflation factor in excess of 10 is often taken as an indication that multicollinearity may be unduly influencing the least squares estimates" (7:392). The second statistic that is used to measure multicollinearity is the tolerance factor. Because the tolerance factor is the inverse of the variance inflation factor, a tolerance factor less than .1 therefore indicates that multicollinearity is probably present. The tolerance factors for each variable in the model clearly show that multicollinearity exists. The regression statistic "F value" in Table IV is calculated from the regression model. This "calculated" F value is compared with the F "table" value for the particular model. This is called the F test. This comparison tests whether there is a regression relation between the dependent variable Y and the set of X

TABLE IV

Results of Regression Model for
ENV/LANG/INTF/INPT/OUTPT

F VALUE:	N/A		
R SQUARE:	1.000		
VARIANCE INFLATION:		TOLERANCE:	
ENV:	132.29	ENV:	.00756
LANG:	39.23	LANG:	.02549
INTF:	77.74	INTF:	.01286
INPT:	2079.72	INPT:	.00048
OUTPT:	2933.50	OUTPT:	.00034

independent variables. In this model, the regression program did not report the F value because the severe multicollinearity made its value meaningless. The R^2 value is listed next in Table IV. The R^2 value measures the proportion of variance in the dependent variable explained or accounted for by the independent variable(s). In other words, it is the degree of association between the dependent variable and the independent variable(s). However, because of multicollinearity in this model, there were no unique sum of squares which had any effect in reducing the total variation in dependent variable (LOC). Therefore, because the correlation coefficient (R^2) depends on the sum of squares, it could not be calculated properly and the SAS program reported a value of one. Lastly, because of the multicollinearity in this model, the t statistics for each variable were not reported. The t calculated values (t_{calc}) are the statistics calculated from the regression model. They are compared with the model's associated t distribution to see which variables are statistically significant in the model. This is called the t test. The statistically significant variables are then used to build better regression models. Consequently, because this model could not report t values, various other combinations of the variables for the BMO data lease were run. It was found that the variables ENV and INTF always had the largest t_{calc} values. As a result, the model containing ENV and INTF (ENV/INTF) was analyzed further. (The summary statistics are shown in Table V.) Multicollinearity is not a problem since the variance inflation factors are well below 10 and the tolerance factors are above .1. However, when outliers with respect to Y were checked, it was found that the absolute values of the studentized deleted residuals for observations four and five (see the BMO data base in Appendix B) in the ENV/INTF model are higher than the associated t distribution. Outliers with respect to (w.r.t.) X and/or Y often involve large residuals and often have dramatic effects on the fitted least squares regression function. Consequently, the outlying observations must be examined and a decision reached on whether they should be retained or eliminated. Outliers w.r.t. X are identified by their leverage values being greater than two times

TABLE V

Results of Regression Model for ENV/INTF

F VALUE: .555	F_{table} (.95; 2,4) = 6.94
R SQUARE: .2172	
ROOT MSE: 40352.61	
t_{calc}:	t_{table} (.975; 5) = 2.571
ENV: -.953	
INTF: -.520	
VARIANCE INFLATION:	TOLERANCE:
ENV: 1.006	ENV: .9942
INTF: 1.006	INTF: .9942
OUTLIERS w.r.t. Y:	STUDENTIZED DELETED RESIDUAL (SDR):
t DISTRIBUTION	OBS #4: 5.3820
t(.95;3) = 2.353	OBS #5: -2.4546
INFLUENTIAL OUTLIERS:	
F DISTRIBUTION	
F(3;4) = .941 (50th Percentile)	
COOK'S D:	
OBS #4: .642 (35th Percentile)	
OBS #5: .872 (46th Percentile)	

the number of parameters (p) in the model divided by the number of observations (n) used in the model. Outliers w.r.t. Y are identified by comparing the absolute value of the observation's studentized deleted residual (the deleted residual divided by its standard deviation) with the appropriate two-tailed t distribution for the model. If the studentized deleted residual is greater than the value of the t distribution, then the observation is an outlier with respect to Y. To determine if the outlier was influential on the fitting of the regression function, the regression statistic Cook's distance measure D (Cook's D) is used. This measure shows in the aggregate the differences between the fitted values for each observation when all n observations are used in the data base and the fitted values when the i^{th} observation is deleted. The Cook's D for each outlier is then compared to the model's appropriate F distribution. The rule of thumb states that if the Cook's D value is less than the 20th percentile of the associated F distribution, then the outlier is not influential. If the Cook's D is greater than the 50th percentile level, then the outlier is influential. Table V shows that the Cook's Ds for observations four and five, by interpolation, are not above the 50th percentile level. They fall in the questionable region (between the 20th and 50th percentiles). Therefore, because the data base is small to begin with and they are not above the 50th percentile level, they were kept in the data base. Even though there was no multicollinearity or influential outliers, the R^2 was only .2172. More importantly, both the F test ($F_{calc} = .555 < F_{table} = 6.94$) and the t test on the independent variables fail ($t_{calc} = |-.953| < t_{table} = 2.571$ and $t_{calc} = |-.520| < t_{table} = 2.571$). The F test failure means that there was not significant relationship between the dependent variable LOC and the independent variables ENV and INTF. The failure of the t test on each independent variable means that neither variable was statistically significant in helping to predict LOC.

Finally, each of the seven independent variables were run separately to determine if there was a relationship between the independent variable and LOC. (The results are in Table VI.) As can be seen none of the variables were statistically significant; all models and variables failed their F tests and t tests respectively. Only the R^2 for the variable ENV was noteworthy.

In summary, no statistically significant model able to predict LOC for the BMO data base could be developed. However, it must be kept in mind that this was a very small data base; only seven data points. The best model found had a correlation coefficient (R^2) value of .2172 and contained the variables of ENV (the environment the software will operate in) and INTF (the number of interfaces in the program). If the BMO data base can be enlarged, the chances are good that the variables ENV and INTF can be proven to be statistically significant. This larger data base could also cause the other independent variables to become significant.

The ESD Data Base. The ESD data base contains seven functional variables in addition to the number of LOC for each of 26 data points. The variables are: environment (ENV), development

TABLE VI

Results of Individual Regression Models for
ENV, LANG, INTF, INPT, OUTPT, EXP, and DM

ENV:		LANG:	
F VALUE:	.871	F VALUE:	.145
R SQUARE:	.1789	R SQUARE:	.0351
t_{calc}:	-.933	t_{calc}:	.381
INTF:		INPT:	
F VALUE:	.261	F VALUE:	.006
R SQUARE:	.0613	R SQUARE:	.0016
t_{calc}:	-.511	t_{calc}:	-.080
OUTPT:		EXP:	
F VALUE:	.000	F VALUE:	.148
R SQUARE:	.0001	R SQUARE:	.0356
t_{calc}:	-.021	t_{calc}:	-.384
DM:			
F VALUE:	.167		
R SQUARE:	.0401		
t_{calc}:	-.409		

F_{table} (.95;1,5) = 6.61

t_{table} (.975;5) = 2.571

hours in manhours (DHRS), language (LANG), reliability (REL), complexity (COMPX), experience level of programmers (EXP), and the quality of the specification (QSPEC).

As before, a regression model containing all the independent variables was run first to establish a starting point. (The results are summarized in Table VII.) First, multicollinearity was not a problem because the variance inflation factors were below 10 and the tolerance factors above .1. Second, outliers w.r.t. X and Y were looked for. There were no outliers w.r.t. X, but there were three observations as outliers w.r.t. Y. However, none of these outliers were influential. (See Table VII.) Even though these two conditions were not problems the F test for the model fails ($F_{calc} = .608 < F_{table} = 2.58$) and the R^2 value (.1912) is poor. Finally, all of the independent variables fail their individual t tests at the ($1-\alpha/2 =$) .975 level of confidence.

As before, various combinations of the independent variables were run. These results, in turn, lead to further models. It was found that the variables QSPEC and EXP consistently had the largest t_{calc} values. Therefore, a model containing QSPEC and EXP was analyzed.

Because of the fairly large data base and the need to test any significant model on a known observation in order to test the model's prediction capability, a data point was randomly deleted from the ESD data base (observation #6). Therefore, using the remaining 25 data points the QSPEC/ENV regression model was run. (The results are summarized in Table VIII.) First, there was no multicollinearity because the variance inflation factors are less than 10 and the

TABLE VII

Results of Regression Model for ENV/DHRS/
LANG/REL/COMPX/EXP/QSPEC

F VALUE:	.608	VARIANCE INFLATION:	
		ENV:	1.497
R SQUARE:	.1912	DHRS:	1.406
		LANG:	1.209
ADJ R SQUARE:	-.1233	REL:	1.572
		COMPX:	1.452
ROOT MSE:	17033.26	EXP:	1.373
		QSPEC:	1.308
F_{table} (.95;7, 18) = 2.58		TOLERANCE:	
		ENV:	.668
t_{table} (.975;24) = 2.064		DHRS:	.711
		LANG:	.827
		REL:	.636
t_{calc}:		COMPX:	.689
ENV:	- .568	EXP:	.728
DHRS:	.487	QSPEC:	.765
LANG:	.249		
REL:	-1.126		
COMPX:	1.286		
EXP:	- .725		
QSPEC:	1.041		
OUTLIERS w.r.t. Y:		STUDENTIZED DELETED RESIDUAL (SDR):	
OBS #:			
3		2.3486	
5		2.8730	
6		2.8278	
t DISTRIBUTION (.95; 17) = 1.740			
INFLUENTIAL OBSERVATIONS:		COOK'S D:	
F (8; 16) = 2.564		OBS #3:	.097
		OBS #5:	1.633
		OBS #6:	1.088

TABLE VIII

Results of Regression Model for QSPEC/EXP

QSPEC/EXP:

F VALUE:	6.163
R SQUARE:	.3591
ADJ R SQUARE:	.3008
ROOT MSE:	12948.24

F_{table} (.95; 2,22) = 3.445

t_{calc}:

QSPEC:	3.391
EXP:	-2.281

OUTLIERS w.r.t. X:
2p/n = .24

OUTLIERS w.r.t Y:
t DISTRIBUTION:
t (.95; 21) = 1.721

INFLUENTIAL OUTLIERS:
F DISTRIBUTION:
F (3,22) = .814 (50th Percentile)

COOK'S D VALUES:

OBS # 1 =	.362
OBS # 2 =	.306
OBS # 3 =	.211
OBS # 5 =	.776
OBS # 6 =	.012
OBS #17 =	.071

VARIANCE INFLATION:

QSPEC:	1.227
EXP:	1.227

TOLERANCE:

QSPEC:	.815
EXP:	.815

t_{table} (.975; 23) = 2.069

LEVERAGE VALUES:

OBS #1 =	.3998
OBS #2 =	.3532
OBS #5 =	.4557
OBS #6 =	.3842

SDR:

OBS #3 =	2.4061
OBS #5 =	1.7426
OBS #17 =	2,3841

MODEL: LOC = -16977.4 + 28268.83 QSPEC - 393.079 EXP tolerance factors greater than .1. Second, outliers w.r.t. X and Y are also identified in Table VIII. As can be seen none of the outliers were influential. Third, since $F_{calc} = 6.163$ was greater than $F_{table} = 3.445$ at the 95 percent level of confidence, there is a regression relationship between the two independent variables (QSPEC and EXP) and the dependent variable (LOC). Also, note that the individual t tests on QSPEC and EXP proved significant at the $(1-\alpha/2 =)$ .975 level of confidence, thus indicating linear associations between the independent variables (QSPEC and EXP) and the dependent variable (LOC). Fourth, the model has a correlation coefficient (R^2) of only .3591 which means it explains or accounts for a little more than a third of the variation in the data. The next important statistic is the standard error of the estimate (ROOT MSE in Table VIII.) The standard error of the estimate is quite high (12948.24 lines of code) indicating poor prediction capabilities for the model. The last major criteria to be examined was the model specification. Unfortunately, the model specification seems to be only half correct. A priori logic would suggest that the sign of QSPEC should be negative. That is, by defining what the program should do in some detail before programming actually begins, the programmers should have a better idea of what to include in the program. Therefore, the total number of lines of code should be reduced. (Unfortunately, the ESD data base numbers for QSPEC do not support this logic; more data is needed to prove the a priori logic one way or another.) This, however, is not what the regression produced. The regression coefficient for QSPEC is +28268.83. The a priori logic for EXP would also suggest a negative relationship. If the programmers have a lot of experience, this should help reduce LOC because they should be able to write more efficient programs. This is the case for EXP in the model. (Again, as with QSPEC, there is not enough variation in the data for EXP to support this logic. There are too many other variables influencing LOC and not enough data in this data base to graphically illustrate the a priori logic.) The regression model produced is LOC = -16977.4 + 28268.83 QSPEC - 393.079 EXP.

