A CONTRIBUTION TO COMPUTER AIDED DESIGN EVALUATION
UTILIZING FLEXIBLE ESTIMATION
AND MULTIPLE LINEAR REGRESSION ANALYSIS

by

BRUCE W.R. FORDE

A THESIS IN PARTIAL FULFILMENT OF THE
REQUIREMENTS FOR THE DEGREE OF
MASTER OF APPLIED SCIENCE

in

THE FACULTY OF GRADUATE STUDIES
(The Department of Civil Engineering)

We accept this thesis as conforming
to the required standard

September, 1984

© Bruce W.R. Forde
In presenting this thesis in partial fulfilment of the requirements for an advanced degree at the University of British Columbia, I agree that the Library shall make it freely available for reference and study. I further agree that permission for extensive copying of this thesis for scholarly purposes may be granted by the head of my department or by his or her representatives. It is understood that copying or publication of this thesis for financial gain shall not be allowed without my written permission.

Department of CIVIL ENGINEERING

The University of British Columbia
1956 Main Mall
Vancouver, Canada
V6T 1Y3

Date October 12, 1984
ABSTRACT

Improvements in design, achieved through a better understanding of fabrication costs, may cause an overall cost reduction for steel structures. A flexible estimation routine, able to satisfy the needs of the particular user, has been developed to provide the mechanism for design evaluation. The functions used by the estimation program are provided by a Multiple Linear Regression (MLR) analysis of data collected by an Information System (IS). The integration of fabrication control and analysis provided by this system permits its implementation in existing environments, and presents an important technological gain for the fabrication process. Traditional design approaches which have relied heavily on experience can now be evaluated and improved in terms of cost competitiveness, prior to or during fabrication, by the proposed MLR strategy.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABSTRACT</td>
<td>ii</td>
</tr>
<tr>
<td>TABLE OF CONTENTS</td>
<td>iii</td>
</tr>
<tr>
<td>ACKNOWLEDGEMENTS</td>
<td>iv</td>
</tr>
<tr>
<td>1. INTRODUCTION</td>
<td></td>
</tr>
<tr>
<td>1.1 Background Information</td>
<td>1</td>
</tr>
<tr>
<td>1.2 Research at U.B.C.</td>
<td>2</td>
</tr>
<tr>
<td>2. STEEL STRUCTURES - Design and Fabrication</td>
<td></td>
</tr>
<tr>
<td>2.1 Design Efficiency</td>
<td>5</td>
</tr>
<tr>
<td>2.2 Fabrication Costs</td>
<td>6</td>
</tr>
<tr>
<td>2.3 Erection Costs</td>
<td>9</td>
</tr>
<tr>
<td>2.4 Current Practice</td>
<td>10</td>
</tr>
<tr>
<td>2.5 Design/Fabrication Interaction</td>
<td>13</td>
</tr>
<tr>
<td>3. COMPUTER AIDED DESIGN EVALUATION</td>
<td></td>
</tr>
<tr>
<td>3.1 The Design Evaluation Program (ESTImate)</td>
<td>17</td>
</tr>
<tr>
<td>3.2 Multiple Linear Regression and the Information System</td>
<td></td>
</tr>
<tr>
<td>3.2.1 Motivation for MLR</td>
<td>21</td>
</tr>
<tr>
<td>3.2.2 The Information System</td>
<td>24</td>
</tr>
<tr>
<td>3.2.3 Example Use of MLR</td>
<td>26</td>
</tr>
<tr>
<td>3.2.4 MLR - Design and Theory</td>
<td>30</td>
</tr>
<tr>
<td>3.3 Applications of Program ESTImate</td>
<td></td>
</tr>
<tr>
<td>3.3.1 Design Optimization</td>
<td>34</td>
</tr>
<tr>
<td>3.3.2 Improvement of the Fabrication and Estimation Processes</td>
<td>36</td>
</tr>
<tr>
<td>3.3.3 Research and Education</td>
<td>37</td>
</tr>
<tr>
<td>4. SUMMARY</td>
<td></td>
</tr>
<tr>
<td>4.1 Conclusions</td>
<td>38</td>
</tr>
<tr>
<td>4.2 Extensions of this Research</td>
<td>39</td>
</tr>
<tr>
<td>REFERENCES</td>
<td>40</td>
</tr>
<tr>
<td>APPENDIX</td>
<td>42</td>
</tr>
</tbody>
</table>
ACKNOWLEDGEMENTS

The investigations and research contained in this thesis were made possible by a Natural Sciences and Engineering Research Council of Canada (NSERC) postgraduate scholarship.