Because the model is statistically significant based on the F test and t test, even though the correlation coefficient is low and the model specification possibly wrong, confidence interval estimations were made for the two population parameters β_1 (QSPEC) and β_2 (EXP). (See Appendix A for the formula.) (Table IX shows the calculations.) As can be seen, there is a very wide range for both values due to the large standard errors for each variable. Finally, a prediction using the model developed was made using the QSPEC and EXP data from the previously deleted data point (observation #6 in the ESD data base). For this data point QSPEC = 1 and EXP = 54 months. The prediction is:

$$\text{LOC} = -16977.4 + 28268.83(1) - 393.079(54) = -9934.836 \qquad (3)$$

TABLE IX

Confidence Interval Estimations for the QSPEC/EXP Model

β_1 (QSPEC):

$\alpha = .05 \quad n = 25 \quad p = 3$

$28269 \pm t(.975; 22)\ (8337)$
$28269 \pm 2.074\ (8337)$
28269 ± 17291
$10978 - 45560$

β_2 (EXP):

$\alpha = .05 \quad n = 25 \quad p = 3$

$-393 \pm t(.975, 22)\ (172)$
$-393 \pm 2.074\ (172)$
-393 ± 357
$(-750) - (-36)$

The predicted value for LOC is negative and therefore makes no sense. The actual value is 47525 lines of code. This is because the R^2 value is low, the estimated regression coefficients can vary over an extremely wide range, and the model specification is most likely wrong.

In summary, the independent variables QSPEC and EXP together in a model have been shown to be statistically significant. However, the model has a low correlation coefficient ($R^2 = .3591$), high standard errors, and probably the wrong model specification.

Further testing of this model with a larger data base may correct the model specification, lower the standard errors, and raise the R^2 value.

The AD Data Base--Ground Systems. As previously described, the AD data base was divided into ground programs and airborne programs. This section will describe the analysis of the ground programs. This data base contains 12 data points with one data point already randomly deleted for use in predictions. The ground program AD data base contains six functional variables in addition to the number of LOC. They are: development months (DM), language (LANG), quality of specification (QSPEC), reliability (REL), function or the type of system (missile, range, or munition) the program functions in.

Again, a regression model containing all the independent variables was run first. (The results are summarized in Table X.) Because DM data was only available for eight of the twelve data points, the SAS regression program only used these eight data points any time DM was included in a model. As can be seen, the multicollinearity statistics do not indicate multicollinearity. However, the outliers w.r.t. Y are very influential as noted by the Cook's D values; all are well over the 50th percentile level for the associated F distribution. (See Table X.) As explained earlier, this means that the regression function was greatly distorted and resulted in the

TABLE X

Results of Regression Model for DM/LANG/QSPEC/REL/FUNC/COMPX

F VALUE:	5.426
R SQUARE:	.9702
ADJ R SQUARE:	.7914
ROOT MSE:	12131.24

VARIANCE INFLATION:

DM:	4.495
LANG:	1.523
QSPEC:	5.884
REL:	4.388
FUNC:	3.932
COMPX:	2.658

F_{table} (.95; 6, 1) = 234

t_{table} (.975; 6) = 2.447

TOLERANCE:

DM:	.222
LANG:	.657
QSPEC:	.170
REL:	.228
FUNC:	.254
COMPX:	.376

$t_{calculated}$:

DM:	.303
LANG:	-.859
QSPEC:	.267
REL:	-.762
FUNC:	-1.769
COMPX:	3.046

F DISTRIBUTION FOR COOK'S D
F(7, 1) = .506 (50th Percentile)

COOK'S D:

OBS #1:	2.615	OBS #6:	2.540
OBS #2:	3.131	OBS #7:	3570.328
OBS #5:	2.540	OBS #8:	2.694

high R^2 value. Even though the R^2 value is at the 97 percent level, the model fails the F test ($F_{calc} = 5.426 < F_{table} = 234$) indicating no regression relationship. Lastly, each of the six independent variables were evaluated using the t test. COMPX proved significant at the ($1-\alpha/2 =$) .975 level of confidence ($t_{calc} = 3.046 > t_{table} = 2.447$).

Because COMPX proved statistically significant in the first model, various other combinations of the independent variables were tested with COMPX. Unfortunately, no statistically significant models were found.

Lastly, individual models for each of the six independent variables were run. (The results are in Table XI.) Note, that observation one for the AD--Ground data base (see Appendix B) was an influential outlier w.r.t. Y for the COMPX and DM models. However, it was decided to leave it in the data base because of the small size of the data base (only 12 data points) and because the data point is correct and simply represents an unlikely event which could very well occur again (7:409). As can be seen, two of the variables are statistically significant. COMPX is significant at the 95 percent level of confidence for the F test. The t test on COMPX is significant at the ($1-\alpha/2 =$) .975 level of confidence. DM is significant at the 90 percent level for the F test and for the t test. DM is significant at the ($1-\alpha/2 =$) .95 level of confidence.

As noted before, all combinations of the independent variables

TABLE XI

Results of Regression Models for DM, LANG, QSPEC, REL, FUNC, and COMPX

DM:		LANG:	
F VALUE:	4.361	F VALUE:	.247
R SQUARE:	.4209	R SQUARE:	.0241
t_{calc}:	2.088	t_{calc}:	-.497
QSPEC:		REL:	
F VALUE:	.252	F VALUE:	.269
R SQUARE:	.0246	R SQUARE:	.0262
t_{calc}:	-.502	t_{calc}:	-.519
FUNC:		COMPX:	
F VALUE:	1.492	F VALUE:	3.166
R SQUARE:	.1298	R SQUARE:	.2602
t_{calc}:	-1.221	t_{calc}:	1.779

FOR LANG, QSPEC, REL, FUNC, COMPX:

$F_{table}\ (.95;\ 1, 10) = 4.96$

$F_{table}\ (.90;\ 1, 10) = 3.29$

$t_{table}\ (.975;\ 10) = 2.228$

$t_{table}\ (.95;\ 10) = 1.812$

FOR DM:

$F_{table}\ (.90;\ 1, 6) = 3.78$

$t_{table}\ (.95,\ 5) = 2.015$

were tried. Because COMPX and DM proved significant individually, the model containing both COMPX and DM was tested again. (The results are in Table XII.) For the reasons stated earlier in this section, the influential Y outlier (observation #1) was left in the data base. The results indicate that neither the model nor the independent variables are statistically significant. The model fails the F test at both the 95 and 90 percent levels of confidence. The variables fail the t test at both ($1-\alpha/2 = .975$ and $1-\alpha/2 = .95$) levels of confidence.

In summary, no combination of the six independent variables proved statistically significant. When the six independent variables were tested individually, COMPX and DM proved statistically significant at the 95 percent and 90 percent levels of confidence respectively. It should be noted, however, that in both cases an influential, but never-the-less correct, observation was left in the data base. Finally, it may be possible with a larger data base, especially with more data for the variable DM, that a statistically significant model containing COMPX and DM and possibly some other variable could be developed.

<u>The AD Data Base--Airborne Systems</u>. This portion of the AD data base contains the programs written for the airborne systems. The data base contains 13 data points with one data point already randomly deleted for use in prediction tests of any suitable model developed. The airborne program AD data base contains six

TABLE XII

Results of Regression Model for COMPX/DM

F VALUE: 3.496

R SQUARE: .5831

ADJ R SQUARE: .4163

t_{calc}:
COMPX: .3913
DM: .2219

ROOT MSE: 20292.87

OUTLIERS w.r.t. X:
2p/n = .5

LEVERAGE VALUES:
OBS #1 = .586
OBS #2 = .690

INFLUENTIAL OUTLIERS:
F DISTRIBUTION:
F(3,9) = .852

VARIANCE INFLATION:
COMPX: 1.584
DM: 1.584

TOLERANCE:
COMPX: .631
DM: .631

F_{table} (.95; 2,5) = 5.79

F_{table} (.90; 2,5) = 3.78

t_{table} (.975; 6) = 2.447

t_{table} (.95; 6) = 1.943

OUTLIERS w.r.t. Y:
t DISTRIBUTION:
t(.95; 8) = 1.860

SDR:
OBS #1 = 4.609

COOK'S D VALUES:
OBS #1 = 1.981
OBS #2 = .008

functional variables in addition to the number of LOC. They are: development months (DM), language (LANG), quality of specification

(QSPEC), reliability (REL), the function (FUNC) the program will support (missile, range, or munition), and complexity (COMPX).

First, a model containing all of the variables was run and analyzed. (The results are summarized in Table XIII.) As before, only a portion of the data base has DM data (nine out of thirteen). Unfortunately, multicollinearity has inflated the R^2 value. The two independent variables DM and COMPX both have variance inflation factors over 10 and tolerance factors below .1. Also, REL is on the border of contributing to the multicollinearity (variance inflation = 9.692 and tolerance = .103). The F test for the model fails at the 95 percent level but does pass at the 90 percent level. Finally, only two of the six independent variables, DM and FUNC, are statistically significant at the ($1-\alpha/2 =$) .975 level of confidence.

As before, these results were used to test various combinations of the variables, especially with DM and FUNC. After considerable testing, a model with FUNC and REL seems good. (See Table XIV for the results.) Two of the 13 original data points have been deleted because of severe outlier problems--observations one and six (see the AD--Airborne data base in Appendix B). First, there is no multicollinearity because the variance inflation factors are less than 10 and the tolerance values greater than .1. Second, the model passes the F test at the 95 percent level of confidence and the t tests on the independent variables are significant at the ($1-\alpha/2 =$) .975 level of

TABLE XIII

Results of Regression Model for DM/LANG/
QSPEC/REL/FUNC/COMPX

F VALUE:	14.177
R SQUARE:	.9770
ADJ R SQUARE:	.9081

F_{table} (.95; 6,2) = 19.13

F_{table} (.90; 6,2) = 9.33

t_{table} (.975; 7) = 2.365

t_{calc}:

DM:	3.625
LANG:	-1.632
QSPEC:	.364
REL:	1.752
FUNC:	-3.558
COMPX:	-1.692

VARIANCE INFLATION:

DM:	10.952
LANG:	5.768
QSPEC:	3.546
REL:	9.692
FUNC:	3.689
COMPX:	26.627

TOLERANCE:

DM:	.091
LANG:	.173
QSPEC:	.282
REL:	.103
FUNC:	.271
COMPX:	.038

confidence. Next, the correlation coefficient (R^2) is a respectable .6583. Fourth, the model specification also appears good. REL is directly related to LOC; i.e., the more reliable the software program must be the more the lines of code should increase. (See Figure 1.) FUNC is inversely related to LOC; i.e., it seems reasonable that programs written for a certain function should generally be about the

TABLE XIV

Results of Regression Model for FUNC/REL

F VALUE: 7.707

R SQUARE: .6583

ADJ R SQUARE: .5729

t_{calc}:
- FUNC: -2.728
- REL: 2.329

ROOT MSE: 6795.236

STANDARD ERROR TERMS:
- FUNC: 3169.62
- REL: 1829.981

$F_{table}(.95; 2,8) = 4.46$

$t_{table}(.975; 9) = 2.262$

VARIANCE INFLATION:
- FUNC: 1.029
- REL: 1.029

TOLERANCE:
- FUNC: .972
- REL: .972

MODEL: $LOC = 9757.322 - 8647.28\ FUNC + 4262.762\ REL$

CONFIDENCE INTERVAL ESTIMATIONS OF β_1 (FUNC) AND β_2 (REL):

$\alpha = .05 \quad n = 11 \quad p = 3$

FUNC: $-8647 \pm t(.975; 8)\ (3170)$
$-8647 \pm 2.306\ (3170)$
$(-15957) - (-1337)$

REL: $4263 \pm 2.306\ (1830)$
$43 - 8483$

same length or fall within certain ranges. Therefore, since the function should always be known, the programmer will have a general idea of how many lines of code the program might have and thus will help lower the initial estimate of the number of lines of code.

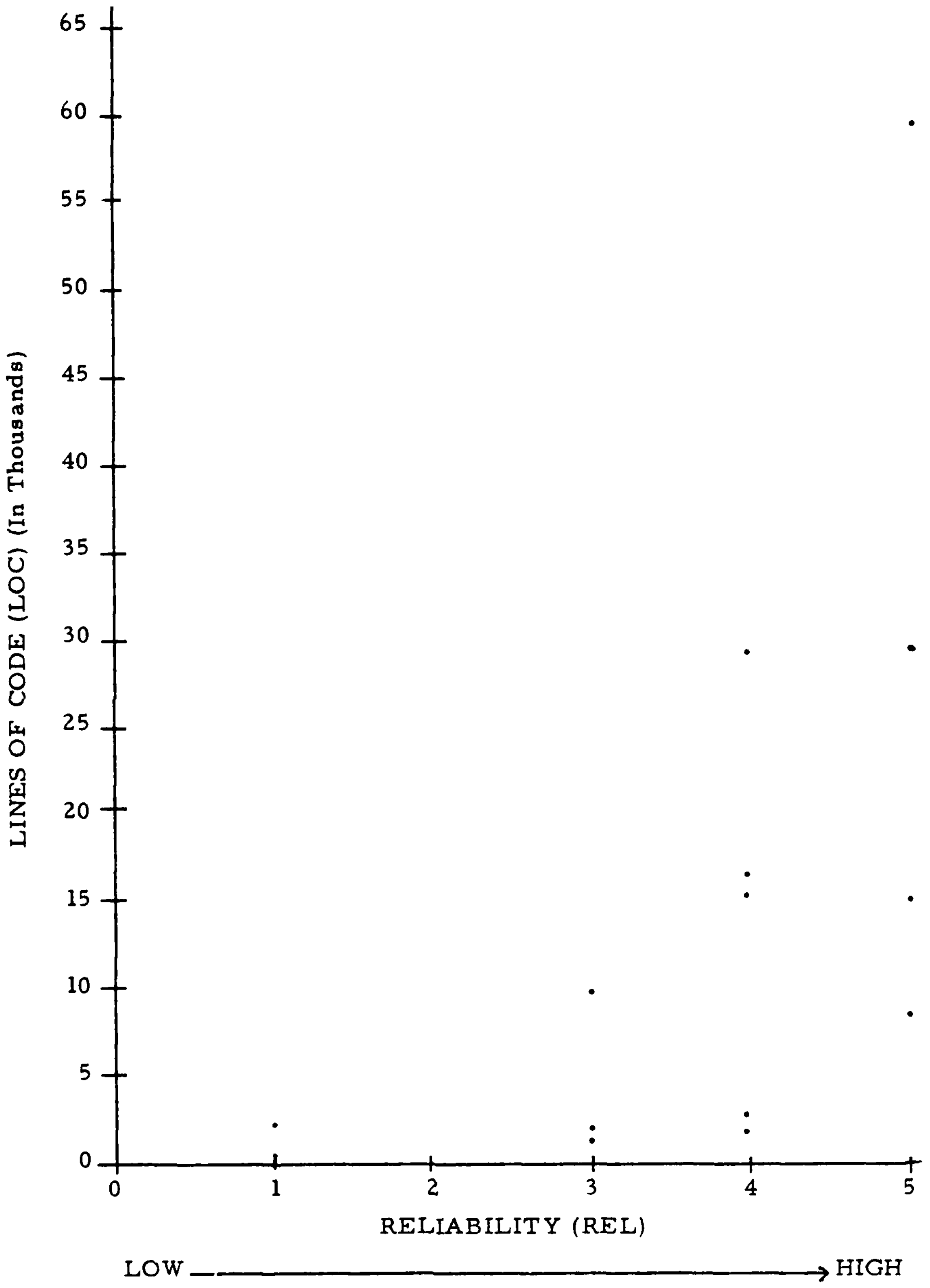


Figure 1. Lines of Code Versus Reliability

However, this small data base cannot show this relationship. Fifth, the linearity of the model is shown by the randomness of the points in the net scatter diagrams for the two independent variables. (See Figure 2 and Figure 3.) Unfortunately, the standard error term for the model (ROOT MSE = 6795 lines of code) is relatively high for accurate predictions. Finally, in Table XIV, confidence level intervals at the 95 percent level have been computed to show the ranges for the true population parameters (β_1 = FUNC and β_2 = REL.)

Another model with three variables is also good, but the extra variable does not help in any way. This model is the same as the previous one, but with QSPEC added. The results are in Table XV. Because another independent variable is added, the R^2 value is slightly higher than in Table XIV; but the adjusted R^2 value is lower, indicating QSPEC could not explain more of the variation in the data to make up for the degrees of freedom lost by adding QSPEC. As can be seen, multicollinearity is not a problem. Therefore, individual t tests were used. Both FUNC and REL are significant at $\alpha = .05$, but QSPEC is not significant until $\alpha = .10$. Furthermore, this is a good example of why the adjusted R^2 value should also be looked at when new variables are added and deleted from a basic model. The conclusion is that QSPEC adds no more explanatory power to the previous model.

A fourth model containing FUNC/COMPX/REL was tested next.

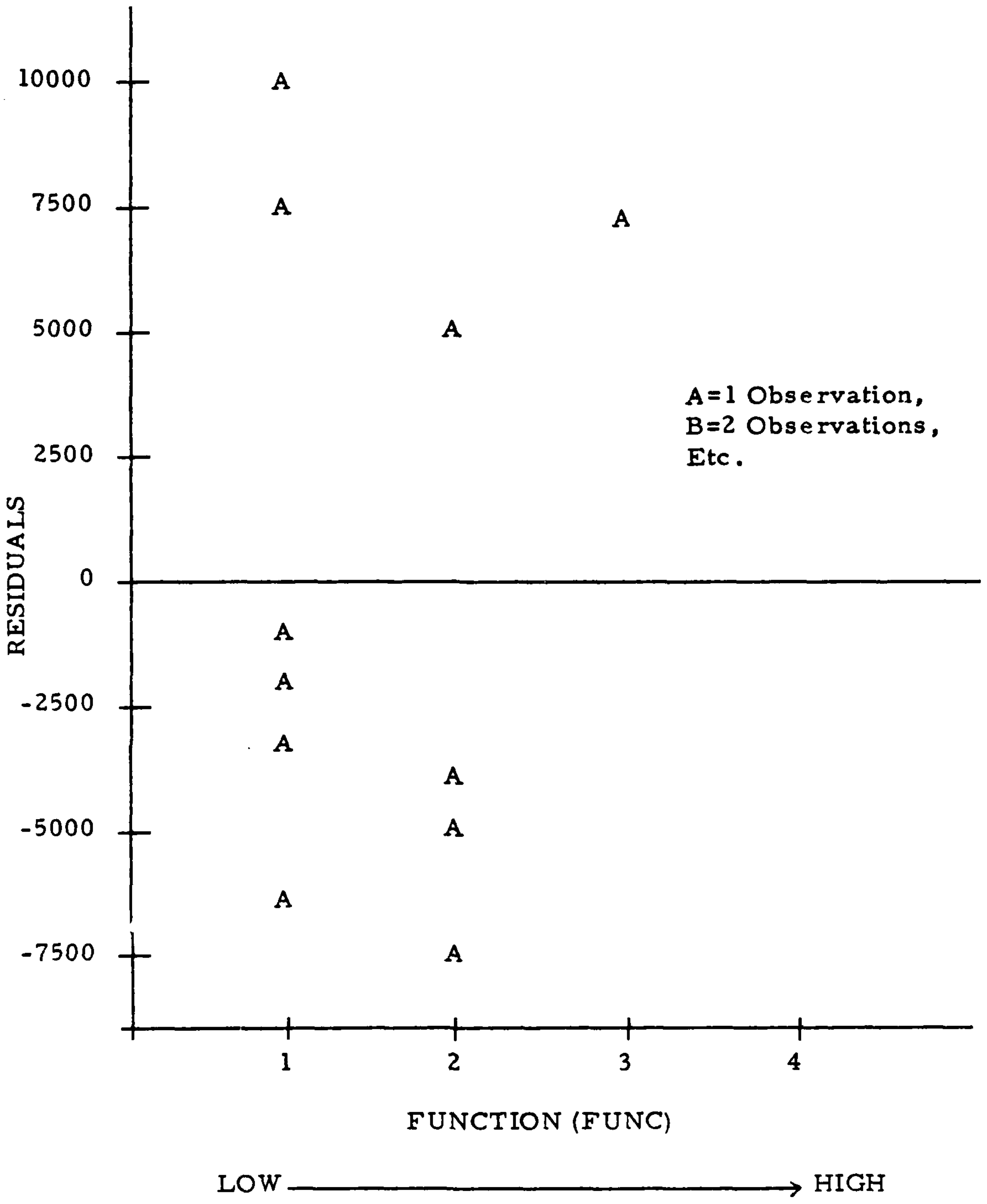


Figure 2. Net Scatter Diagram--Residuals Versus Function

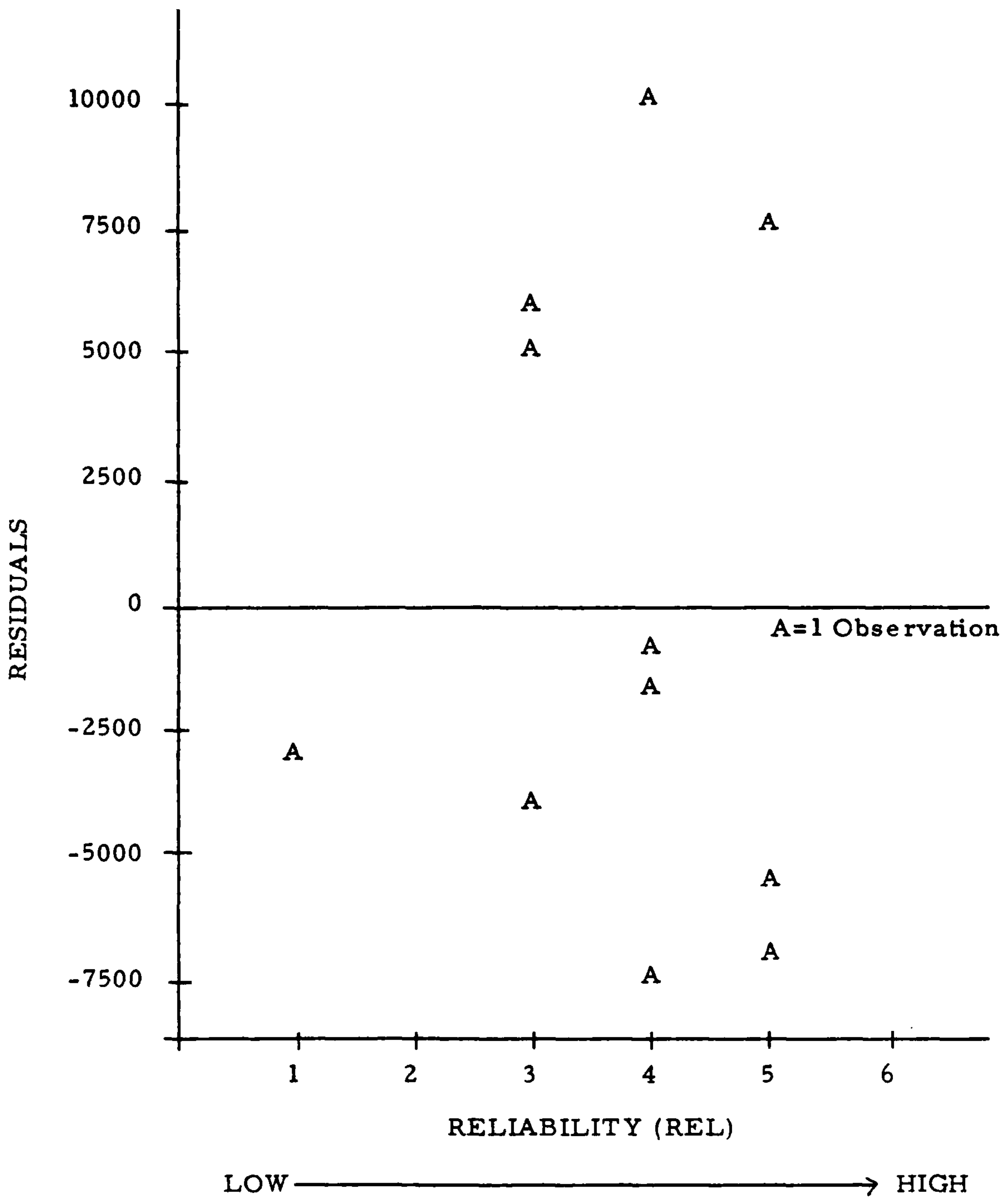


Figure 3. Net Scatter Diagram--Residuals Versus Reliability

TABLE XV

Results of Regression Model for FUNC/REL/QSPEC

F VALUE:	5.708
R SQUARE:	.6816
ADJ R SQUARE:	.5622
t_{calc}:	
FUNC:	-2.728
REL:	2.329
Q SPEC:	-2.076
ROOT MSE:	6795.236
STANDARD ERROR:	
FUNC:	3169.62
REL:	1829.981
QSPEC:	7302.308

F_{table} (.95; 3,7)	= 4.35
F_{table} (.90; 3,7)	= 3.07
t_{table} (.975; 9)	= 2.262
t_{table} (.95; 9)	= 1.833
VARIANCE INFLATION:	
FUNC:	1.088
REL:	1.033
QSPEC:	1.056
TOLERANCE:	
FUNC:	.919
REL:	.968
QSPEC:	.945

MODEL: LOC = 40079.5 - 8647.28 FUNC + 4262.762 REL - 15161.1 QSPEC

The results are in Table XVI. The F test is significant at the 95 percent level. Note again the lower adjusted R^2 value compared to the one in Table XIV. Also, in this model multicollinearity is somewhat high to COMPX and REL. Because COMPX and REL have low t_{calc} values, probably because of the multicollinearity, a partial F test was conducted to test the hypothesis H_o:β_2 (COMPX) = β_3 (REL) = 0. (See Appendix A for the equation.) The reduced model, therefore, only

TABLE XVI

Results of Regression Model for FUNC/COMPX/REL

F VALUE:	5.310	F_{table} (.95; 3,7) = 4.35	
R SQUARE:	.6947	t_{table} (.975; 9) = 2.262	
ADJ R SQUARE:	.5639		
		VARIANCE INFLATION:	
t_{calc}:		FUNC:	1.607
FUNC:	-2.708	COMPX:	5.865
COMPX:	- .914	REL:	4.822
REL:	1.875		
ROOT MSE:	6866.438	TOLERANCE:	
		FUNC:	.622
		COMPX:	.170
		REL:	.207

FULL MODEL (FUNC/COMPX/REL):
SS ERROR = 330035821
DEGREES OF FREEDOM (ERROR) = 7

REDUCED MODEL (FUNC):
SS ERROR = 619953846
DEGREES OF FREEDOM (ERROR) = 9

$$\text{PARTIAL } F_{calc} = \frac{\dfrac{619953846 - 330035821}{9-7}}{\dfrac{330035821}{7}} = 3.075$$

F_{table} (.95; (9-7),7) = (.95; 2,7) = 4.74

F_{table} (.90; 2,7) = 3.26

contained FUNC. (See Table XVI.) Because partial F_{calc} is less than partial F_{table}, H_o is concluded. This means $\beta_2 = \beta_3 = 0$ and together they are not significant and the reduced model should be used. In other words, the model containing FUNC/COMPX/REL is really insignificant.