Manfred Frank, Chief Engineer of the Western Bridge Division of CANRON INC., and Siegfried F. Stiemer, Associate Professor in the Department of Civil Engineering at the University of British Columbia, who was my supervisor, are due sincere thanks for their guidance and cooperation.

I am especially grateful to Elaine Lee, my friend, who provided me with motivation throughout my research.
1. INTRODUCTION

1.1 Background Information

Over the past few decades, the design and fabrication of structural steel in Canada and the United States has followed a procedure which was based on the experience of the design profession and the results of a few independent studies. Improvements in design were found by minimization of material weight by von Hoerner (1982) [18], and Schmit (1960) [16], by repetition of similar shapes by Moses and Goble (1970) [11], and by stimulation of design/fabrication interaction by Forde, Leung and Stiemer (1984) [7]. Until now, most of these efforts have fallen short of providing "minimum cost designs", simply because the costs incurred during fabrication have been only approximately known. The cause of this uncertainty can be attributed to the nature of the design and fabrication processes which make each project unique.

The problem facing fabricators is to identify where costs are incurred, to predict these costs for future jobs, and to control and monitor the process. Recording fabrication times and process costs, in the detail necessary for the estimation process, has been found to be too tedious and expensive to justify the information that it provides. For this reason, there has been a trend towards collecting information on a global perspective.

Recent developments in microcomputer technology offer a means of economical data storage and analysis. The addressable memory and storage capability of many systems easily satisfy the requirements of designers and fabricators. The simplifications, and improved efficiency, that are
potentially available from a computerized fabrication process, have provided the predominant motivation for research in this area.

1.2 Research at U.B.C.

An extensive research project has been started at the University of British Columbia in the areas of design evaluation and fabrication analysis. One goal of this project is to establish a mechanism which integrates the design and fabrication processes so that a better understanding of costs is obtained. To meet this goal, several smaller projects must be completed. Fabrication analysis can be divided into the collection, analysis, and feedback of fabrication data. Design evaluation requires the development of a flexible estimation program, structural analysis and code requirement modules, and improvement or optimization routines. Further projects that are required in order to assure integration of the whole include several modules that provide user friendly interface for interactive input, and compatibility for transfer of information between modules.

This thesis provides an overview of the total project, with extensive research in two areas:

(1) Estimation of fabrication times;

(2) Analysis of fabrication data.

Design evaluation in terms of cost can be done using the estimation program which is described in this thesis (ESTImate). This program systematically breaks projects down into basic elements, estimating the time and cost involved in performing a set of standard fabrication operations. Owing to the individualistic nature of each fabrication plant, the estimation programs used by various contractors are not the same. Since there exists no
universal standard, nor should there as this would exclude the potential for improvement, a flexible estimation program is essential. Program ESTImate has been developed by adopting a fourth generation approach to programming [13] which makes it exceptionally versatile for the construction of estimation equations.

Factors contributing to the requirement of flexible estimation are:

(1) The fabrication process is not a mass production, so the estimation routine must be able to accept a wide variety of projects.

(2) Fabrication plants are application oriented, with some being better equipped to do specific projects than others, hence one must allow for a variety of fabrication techniques.

(3) Design evaluation must be done in a detailed manner, as provided by an estimation program, so that improvement or optimization can be achieved.

The conversion of information which is gathered from a global perspective, to detailed information which is required for estimation purposes, can be done through the use of a comprehensive information system by Berry (1984) [2] coupled with a multiple linear regression program.

The implications arising from the use of MLR are:

(1) Information need only be collected at "checkstations" which are located at key locations throughout the plant.

(2) Integration of scheduling, controlling, and estimating is made possible through the use of an information system which collects this data at the global level.