Finally, the independent variables were tested in separate models. (The results are summarized in Table XVII.) As can be seen, the DM model passes the F test at the 95 percent level of confidence and the independent variable DM passed the t test at the ($1-\alpha/2$ =) .975 level of confidence. Also note that FUNC and REL are both significant at the 90 percent level of confidence.

It should be noted that many models with DM as a variable were tested, because DM is the most significant variable by itself. However, in each case multicollinearity was present which distorted the regression statistics. However, if more data points for DM could be found, DM along with some of the other variables could prove to produce excellent models.

In summary, three of the six variables are significant by themselves--DM (95 percent level) and FUNC and REL (90 percent level). The model with FUNC/REL proved to be good, but with high standard errors. Two other models, both containing FUNC and REL and one other variable, when evaluated, showed that the third variable was not helpful in improving the prediction capability.

The Assembly Language Data Base. As described in chapter three, this data base and the following data base contain programs divided into the two languages found most prevalent in the four major

TABLE XVII

Results of Regression Models for DM, LANG, QSPEC, REL, FUNC, and COMPX

DM:		LANG:	
F VALUE:	12.628	F VALUE:	.040
R SQUARE:	.6434	R SQUARE:	.0036
t_{calc}:	3.554	t_{calc}:	-.200
QSPEC:		REL:	
F VALUE:	.563	F VALUE:	4.786
R SQUARE:	.0487	R SQUARE:	.3032
t_{calc}:	-.751	t_{calc}:	2.188
FUNC:		COMPX:	
F VALUE:	3.44	F VALUE:	2.284
R SQUARE:	.2383	R SQUARE:	.1859
t_{calc}:	-1.855	t_{calc}:	1.511

F_{table} AND t_{table} FOR LANG, QSPEC, REL, AND FUNC:

F_{table} (.95; 1, 11) = 4.84

F_{table} (.90; 1, 11) = 3.24

t_{table} (.975; 11) = 2.201

t_{table} (.95; 11) = 1.796

F_{table} AND t_{table} FOR DM:

F_{table} (.95; 1,7) = 5.59

t_{table} (.975; 7) = 2.365

data bases used in this research--Assembly and Fortran. This section describes the analysis on the Assembly data base. The data base contains 30 data points. It is described by four functional variables: quality of specification (QSPEC), reliability (REL), environment (ENV), and complexity (COMPX). It should be noted that only 15 of the data points contain all four variables, while all 30 data points contain the ENV and COMPX variables. (See Appendix B.)

As before, the first model contained all four independent variables for the Assembly data base. Unfortunately, as with the BMO data base, a singular matrix is formed when all four variables are regressed against LOC. It is found that QSPEC is a linear combination of two times the intercept value because all the data for QSPEC is rated nominal (a value of 2). As noted before, a singular matrix is not full rank and does not have an inverse matrix. Therefore, all the regression statistics are meaningless. Consequently, the remaining three variables were regressed against LOC. (The results are summarized in Table XVIII.) There is no multicollinearity and no influential outliers. Unfortunately, the model fails the F test at the 95 and 90 percent levels of confidence and all the variables fail their individual t tests indicating no linear relationships at all. The highest t_{calc} value is 1.427 for COMPX. Even this fails at the 90 percent level.

Next because QSPEC cannot be used in any model, as explained

TABLE XVIII

Results of Regression Model for ENV/COMPX/REL

F VALUE:	1.566	$F(.95; 3,11) = 3.59$	
R SQUARE:	.2993	$F(.90; 3,11) = 2.67$	
ADJ R SQUARE:	.1081		
		$t_{calc}(.975; 13) = 2.160$	
t_{calc}:		$t_{calc}(.95; 13) = 1.771$	
ENV:	-.948		
COMPX:	1.427		
REL:	-.946		
		VARIANCE INFLATION:	
		ENV:	1.755
		COMPX:	1.971
		REL:	2.058
		TOLERANCE:	
		ENV:	.570
		COMPX:	.507
		REL:	.486

above, the remaining three models that could be run with two variables in each model were analyzed. (See Table XIX for the results.) The only conclusion that can be drawn is that COMPX is the only statistically significant variable.

In summary, the only significant variable for this data base is COMPX. Also, in order to test the quality of specification (QSPEC), more data points with different levels of the quality of specification need to be acquired. Finally, the variables of ENV and REL prove to

TABLE XIX

Results of Regression for Models ENV/COMPX, REL/COMPX, and ENV/REL

ENV/COMPX:			
F VALUE:	3.112	$F_{table}(.95;\ 2,27)$ =	3.35
R SQUARE:	.1874	$F_{table}(.90;\ 2,27)$ =	2.52
ADJ R SQUARE:	.1272	$t_{table}(.975;\ 28)$ =	2.048
t_{calc}:		VARIANCE INFLATION:	
ENV:	.175	ENV:	1.168
COMPX:	2.369	COMPX:	1.168
ROOT MSE:	7968.37		
STANDARD ERROR:		TOLERANCE:	
ENV:	4625.868	ENV:	.856
COMPX:	1863.292	COMPX:	.856

REL/COMPX:			
F VALUE:	1.916	$F_{table}(.95;\ 2,12)$ =	3.89
R SQUARE:	.2420	$F_{table}(.90;\ 2,12)$ =	2.81
ADJ R SQUARE:	.1157	$t_{table}(.975;\ 13)$ =	2.160
t_{calc}:		$t_{table}(.95;\ 13)$ =	1.771
REL:	-.648		
COMPX:	1.812	VARIANCE INFLATION:	
ROOT MSE:	8821.518	REL:	1.786
		COMPX:	1.786
		TOLERANCE:	
		REL:	.560
		COMPX:	.560

TABLE XIX

Results of Regression for Models ENV/COMPX, REL/COMPX, and ENV/REL (Continued)

ENV/REL:		
F VALUE:	1.224	$F_{table}(.95; 2,12) = 3.89$
R SQUARE:	.1694	$F_{table}(.90; 2,12) = 2.81$
ADJ R SQUARE:	.0310	$t_{table}(.975; 13) = 2.160$
t_{calc}:		$t_{table}(.95; 13) = 1.771$
ENV:	-1.396	
REL:	- .289	VARIANCE INFLATION:
ROOT MSE:	9234.244	ENV: 1.590
		REL: 1.590
		TOLERANCE:
		ENV: .629
		REL: .629

be very ineffective by themselves, when in combination with each other, or when in combination with COMPX.

The Fortran Language Data Base. This section describes the analysis on the Fortran data base. The data base contains 55 data points. It is described by the same four variables as the Assembly data base: quality of specification (QSPEC), reliability (REL), environment (ENV) (ground or airborne), and complexity (COMPX). It should be noted that only 21 of the 55 data points contain all four variables. All 55 data points contain ENV and COMPX data.

As usual, the first model contained all four independent variables. (The results are summarized in Table XX.) First of all, there is no multicollinearity because the variance inflation factors for each variable is less than 10 and the tolerance factors for each variable is greater than .1. Secondly, there are no influential outliers. Unfortunately, the F test for the model fails at both the 95 and 90 percent levels of confidence which indicates no regression relationship between the four independent variables and the dependent variable LOC. However, the individual t tests indicate that COMPX (95 percent level) and ENV (90 percent level) are related to LOC.

Next, various combinations of the independent variables were tested. As in the Assembly data base models, only the model containing the independent variables COMPX and ENV proved to be statistically significant at the 95 percent level of confidence. (See Table XXI for the results.) Also, as in the Assembly model of COMPX/ENV, only COMPX is significant (95 percent level) in the t tests.

Finally, the four independent variables (QSPEC, REL, ENV, and COMPX) were tested in individual models. (The results are summarized in Table XXII.) As can be seen, the independent variables QSPEC, REL, and ENV all have extremely poor models. All F tests and t tests fail at the 95 and 90 percent levels of confidence. The first model for COMPX indicated that the second observation in

TABLE XX

Results of Regression Model for QSPEC/REL/ENV/COMPX

F VALUE:	2.014	$F_{table}(.95; 4,16) = 3.01$
R SQUARE:	.3348	$F_{table}(.90; 4,16) = 2.32$
ADJ R SQUARE:	.1685	
		$t_{calc}(.975; 19) = 2.093$
t_{calc}:		
QSPEC:	.306	$t_{calc}(.95; 19) = 1.729$
REL:	-.179	
ENV:	-1.879	
COMPX:	2.359	VARIANCE INFLATION:
		QSPEC: 1.014
ROOT MSE:	17829.68	REL: 1.589
		ENV: 1.328
		COMPX: 1.407
		TOLERANCE:
		QSPEC: .986
		REL: .629
		ENV: .753
		COMPX: .711

the data base (see Appendix B) produced an influential outlier w.r.t. Y. Consequently, this data point was eliminated--mainly because of the large data base and the model rerun. As can be seen in Table XXII, COMPX has a very strong linear relationship (see net scatter diagram in Figure 4) between itself and LOC. Both the F test and t test are overwhelmingly superior at the 95 percent level of confidence. The R^2 value suggests that about 25 percent of the variation

TABLE XXI

Results of Regression Model for ENV/COMPX

F VALUE:	9.655	$F_{table}(.95; 2,51) = 3.18$
R SQUARE:	.2746	$t_{table}(.975; 52) = 2.01$
ADJ R SQUARE:	.2462	$t_{table}(.95; 52) = 1.675$
t_{calc}:		
ENV:	-1.282	VARIANCE INFLATION:
COMPX:	4.218	ENV: 1.430
		COMPX: 1.430
ROOT MSE:	9696.29	
STANDARD ERRORS:		TOLERANCE:
ENV:	4062.657	ENV: .699
COMPX:	1612.794	COMPX: .699

MODEL: LOC = -5970.29 - 5210.08 ENV + 6802.18 COMPX

in the data is explained by the complexity variable. Also, the model specification agrees with the a priori logic, the sign for COMPX is positive indicating that as the complexity of the function the program runs increases, the number of lines of code are increased. (See the plot in Figure 5 based on the Fortran data base.)