(3) The fabrication plant can be evaluated in terms of performance and economy by analyzing the data collected by this information system over a period of time.
The multiple linear regression module that has been developed as part of this thesis is capable of analyzing data collected from a global perspective only. Establishment of intermediate checkstations will provide information which requires a more sophisticated analysis.
2. STEEL STRUCTURES

   Design and Fabrication

2.1 Design Efficiency

   The efficiency of a structural design can be defined as the extent to which it satisfies structural, architectural, and cost constraints. The role of the structural engineer in the design process has traditionally been to choose and design the most efficient structural system subject to the given constraints. This has been no easy task, since the constraints are linked so that there must be some tradeoff made in reaching the optimal design. Architectural requirements are generally defined by the owner as a given requirement of the project, so structural form is to be varied to provide the most economical design. Another important fact complicates the situation: in most cases, the designer will not have access to the necessary cost data; hence, the formulation of the design objective can not be in terms of a cost function, and the designer is unable to perform formal structural cost optimization.

   In the past, "experience" has been the key to efficient structural steel design. Successful engineers have established design guidelines which take advantage of information that has been acquired through interaction with fabricators. However, in an effort to maintain confidentiality of their direct costs, fabricators are likely to provide the designer with information of only a qualitative nature. The fabricators have a more quantitative look at the information since they are able to directly view the operation and can account for the costs as they are incurred. Nevertheless, sometimes even the fabricators are unsure of the exact costs associated with various structural forms, due to the variation of the type and quantity of work done for any one project.
The size and complexity of the projects that are undertaken by most structural steel fabricators vary significantly over time. It is this inherent variability of the structural steel fabrication process which poses the largest problem to the designer. In order to assign a cost to an arbitrary structural form, costs for each component can be calculated from a standard set of fabrication variables, which are in turn determined and updated through analysis of past projects. The fabrication process, although composed of physically simple operations, as a whole is extremely complicated; hence, associating costs with the fabrication of specific structural components is difficult. In fact the volume of work involved in the collection, organization, and analysis of data required to identify where costs are incurred has effectively eliminated this approach from conventional practice. To surpass this barrier, the designer must investigate the nature of the costs associated with the fabrication process.

2.2 Fabrication Costs

The cost of fabricating steel structures may be broken down into four major categories:

1. Material;
2. Labour;
3. Equipment;
4. Overhead.

Steel is purchased from a mill at a unit cost per weight, plus some additional charges also based on weight; hence, the cost of material is approximately proportional to the weight of the structure. This has been the major driving force for minimum weight design, since minimizing the weight
also minimizes the material cost. Minimum weight approaches, although academically appearing to be reasonable, have been found to provide designs which are often impractical [10].

\[\text{I know I said "minimum weight"}
\]
\[\text{... but this is not quite what I had in mind.}\]

Figure 1. Minimum weight + Minimum cost?

Labour and equipment costs are not so easily related to the form of the structure. Obviously some designs are more efficient than others in terms of ease of fabrication, but it is difficult to determine why. A key factor in these costs is material handling, which appears to be one area in the fabrication process which has large potential for improvement. Handling costs are presently assumed to be some percentage of the costs directly related to the fabrication operation. The reduction in costs associated with repetitive structural forms may be attributable to improvement in productivity for many operations in the fabrication process, such as material handling, set-up time, and labour efficiency. Designers try to account for this by using as
many similar components as possible in a structure. The rising labour and machinery costs are becoming an increasingly important part of the overall cost, so a more deterministic approach to their evaluation is needed.

The fabrication cost of the \( i \)th structural member type can be written as:

\[
c_i = N \left[ \sum_{j=1}^{n_i} c_{m j} W_j + \sum_{k=1}^{o_i} c_{l k} T_k \right]
\]

where \( c_{m j}, c_{l k} \) are unit costs for material and labour.

\( W_j \) is the weight of material \( j \).

\( T_k \) is the time required to perform operation \( k \).

\( n_i, o_i \) are the number of components and operations for member \( i \).

\( N \) is the number of identical members.