In summary, only the complexity variable (COMPX) is significant as it was in the Assembly data base. However, when comparing the models containing COMPX for the Assembly and Fortran data bases, it can be seen that the regression statistics for COMPX in the Fortran data base are significantly higher than in the Assembly data

TABLE XXII

Results of Regression Models for QSPEC, REL, ENV, and COMPX

QSPEC:		
F VALUE:	.073	$F_{table}(.95; 1,19) = 3.52$
R SQUARE:	.0038	
ADJ R SQUARE:	-.0486	$t_{table}(.975; 19) = 2.093$
t_{calc}:		
QSPEC:	.270	
REL:		
F VALUE:	.039	$F_{table}(.95; 1,19) = 3.52$
R SQUARE:	.0020	
ADJ R SQUARE:	-.0505	$t_{table}(.975; 19) = 2.093$
t_{calc}:		
REL:	.197	
ENV:		
F VALUE:	.252	$F_{table}(.95; 1,52) = 4.03$
R SQUARE:	.0047	
ADJ R SQUARE:	-.0140	$t_{table}(.975; 52) = 2.01$
t_{calc}:		
ENV:	.502	
COMPX:		
F VALUE:	17.449	$F_{table}(.95; 1,52) = 4.03$
R SQUARE:	.2512	
ADJ R SQUARE:	.2368	$t_{table}(.975; 52) = 2.01$
t_{calc}:		
COMPX:	4.177	STANDARD ERROR:
ROOT MSE:	9756.207	COMPX: 1356.814

MODEL: LOC = - 8657.43 + 5667.605 COMPX

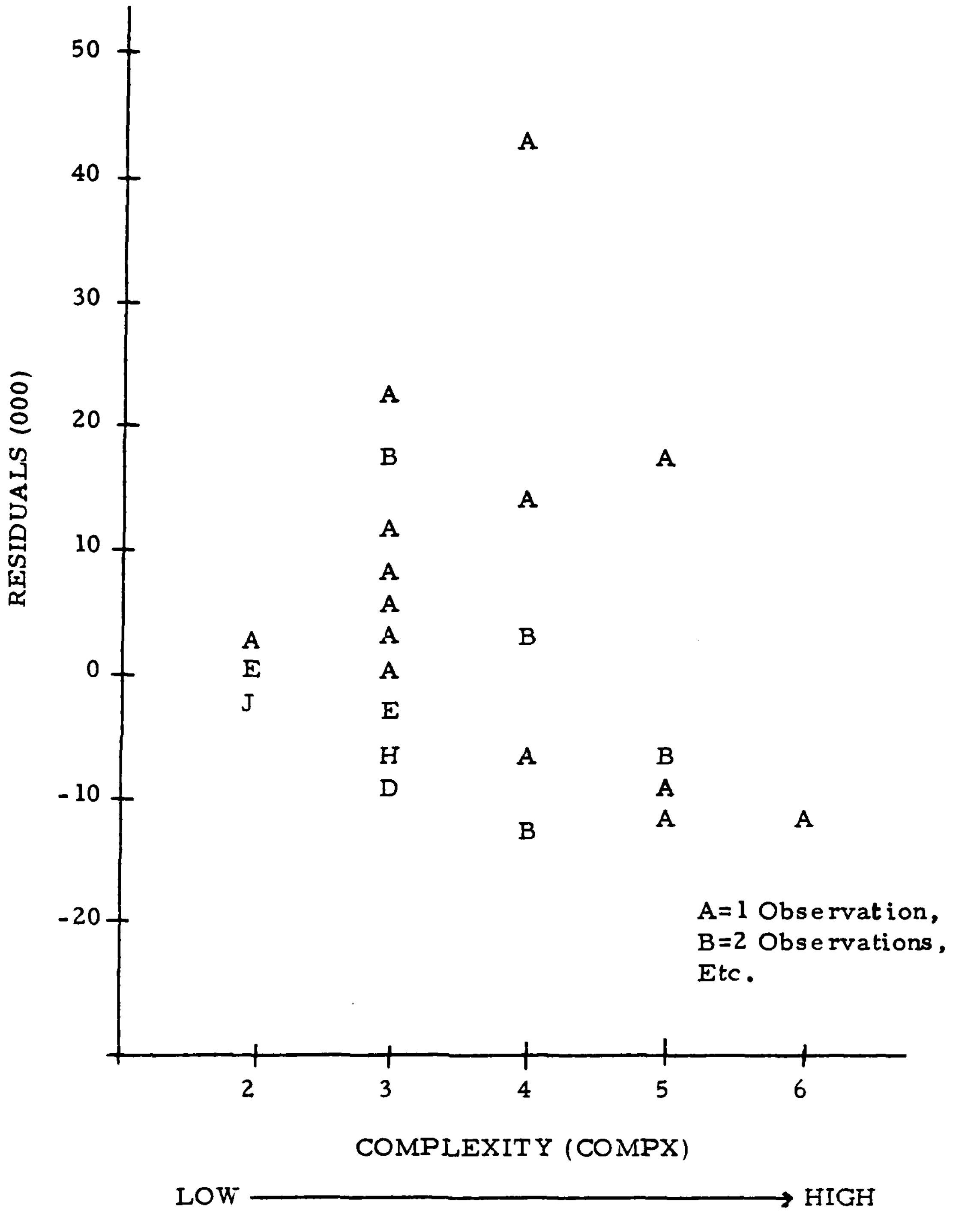


Figure 4. Net Scatter Diagram for Complexity Residuals Versus Complexity

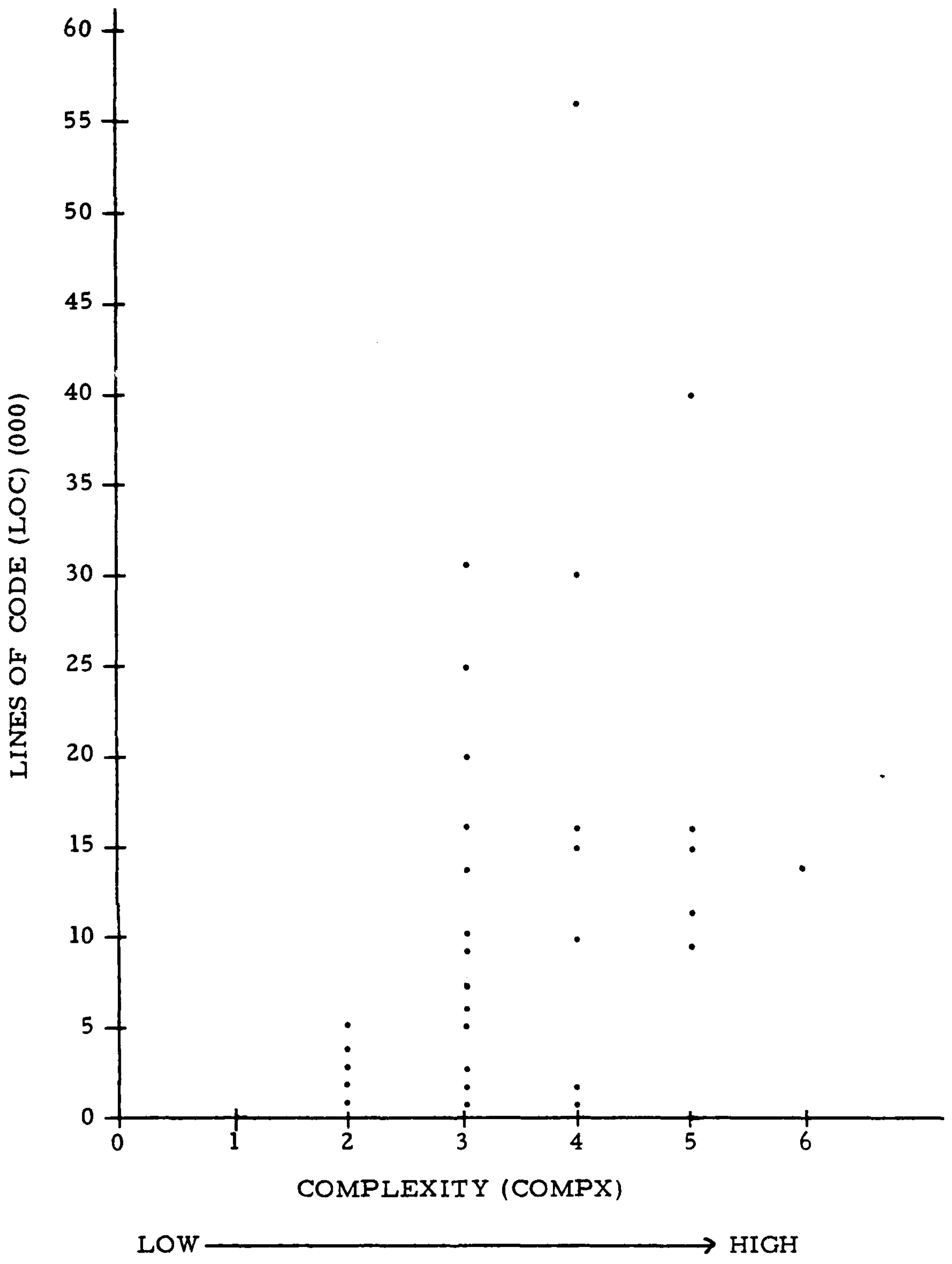


Figure 5. Lines of Code Versus Complexity

base. (See Table XXIII.) This suggests that either the complexity of the function the program must run is more significant when Fortran is used or that the larger number of data points in the Fortran data base contributes to the difference. Lastly, in both the Fortran and Assembly data bases the REL and ENV variables are insignificant. However, QSPEC is insignificant in the Fortran data base and still needs to be tested in the Assembly data base.

TABLE XXIII

Comparison of Results for the Assembly and Fortran Models for ENV/COMPX and COMPX

ASSEMBLY		FORTRAN	
COMPX:		COMPX:	
F VALUE:	6.416	F VALUE:	17.449
R SQUARE:	.1864	R SQUARE:	.2512
t_{calc}:		t_{calc}:	
COMPX:	2.533	COMPX:	4.177
ROOT MSE:	7829.229	ROOT MSE:	9756.207
STANDARD ERROR:		STANDARD ERROR:	
COMPX:	1693.755	COMPX:	1356.814
COMPX/ENV:		COMPX/ENV:	
F VALUE:	3.112	F VALUE:	9.655
R SQUARE	.1874	R SQUARE:	.2746
t_{calc}:		t_{calc}:	
COMPX:	2.369	COMPX:	4.218
ENV:	.175	ENV:	-1.282
ROOT MSE:	7968.37	ROOT MSE:	9696.29
STANDARD ERROR:		STANDARD ERROR:	
COMPX:	1863.292	COMPX:	1612.794
ENV:	4625.868	ENV:	4062.657

V. Conclusions and Recommendations

Conclusions

This thesis has in part answered the two primary research questions posed in chapter one. The first question asked--Given several software sizing data bases, what are some of the possible statistically significant software size drivers? In answering this first question, the six different data bases used in this thesis all are described by possible statistically significant size drivers. Each of these data bases were described in detail in chapter three. However, the analysis in chapter four actually revealed the statistically significant size drivers for each data base. Chapter four also answered the second primary research question--Given the possible software size drivers for each data base, can a statistically significant multiple regression model be developed to predict software size for each data base? (See Table XXIV for a summary of the results of the regression analyses.)

For the BMO data base there were no statistically significant variables or regression models. However, it should be kept in mind that this data base contains only seven data points. The two most "promising" independent variables were the number of interfaces the program has (INTF) and what type of environment (ground or airborne) the software will operate in.

TABLE XXIV

Summary of Results

Data Bases	BMO	ESD	AD: Ground Programs	AD: Airborne Programs
Statistically Significant Variables	None	None	DM (Development Months) COMPX (Complexity)	DM FUNC REL (Reliability)

Statistically Significant Regression Models	None	LOC = -16977.4 + 28268.4 QSPEC - 393.079 EXP
		
		QSPEC = quality of specification
		
		EXP = experience of programmers
		
		R^2 = .3591
			None	LOC = 9757.322 - 8647.28 FUNC + 4262.762 REL
				
				R^2 = .6583

TABLE XXIV

Summary of Results (Continued)

Data Bases	Language: Assembly	Language: Fortran
Statistically Significant Variables	COMPX	COMPX
Statistically Significant Regression Models	$LOC = -6934.64 + 810.2997\ ENV + 4414.196\ COMPX$ Note: ENV fails t test at $\alpha = .05$ and $\alpha = .10$ $R^2 = .1874$ Adjusted $R^2 = .1272$	$LOC = -5970.29 - 5210.08\ ENV + 6802.18\ COMPX$ Note: ENV fails t test at $\alpha = .05$ and $\alpha = .10$ $R^2 = .2746$ Adjusted $R^2 = .2462$

The results of the regression analysis on the ESD data base revealed that both the quality of the specification (QSPEC) and the experience of the programmers (EXP) proved statistically significant when in combination in a regression model. The coefficient of correlation (R^2) was .3591 which means the model explains slightly over a third of the variation in the data when regressed against the number of lines of code (LOC). Unfortunately, the high standard error of the estimate for the model and the high standard error terms for the two independent variables does not allow the model to be useful for predicting LOC. Finally, none of the independent variables prove statistically significant when analyzed in individual regression models. However, the quality of specification (QSPEC) and the experience level of the programmers (EXP) were the two highest in terms R^2 values and t test results.

As described in chapter three, the AD data base was divided into two subgroups--ground programs and airborne programs. This was done in order to better separate the different types of software programs. These two sets of programs are designed for three different functions: missile, range, or munition. The results of ground subgroup indicated that the relative complexity of the function (COMPX) and the number of development months (DM) were both statistically significant in individual regression models. However, when combined into a single model, neither the model nor the two independent variables (COMPX and DM) proved statistically significant. Also, other combinations of the variables revealed no statistically significant regression models.

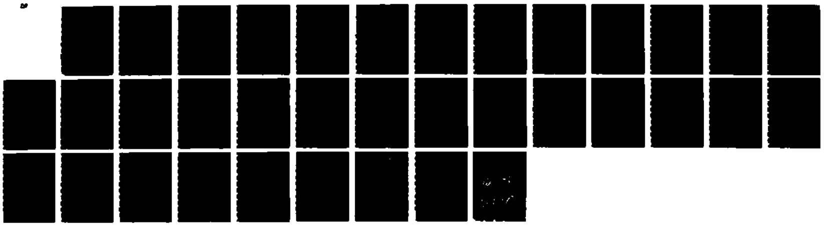
The regression analysis on the airborne subgroup of the AD data base revealed that three of the independent variables were statistically significant when run in individual models. These variables were development months (DM), the function the program is designed for (FUNC), and the reliability required in the program (REL). It was discovered, however, that when DM is used in combinations with the other independent variables, multicollinearity always distorts the regression statistics. This may be due to the small number of data points for DM. Even so, DM probably is not a good cost driver of lines of code anyway. On the other hand, the combination of FUNC and REL, produces a good regression model. First, there is no multicollinearity or influential outliers. Second, the coefficient of correlation (R^2) is about 66 percent. Next, the model passes the F test at the 95 percent level of confidence and the independent variables their t tests at the ($1 - \alpha/2 =$) .975 level of confidence. Finally, the model specification seems logical--function inversely related to lines of code and reliability directly related to lines of code. On the other hand, as shown in Table XIV, the standard error terms are too high for very accurate predictions of lines of code.

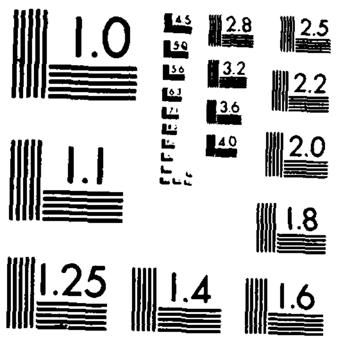
Lastly, as described in chapter three, the final two data bases were a combination of the four major data bases used in this thesis. These two data bases contained data split into the two most numerous computer languages found in the four major data bases: Assembly and Fortran. This grouping was done in order to see if programs catagorized by the language they are written in is a way to develop a good regression model to predict lines of code. Also, each of these data bases contain only four functional variables in addition, of course, to the number of lines of code. They are quality of specification (QSPEC), the environment (ENV), reliability (REL), and complexity (COMPX). The regression analysis on the Assembly data base revealed that the only statistically significant independent variable is complexity. No combination of the variables uncovered any useful model to predict lines of code.

The Fortran data base analysis produced almost exactly the same results as the Assembly data base analysis. Complexity is the only statistically significant independent variable both in a model by itself and when in combination with the other variables. However, as Table XXIII in chapter four shows, the complexity variable models for the Fortran data base are somewhat better than those for the Assembly data base.

Overall, the regression analysis on the six data bases indicated that complexity (COMPX), development months (DM), reliability

AD-A174 335   DEVELOPING SOFTWARE SIZE ESTIMATING RELATIONSHIPS BASED   2/2
ON FUNCTIONAL DES (U) AIR FORCE INST OF TECH
WRIGHT-PATTERSON AFB OH SCHOOL OF SYST   M J WHETSTONE
UNCLASSIFIED   SEP 86 AFIT/GSM/LSY/86S-24   F/G 9/2   NL




1.0
4.5
5.0
5.6
6.3
7.1
2.8
3.2
3.6
4.0
2.5
2.2
2.0
1.1
1.8
1.25
1.4
1.6

(REL), and function (FUNC) are the statistically significant independent variables when each of the variables were run in separate models. Of these, complexity, is the most frequent. Five of the six data bases contain the variable of complexity and in three of the five data bases (60%) it is statistically significant. (See Table XXIV.) Also, development months/hours is a variable in four of the six data bases and was statistically significant in two of the four (50%) of those data bases.

The best multivariate model was found for the airborne data base for Armament Division. The independent variables of reliability (REL) and function (FUNC) are in the model. The coefficient of correlation is a respectable 66 percent. (See chapter four and Tables XIV and XXIV.)

Unfortunately, the statistically significant regression models found for each data base are not very useful for predicting the number of lines of code. This is mainly due to low R^2 values which indicate there are other variables related to lines of codes, but which are not known. If these unknown variables could be included in the models found in this research, then possibly the models will prove to be better. Also, not enough data points for each data base could cause low R^2 values. Lastly, large standard error of the estimates for the models and large standard error terms for the independent variables produce such wide ranges of values for the true progression parameters, that prediction is impossible on a precise level.

Finally, the research subquestion--How generalizable are the regression equations developed?--was only partially answered. The two computer language data bases were only a small attempt to generalize a regression model. This was accomplished by combining common valuables from the four major data bases and separating the programs by the two most prevalent languages. Complexity was found to be the only statistically significant variable for both data bases. A model containing two variables (COMPX and ENV) was the only statistically significant model found for each data base. However, they both have low R^2 values. (See Table XXIV.)

Recommendations

The results of this thesis have generated a number of suggestions and ideas for further research. They are presented below in the order in which the author feels they should be accomplished, although this may not be possible due to other constraints.

As a general recommendation, more data points and independent variables should be added to each data base before any further research is accomplished. These actions will make the statistical results more substantial and meaningful.

Recommendation 1. The method used in this research to quantify qualitative type variables (for example, language) is very simple.

The use of indicator variables is a much better way to quantify qualitative type variables. The reader should consult reference seven or similar books on regression for more on the use of indicator variables. The results of using indicator variables may produce significantly different conclusions. In the author's opinion, this recommendation is just as important as the first.

Recommendation 2. Even though the net scatter diagrams and model specifications probably imply that the data is linear, this may not be the case. Therefore, using transformations on the data such as squaring variables or using the log function may produce better models. Of course, justification must be given for performing any type of transformation.

Recommendation 3. Another approach to estimating the number of lines of code may be to find regression equations for more specific data bases. In other words, try to find a regression model to predict the number of lines of code for Air Force avionic systems or even avionic systems for DoD in general. Looking at typical work breakdown structures for aircraft, missile, or space systems should provide other ideas.

Appendix A: Regression Statistics Used

This appendix describes the ANOVA table and the other regression statistics used in this thesis. For a more detailed explanation the reader should consult reference 7.

The Analysis of Variance Table

The ANOVA table shows the statistical relationships between the dependent variable and the independent variable(s) in a regression model (see Figure 6). "The analysis of variance approach is based on the partitioning of sums of squares and degrees of freedom associated with the response variable 'Y'" (7:84). In this thesis the response variable is source lines of code (LOC).

There is variation in all statistical data. If all observations (each data point) Y_i are the same, $Y_i = \overline{Y}$ (the mean of the data points in each data base), and there would be no statistical problems (7:84). The variation of the Y_i is measured in terms of the deviation $Y_i - \overline{Y}$. The measure of total variation, denoted by C TOTAL on the SAS printouts, is the sum of the squared deviations:

$$\text{C TOTAL} = \Sigma\,(Y_i - \overline{Y})^2 \tag{4}$$

"If C TOTAL = 0, all observations are the same. The greater is C TOTAL, the greater is the variation among the Y_i observations" (7:85).

SAS
ANALYSIS OF VARIANCE

Source	DF	Sum of Squares	Mean Square	F Value	Prob > F
MODEL	1	43	43	3.166	0.1089
ERROR	9	90	10		
C TOTAL	10	133			
ROOT MSE		30	R-SQUARE	0.2602	
			ADJ R-SQ	0.1780	

PARAMETER ESTIMATES

Variable	DF	Parameter Estimate	Standard Error	T for H0: Parameter = 0	Prob > T
INTERCEP	1	-135.6	88.53	- .847	0.4189
COMPX	1	80.135	45.73	1.779	0.1089

Figure 6. An Example of an ANOVA Table

The smaller the C TOTAL, the less the variation in the data base; therefore, a small C TOTAL is desirable.

"The variation reflecting the uncertainty in the data is that of the Y observations around the regression line $Y_i - \hat{Y}_i$" (7:86). The $\hat{Y}_i$'s are the predicted values of each observation using the regression equation developed. The measure of variation in the data with the regression model, denoted by ERROR on the SAS printouts, is the sum of the squared deviations:

$$\text{ERROR} = \Sigma (Y_i - \hat{Y}_i)^2 \tag{5}$$

"If ERROR = 0, all observations fall on the fitted regression line. The larger ERROR, the greater is the variation of the Y observations around the regression line" (7:86). Therefore, a small ERROR value is desirable.

The difference between the two sums of squares, C TOTAL-ERROR, is another sum of squares denoted by MODEL on the SAS printouts:

$$\text{MODEL} = \Sigma (\hat{Y}_i - \overline{Y})^2 \tag{6}$$

> The deviations are $\hat{Y}_i - \overline{Y}$. Each deviation is simply the difference between the fitted value on the regression line and the mean of the observations. If the regression line is horizontal so that $\hat{Y}_i - \overline{Y} = 0$, then MODEL = 0. Otherwise, MODEL is positive [7:86].

Therefore, as can be inferred from the above discussion, MODEL was one of the key statistics; and the larger MODEL in relation to C TOTAL, the better the regression model. "MODEL may be considered a measure of the variability of the Y_i's associated with the regression line. The larger MODEL is in relation to C TOTAL, the greater is the effect of the regression relation in accounting for the total variation in the Y_i observations" (7:86).

In summary, C TOTAL = MODEL + ERROR. The total deviation $(Y_i - \overline{Y})$ equals the deviation of the fitted regression value around the mean $(\hat{Y}_i - \overline{Y})$ plus the deviation around the regression line $(Y_i - \hat{Y}_i)$.

The next section of the ANOVA table is the degrees of freedom associated with each sum of square. The SAS printout lists each degree of freedom for each of the respective sum of squares.

The last section of the ANOVA table lists the mean squares. The mean square is simply the sum of square component divided by its respective degree of freedom.

Next, the other important SAS statistics used in the research are described.

The R^2 Value

The most common statistic is the R^2 value. This is one measure of the degree of linear association between Y (=LOC) and the independent variables used in each data base.

> The coefficient of multiple determination, denoted by R^2 (R-SQUARE on the SAS printouts), is defined as follows:

$$R^2 = \text{MODEL/C TOTAL} = 1 - \text{ERROR/C TOTAL} \tag{7}$$

It measures the proportionate reduction of total variation in Y associated with the use of the set of X variables $X_1, \ldots, X_{p-1}$. We have

$$0 \leqq R^2 \leqq 1 \tag{8}$$

R^2 assumes the value 0 when all parameter estimates equal 0. R^2 takes on the value of 1 when all observations fall directly on the fitted response surface, i.e., when $Y_i = \hat{Y}_i$ for all i [7:241].

However, two important points should be kept in mind about the R^2 value.

First, a large R^2 does not necessarily imply that the fitted model is a useful one. For instance, observations may have been taken only at a few levels of the independent variables. Despite a high R^2, the fitted model may not be useful because some of the predictions may require extrapolations outside the region of observations. Also, even if R^2 is high, the standard error of the estimate may still be too large for inferences to be useful in a case where high precision is required.

Second, adding more independent variables to the model can only increase R^2 and never reduce it, because ERROR can never become larger with more independent variables and C TOTAL is always the same for a given set of responses. Since R^2 often can be made large by including a large number of independent variables, it is sometimes suggested that a modified measure be used which recognizes the number of independent variables in the model. This adjusted coefficient of multiple determination, denoted by ADJ R-SQ on the SAS printouts, is defined:

$$R^2 = 1 - (n-1/n-p)*(\text{ERROR/C TOTAL}) \tag{9}$$

[7:241].

The F Statistic

One of the most important statistics given by a regression program is the F statistic. The F statistic is the test statistic for the analysis of variance approach. (It is denoted as F VALUE on the SAS printouts.) The F statistic tests whether there is a regression relation between the dependent variable Y and the set of X variables. In other words, to choose the null hypothesis, H_o, that all the regression parameters equal zero and that there is no regression relation; or the alternative hypothesis, H_a, that not all the regression parameters equal zero and that there is a regression relation. (See Table XXV, Equation (10).

The Standard Error

The standard error of the estimate (ROOT MSE on the SAS printouts) is also a very important statistic.

> The standard error of the estimate is a measure of the reliability of the regression prediction. It is a measure of dispersion of observed values away from the regression line. Therefore, when a prediction is made, the standard error of the estimate may be used to estimate the confidence interval around the predicted value [3:253].

The other standard error statistics are those for the parameter estimates of the regression model. These standard error terms are used to measure the confidence limits for the true population parameters of the regression model.

TABLE XXV

Regression Test Statistics

<u>F test</u>:

$$F_{calc} = \text{MS MODEL} / \text{MS ERROR} \qquad (10)$$

Decision rule to control Type I error at α:

If $F_{calc} \leq F_{table}$ $(1 - \alpha; p - 1, n - p)$, conclude H_o

If $F_{calc} > F_{table}$ $(1 - \alpha; p - 1, n - p)$, conclude H_a

where:

p = number of parameters in regression model
n = number of observations

<u>Partial F test</u>:

$$\text{Partial } F_{calc} = \frac{\dfrac{\text{SS ERROR(R)} - \text{SS ERROR(F)}}{\text{DF(F)} - \text{DF(F)}}}{\dfrac{\text{SS ERROR(F)}}{\text{DF(F)}}} \qquad (11)$$

where:

SS ERROR(R) = sum of squares error reduced model
SS ERROR(F) = sum of squares error full model
DF(R) = degrees of freedom reduced model
DF(F) = degrees of freedom full model

TABLE XXV

Regression Test Statistics
(continued)

Decision rule for partial F test:

If partial $F_{calc} \leq F_{table}$ (1 - α; DF(R) - DF(F), DF(F)), conclude H_o

If partial $F_{calc} > F_{table}$ (1 - α; DF(R) - DF(F), DF(F)), conclude H_a

t test (level of significance at α):

If $|t_{calc}| \leq t\ (1 - \alpha/2;\ m - 2)$, conclude H_o: $\beta = 0$

If $|t_{calc}| > t\ (1 - \alpha/2;\ m - 2)$, conclude H_a: $\beta \neq 0$

Multicollinearity

Unfortunately, as in many nonexperimental situations, the independent variables could be correlated among themselves and with other variables that are related to the independent variable LOC, but were not included in the models because they were unknown. This correlation among the independent variables, or multicollinearity, causes problems and has to be considered. If multicollinearity does not exist in models developed, then

> . . . if X_1 and X_2 are uncorrelated, adding X_2 to the regression model does not change the regression coefficient for X_1; correspondingly, adding X_1 to the regression model does not change the regression coefficient for X_2 [7:274].

When multicollinearity does exist, the following four main problems could arise:

> First, when independent variables are correlated, the regression coefficient of any independent variable depends on which other independent variables are included in the model and which ones are left out. Thus, a regression coefficient does not reflect any inherent effect of the particular independent variable on the dependent variable but only a marginal or partial effect, given whatever other correlated independent variables are included in the model.
>
> Second, when independent variables are correlated, there is no unique sum of squares which can be ascribed to an independent variable as reflecting its effect in reducing the total variation in Y. The reduction in the total variation ascribed to an independent variable must be viewed in the context of the other independent variables included in the model, whenever the independent variables are correlated.

> Third, the estimated regression coefficients individually may not be statistically significant even though a definite statistical relation exists between the dependent variable and the set of independent variables.
>
> Last and most important is the common interpretation of regression coefficients as measuring the change in the expected value of the dependent variable when the corresponding independent variable is increased by one unit while all other independent variables are held constant is not fully applicable when multicollinearity exists [7:277,383,385].

Multicollinearity, unfortunately, also causes difficulties in statistical tests of the regression coefficients.

> A not infrequent abuse in the analysis of multiple regression models is to examine the t statistic for each regression coefficient in turn to decide whether or not all the population parameters, β_k for k = 1, . . ., p-1, equal zero [7:278].

For this reason the "partial F test" is used to test whether or not the individual β_k's were zero instead of the t test when multicollinearity exists for the multivariable case. (See Table XXV, Equation (11).) An important point of the partial F test is that when the numerator (degrees of freedom) equals one, then the partial F test is equal to the square of the t test statistic.

However, in cases of simple regression models (one independent variable), the two-sided t test was used. (See Table XXV.) (t_{calc} is identified under T for HO on the SAS printouts.)

Despite all the negative aspects of multicollinearity, there is one bright spot.

> The fact that some or all independent variables are correlated among themselves does not, in general, inhibit the ability to obtain a good fit nor does it tend to affect inferences about mean responses or predictions of new observations, provided these inferences are made within the region of observations [7:384].

The SAS program also gives multicollinearity statistics to help determine if one of the regressors in the model is nearly a linear combination of other regressors in the model. The first of these statistics are the variance inflation factors (VIFs).

> These factors measure how much the variances of the estimated regression coefficients are inflated as compared to when the independent variables are not linearly related. . . . A maximum VIF in excess of 10 is often taken as an indication that multicollinearity may be unduly influencing the least squares estimates. . . . A limitation of variance inflation factors for detecting multicollinearities is that they cannot distinguish between several simultaneous multicollinearities [7:391-393].

A second statistic to measure multicollinearity is the tolerance. The tolerance value is simple 1/VIF for each regressor in the model. The tolerance values will fall between:

$$0 \leq \text{TOLERANCE} \leq 1$$

A tolerance factor close to one implies independence. A rule of thumb states that if the tolerance factor is less than .1 then multicollinearity probably exists. However, as with the VIFs, the tolerance values will measure the multicollinearity among the independent variables, but cannot determine which variables are being affected.

Outliers

A second important condition that must be considered for each data base is the identification of outlying observations.

> Outlying observations may involve large residuals and often have dramatic effects on the fitted least squares regression function. An observation may be outlying or extreme with respect to its Y value, its X value(s), or both. In the scatter plot in Figure 7, observation 1 is outlying with respect to its Y value. Note that this point falls far outside the scatter, although its X value is near the middle of the range of the observations on the independent variable. Observations 2 and 3 are outlying with respect to their X values since they have much larger X values than those for the other observations; observation 3 is also outlying with respect to its Y value.
>
> Not all outlying observations have strong influence on the fitted regression function. Observation 1 may not be too influential because there are a number of other observations that have similar X values, which will keep the fitted regression function from being displaced too far by the outlying observation. Likewise, observation 2 may not be too influential because its Y value is consistent with the regression relation displayed by the nonextreme observations. Observation 3, on the other hand, is likely to be very influential in affecting the fit of the regression function because it is outlying with regard to its X value, and its Y value is not consistent with the regression relation for the other observations [7:400-401].

To determine whether an X value is an outlier the leverage value (h) is computed.

> It indicates whether or not the X values for the i^{th} observation are outlying, because it can be shown that h is a measure of the distance between the X values for the i^{th} observation and the means of the X values of all n observations. Thus, a large leverage value h indicates that the i^{th} observation is distant from the center of the X observations [7:402].

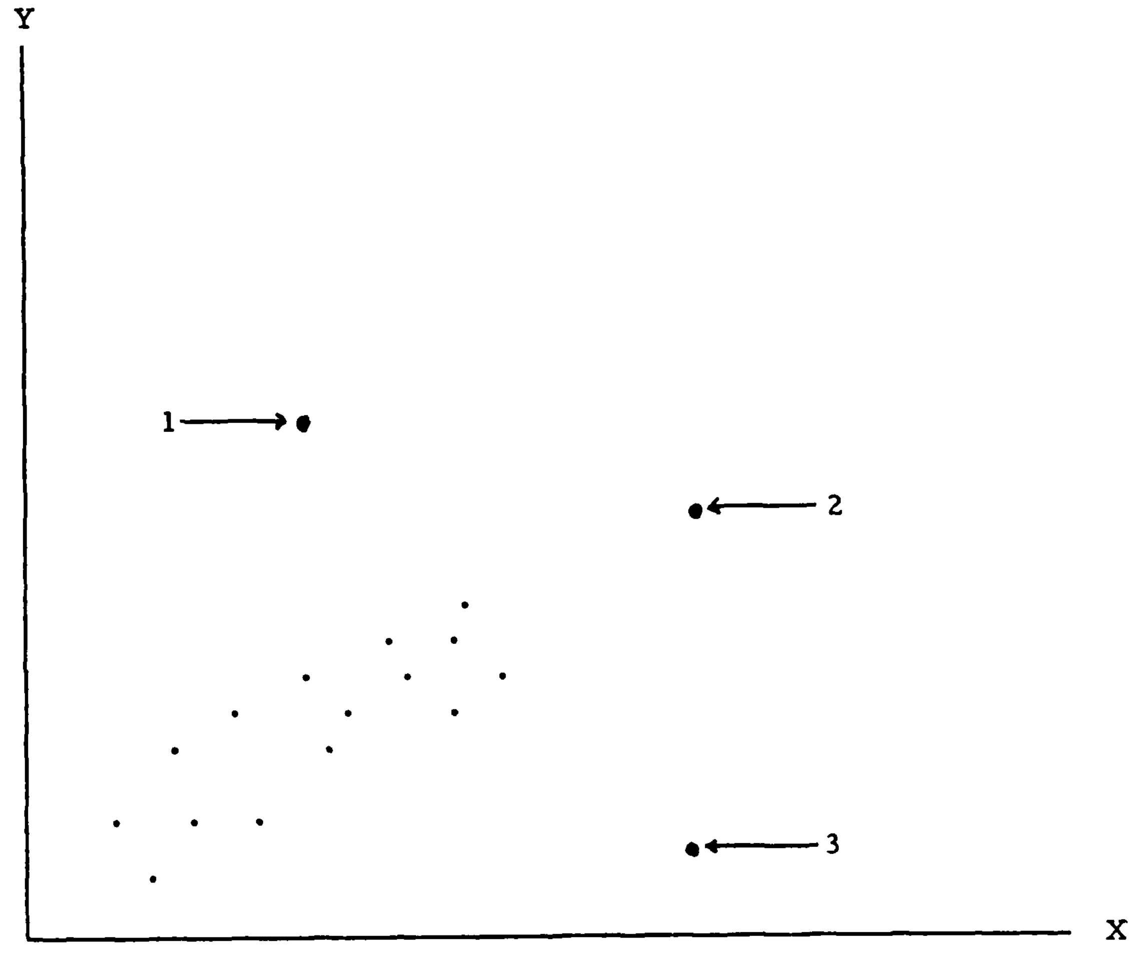


Figure 7. Outlying Observations

The rule of thumb states: "Leverage values, h, greater than or equal to two times the number of parameters, p, divided by the number of observations, n, ($h \geq 2p/n$) indicate outlying observations with regard to the X values" (7:403).

To determine whether a Y value is an outlier the studentized deleted residual is computed.

> To identify outlying Y observations, examine the studentized deleted residuals for large absolute values and use the appropriate t distribution ($t_1 - \alpha$, $n - 1 - p$) to ascertain how far in the tails such outlying values fall [7:406].

Any studentized deleted residual (RRESID on the SAS printouts) greater than the appropriate t distribution value is considered an outlier with respect to Y.

Cook's *D*. After identifying outlying observations with respect to their X and/or Y values, the next step is to determine if they were influential in affecting the fit of the regression function. If the outliers are influential and not corrected, then the fitted regression function will be distorted. An overall measure of the impact of the ith observation on the estimated regression coefficients is Cook's distance measure (COOK'S D on the SAS printouts).

> Cook's distance measure D_i may be viewed as reflecting in the aggregate the differences between the fitted values for each observation when all n observations are used in the data base and the fitted values when the ith observation is deleted [7:409].

To determine if the outlier is influential use the following rule:

> While Cook's D does not follow the F distribution, it has been found useful to relate the value D_i to the corresponding F distribution according to $F(p, n-p)$ and ascertain the percentile value. If the percentile value is less than about 10 or 20 percent, the ith observation has little apparent influence on the fitted regression function. If, on the other hand, the percentile value is near 50 percent or more, the distance should be considered large, implying that the ith observation has a substantial influence on the fit of the regression function [7:408].

Model Specification

The last major area of concern with the regression models developed is the model specification; i.e., is the linear regression model appropriate for the data being analyzed?

> A plot of the residuals, e_i's ($e_i = Y_i - \hat{Y}_i$), against the independent variables is not only helpful to study whether a linear regression function is appropriate but also to examine whether the variance of the error terms is constant [7:113].

If the model is correctly specified, the residual plots should show a random pattern.

The major implication of having a correctly specified regression model is that it will correctly show how the dependent variable, (LOC), changes in response to a one unit change in the independent variables (the other independent variables remaining constant and in the absence of multicollinearity among the independent variables). In other words, what are the correct signs (plus or minus) for each of the independent variables? Does the model derived predict the correct signs for each of the independent variables that the a priori suggested?

The conditi^n of constant or equal error variances over all observations is called homoscedasticity. This is in contrast to non-constant variance or heteroscedasticity. Having homoscedasticity is very important because the estimators of the population regression coefficients ($\beta_0, \beta_1, \beta_2, \ldots$) obtained by ordinary least squares procedures are unbiased and consistent and are minimum variance unbiased estimators (7:170). With heteroscedasticity the estimators are still unbiased and consistent, but they are no longer minimum variance unbiased estimators (7:170). "Heteroscedasticity is inherent when the response in regression analysis follows a distribution in which the variance is functionally related to the mean" (7:170).

Appendix B: Thesis Data Bases

This appendix contains the six data bases used in this thesis.

I. Ballistic Missile Office

OBS	LOC	ENV	LANG	INTF	INPT	OUTPT	EXP	DM
1	43000	1	3	14	10	37	24	36
2	8875	2	3	10	6	10	8	30
3	32000	1	3	21	--	--	54	30
4	112000	1	5	9	42	44	1	26
5	6400	1	5	7	24	25	1	26
6	13010	2	5	16	31	28	8	31
7	16000	1	4.5	18	139	131	16	29

II. Electronic Systems Division

OBS	LOC	ENV	DHRS	LANG	REL	COMPX	EXP	QSPEC
1	26200	1	44868	4	5	5	1.4	2
2	15987	2	9433	1.5	4	4	22.4	1
3	56021	1	28320	1.8	5	4	25.6	2
4	21296	1	36640	7	4	4	13	2
5	63944	1	78020	2.1	4	5	50.4	3
6	47525	1	11000	3	4	5	54	1
7	9000	1	8976	1	4	5	12	1
8	15000	2	185328	1.4	5	5	64.8	2
9	15100	2	64247	1.4	5	5	64.8	2
10	12000	2	27456	1.4	5	5	64.8	2
11	14900	2	14664	1.2	5	6	64.8	2
12	18300	1	48184	3	5	5	50.4	2
13	10800	1	30704	3	5	5	50.4	2
14	10700	1	134824	3	5	5	50.4	2
15	16700	1	205504	3	5	4	50.4	2
16	10500	1	25384	3	5	4	50.4	2
17	6539	1	16568	9	5	5	50.4	2
18	47165	1	84968	9	5	5	50.4	2
19	14200	1	43320	3	5	5	50.4	2
20	26033	1	40280	3	3	3	50.4	2

II. Electronic Systems Division (Continued)

OBS	LOC	ENV	DHRS	LANG	REL	COMPX	EXP	QSPEC
21	24260	1	35112	10	3	3	50.4	2
22	628	1	304	1	3	4	50.4	2
23	29954	1	43320	3	3	4	50.4	2
24	9700	1	110504	3	5	3	50.4	2
25	5600	1	80104	3	5	3	50.4	2
26	32100	1	145920	3	5	5	50.4	2

III. Armament Division--Ground

OBS	LOC	DM	LANG	QSPEC	REL	FUNC	COMPX
1	80000	48	1.3	2	3	2	5
2	25000	48	1.3	2	3	2	3
3	10000	20	1.2	2	5	2	3
4	4000	22	1.2	2	4	2	3
5	3000	24	1	1	2	2	2
6	1400	36	8	3	4	3	4
7	15900	36	1	2	3	3	4
8	1200	36	1	3	3	3	3
9	25000	--	1.3	2	3	2	3
10	40000	--	1	2	3	2	5
11	5800	--	3	2	3	2	4
12	25000	--	8	2	4	2	4

IV. Armament Division--Airborne

OBS	LOC	DM	LANG	QSPEC	REL	FUNC	COMPX
1	3000	24	2.05	3	4	1	5
2	2100	18	3	2	1	1	1
3	2000	42	3	2	3	3	2
4	16000	36	3	2	4	1	4
5	30000	24	1.3	2	5	1	4
6	60000	79	3	2	5	1	6
7	1300	22	1.2	2	3	2	3
8	16000	24	1	2	5	1	5
9	2700	22	1	2	4	2	3
10	10000	--	3	2	3	2	1
11	9000	--	6.8	2	5	2	4
12	17000	--	1	2	4	1	4
13	29000	--	1	2	4	1	4

V. Language--Assembly

OBS	LOC	QSPEC	REL	ENV	COMPX
1	2100	2	1	2	2
2	2000	2	3	2	2
3	16000	2	4	2	4
4	5800	2	3	1	4
5	18300	2	5	1	5
6	10800	2	5	1	5
7	10700	2	5	1	5
8	16700	2	5	1	4
9	10500	2	5	1	4
10	14200	2	5	1	5
11	26033	2	3	1	3
12	29954	2	3	1	4
13	9700	2	5	1	3
14	5600	2	5	1	3
15	32100	2	5	1	5
16	7200	-	-	2	3
17	4056	-	-	1	4
18	2311	-	-	1	4
19	1461	-	-	1	4
20	2158	-	-	1	4

V. Language--Assembly (Continued)

OBS	LOC	QSPEC	REL	ENV	COMPX
21	1454	-	-	1	3
22	7887	-	-	1	3
23	6231	-	-	1	3
24	2241	-	-	1	3
25	1058	-	-	1	3
26	19768	-	-	1	3
27	4008	-	-	1	3
28	9995	-	-	1	3
29	5674	-	-	1	3
30	5841	-	-	1	3

VI. Language--Fortran

OBS	LOC	QSPEC	REL	ENV	COMPX
1	30000	2	5	2	4
2	25000	2	3	1	3
3	10000	2	5	1	3
4	10000	2	3	2	4
5	1300	2	3	2	3
6	4000	2	4	1	3
7	25000	2	3	1	3
8	40000	2	3	1	5
9	6000	2	3	1	3
10	3000	1	2	1	2
11	15900	2	3	1	4
12	1200	3	3	1	3
13	2700	2	4	2	3
14	15987	1	4	2	4
15	56021	2	5	1	4
16	9000	1	4	1	5
17	15000	2	5	2	5
18	15100	2	5	2	5
19	12000	2	5	2	5
20	14900	2	5	2	6

VI. Language--Fortran (Continued)

OBS	LOC	QSPEC	REL	ENV	COMPX
21	4800	2	3	2	3
22	2000	-	-	1	4
23	1550	-	-	1	4
24	723	-	-	1	3
25	669	-	-	1	3
26	1096	-	-	1	2
27	753	-	-	1	2
28	886	-	-	1	2
29	1298	-	-	1	2
30	578	-	-	1	2
31	1079	-	-	1	2
32	1043	-	-	1	2
33	953	-	-	1	2
34	3101	-	-	1	2
35	4847	-	-	1	2
36	3877	-	-	1	2
37	1221	-	-	1	2
38	1381	-	-	1	2
39	3163	-	-	1	2
40	1985	-	-	1	2

OBS	LOC	QSPEC	REL	ENV	COMPX
41	13979	-	-	1	3
42	2798	-	-	1	3
43	2644	-	-	1	3
44	16111	-	-	1	3
45	31748	-	-	1	3
46	20287	-	-	1	3
47	3679	-	-	1	3
48	4708	-	-	1	3
49	4052	-	-	1	3
50	5426	-	-	1	3
51	4729	-	-	1	3
52	2582	-	-	1	3
53	2442	-	-	1	3
54	9025	-	-	1	3

Bibliography

1. Ayers, Everett E. and Kenneth B. Tom. Technical Report Software Sizing and Cost Estimation. Contract N00600-84-D-4045. ARINC Research Corporation, Annapolis MD, July 1985.

2. Boehm, Barry W. Software Engineering Economics. Englewood Cliffs NJ: Prentice Hall, Inc., 1981.

3. Clover, Vernon T. and Howard L. Balsley. Business Research Methods (Second Edition). Columbus OH: Grid Publishing, Inc., 1979.

4. Department of Defense. Defense System Software Development. DOD Standard 2167. Washington: Government Printing Office, 4 June 1985.

5. GJB Associates, Software Engineering and Analytical Services. "SSM Software Sizing Model for Better Software Cost Estimates." Sales Brochure. GJB Associates, Redwood City CA, n.d.

6. Graver, C.A. Cost Method Improvement Group (CMIG) AFSC Cost Research Road Map Final Report. Contract No. F04701-83-D-0103. Santa Barbara CA: Tecolote Research, Inc., September 1985.

7. Neter, John, et al. Applied Linear Regression Models. Richard D. Irwin, Inc., 1983.

8. RCA Corporation. PRICE SZ Reference Manual. RCA Corporation, Cherry Hill NJ, 1985.

9. Space Systems Cost Analysis Group, Software Subgroup. Software Sizing Data Base. The Aerospace Corporation, Los Angeles, 5 June 1985.

10. Steig, Jeff, Ron Stewe and John Ward. "Software Costing." Report for the Seminar in Cost Analysis. School of Systems and Logistics, Air Force Institute of Technology, Summer, 1984.

11. The Aerospace Corporation, Resource Cost Analysis Office. Software Sizing Capability. The Aerospace Corporation, Los Angeles, May 1985.

12. The Analytic Sciences Corporation. Software Data Base Development Volume I, Data Base Design and Collection Methodology. Contract No. F33657-82-D-0253/0014. The Analytic Sciences Corporation, Reading MA, 25 June 1984.

13. United States General Accounting Office. Report to the Congress of the United States. "DoD Needs to Provide More Credible Weapon Systems Cost Estimates to the Congress." Report series GAO/NSIAD-84-70. Washington: Government Printing Office, 24 May 1984.

14. Wheaton, Marilee J. "Functional Software Sizing Methodology," Journal of Parametrics, Volume VI: 17-23, March 1986.

Vita

Captain Mark J. Whetstone was born on 14 May 1956 in Cincinnati, Ohio. He graduated from high school in Cincinnati, Ohio, in 1974 and attended the University of Cincinnati. He received a Bachelor of General Studies degree in 1978. Upon graduation, he received a commission in the USAF through the AFROTC program and was called to active duty in December 1978. Upon completion of Minuteman missile launch officer training he was assigned to the 490th Strategic Missile Squadron, 341st Missile Wing at Malmstrom AFB, Montana. He served in a number of positions including instructor crew commander. While there, he earned a Master of Business Administration degree in June 1982 from the University of Montana through the Minuteman Educational Program. After completing his missile tour of duty in June 1983, he was assigned as a program analyst to the Airlift and Trainer System Program Office, Aeronautical Systems Division (ASD), Wright-Patterson AFB, Ohio. After completing Squadron Officers School in residence, he was assigned to the ASD Comptroller's staff as a program analyst in December 1984, until entering the School of Systems and Logistics, Air Force Institute of Technology, in May 1985.

Permanent address: 1091 Alcliff Lane
Cincinnati, Ohio 45238

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION
UNCLASSIFIED

1b. RESTRICTIVE MARKINGS
A174835

2a. SECURITY CLASSIFICATION AUTHORITY

2b. DECLASSIFICATION/DOWNGRADING SCHEDULE

3. DISTRIBUTION/AVAILABILITY OF REPORT
Approved for public release; distribution unlimited.

4. PERFORMING ORGANIZATION REPORT NUMBER(S)
AFIT/GSM/LSY/86S-24

5. MONITORING ORGANIZATION REPORT NUMBER(S)

6a. NAME OF PERFORMING ORGANIZATION
School of Systems and Logistics

6b. OFFICE SYMBOL *(If applicable)*
AFIT/LSY

7a. NAME OF MONITORING ORGANIZATION

6c. ADDRESS *(City, State and ZIP Code)*
Air Force Institute of Technology
Wright-Patterson AFB, Ohio 45433 -6583

7b. ADDRESS *(City, State and ZIP Code)*

8a. NAME OF FUNDING/SPONSORING ORGANIZATION

8b. OFFICE SYMBOL *(If applicable)*

9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER

8c. ADDRESS *(City, State and ZIP Code)*

10. SOURCE OF FUNDING NOS.

PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO.	WORK UNIT NO.

11. TITLE *(Include Security Classification)*
See Box 19

12. PERSONAL AUTHOR(S)
Mark J. Whetstone, M.B.A., Capt, USAF

13a. TYPE OF REPORT
MS Thesis

13b. TIME COVERED
FROM ________ TO ________

14. DATE OF REPORT *(Yr., Mo., Day)*
1986 September

15. PAGE COUNT
129

16. SUPPLEMENTARY NOTATION

17. COSATI CODES

FIELD	GROUP	SUB. GR.
14	01	
12	02	

18. SUBJECT TERMS *(Continue on reverse if necessary and identify by block number)*
Cost Analysis, Regression Analysis, Estimates, Costs, Computer

19. ABSTRACT *(Continue on reverse if necessary and identify by block number)*

Title: DEVELOPING SOFTWARE SIZE ESTIMATING RELATIONSHIPS
BASED ON FUNCTIONAL DESCRIPTIONS OF THE SOFTWARE

Thesis Advisor: Daniel V. Ferens
Instructor, Acquisition Management

Approved for public release: IAW AFR 190-1.
[illegible] 29 Sep 86
LYNN E. WOLAVER
Dean for Research and Professional Development
Air Force Institute of Technology (ATC)
Wright-Patterson AFB OH 45433

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT
UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT. ☐ DTIC USERS ☐

21. ABSTRACT SECURITY CLASSIFICATION
UNCLASSIFIED

22a. NAME OF RESPONSIBLE INDIVIDUAL
Daniel V. Ferens

22b. TELEPHONE NUMBER *(Include Area Code)*
(513) 255-4845

22c. OFFICE SYMBOL
AFIT/LSY

DD FORM 1473, 83 APR EDITION OF 1 JAN 73 IS OBSOLETE.

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE

This thesis researched the ability to develop regression models to predict the number of source lines of code (LOC) based on functional descriptions of the software. LOC, a major cost driver in currently available software cost estimating models, has been consistently underestimated, thus lowering not only the software cost estimate but also the total cost estimate of the weapon system. Six software sizing data bases containing various functional variables were used. The variables included complexity, reliability, experience level of programmers, etc. For each data base, regression analysis was performed to derive the optimal model to predict LOC. Of the five data bases containing complexity, it was statistically significant in three. The best developed model was for Armament Division's airborne computer programs. The correlation coefficient R^2 was .6583 for the two variables in the model. These were; (1) the system for which the program was developed and (2) the reliability needed in the program. The initial research has been accomplished, but more data and further research is needed.

UNCLASSIFIED

END

12-86

DTIC