This equation appears to be the standard in use by most fabricators. The first component represents the material cost (in terms of weight) and the second comes from the direct labour costs (in terms of time).

Typical operations used in the fabrication process are: shearing, sawing, burning, punching, drilling, making templates, fitting, welding, cleaning, painting, handling, machining, etc. Equations which can be used to estimate the times associated with these operations have been developed by Leung [9], and are used as a basis for the example problem shown in the appendix. The quantity of material fabricated has an unknown influence on the productivity; hence, an interaction diagram must be developed from experience relating \( N \) to \( T \). One method of doing this is to monitor the daily operations of the fabrication plant so that a time-history plot of the work being done is available. This information can be used to identify the rate
at which work is proceeding on various components of the structure as functions of time, which can be translated into curves which represent productivity improvement due to repetition of similar acts.

The total cost of a structure can be written as the sum of the component costs:

\[
\text{Total Cost} = \sum_{j} C_j + O
\]

\( C_j \) = cost of component \( j \) (as given in equation 1)

\( O \) = overhead costs

Examining these equations reveals that: \( C_m \) and \( C_L \) are known constants; \( W \) and \( T \) are known functions of \( N \); and \( N \) is the variable that we can manipulate to reduce the costs.

2.3 Erection Costs

The current procedure for estimation of erection costs has a large uncertainty associated with its accuracy. The estimated cost of erecting an arbitrary structure has commonly been based on the material weight, without provision for the complexity of the erection procedure. Obviously, some structures are more erection friendly than others; hence, some fabricators will alter their costs by a factor from experience. Decisions of this nature have been in error by as much as 100% and more due to the large variations in productivity associated with the erection activities [9]. These variations may be attributed to the same kind of improvement trends which result from repetitive activities as encountered in the fabrication process.

The proposed procedure for evaluating erection costs follows a similar outline as for fabrication costs. Costs associated with an arbitrary
structure will be calculated from a set of standard activities performed during erection (transportation, sorting, lifting, bolting, etc.). A large portion of equipment used during erection is rented; hence, equipment cost is proportional to the erection time. This is quite different from the fabrication process where most equipment is owned by the fabricator and can be essentially considered as overhead.

The programs developed in this thesis are primarily intended for use in the fabrication process; however, they are flexible so that costs for erection and other construction activities can also be incorporated.

2.4 **Current Practice**

Most structures that are constructed in Canada and the United States follow a process which has been established for historical, political, and economical reasons. The three parties involved in the construction process are: the owner, a consultant, and a contractor. The owner employs the consultant to design the structure, and to supervise the contractor who performs the construction of the structure. Since the design is partially or fully complete at the time when the contract is put up for tender, the design usually will have been done without any interaction between the designer and the contractor. This kind of practice is only acceptable if the designer can foresee the costs associated with various structural forms, so that the design with minimum cost can be chosen. For steel structures, the majority of the cost is incurred in a fabrication plant, of which the designer has only a little knowledge. In this case it is apparent that the consultant is unable to design the most cost efficient building, unless there is some interaction with the fabricator.
The internal organization of most structural steel fabrication companies follow the functions shown in Figure 2 as adapted from a paper by Adlard [1]. Three categories are of interest to us. Business, production, and engineering all have equally important roles in the overall construction process, yet there has been a trend in the past towards concentrating effort in areas where productivity is quantifiable. Since there exists a standard method for evaluating the business aspects of one operation, by comparison against the rest of the corporation's portfolio, decisions regarding company policy are usually formulated at this level.

**Figure 2.** Organization of Structural Steel Fabrication Functions.

Strict short term cost justification methods based on return on investment have been shown to provide inadequate analysis of decisions regarding process expansion and improvement [1]. The real problem here is not that the wrong method of analysis is used, since surely all good company policy
criteria are linked in some way to profit maximization, rather that the analysis is being done without the proper information. The various functions involved in fabrication are linked as shown in Figure 3. Conventional methods of documentation lead us to a maze of information contained in: progress statements, manpower utilization summaries, cost control data, etc. The inevitable result is a process which involves much doubling up of effort and a general lack of communication between the various departments. Many of these problems can be eliminated through the use of common databases. This would be possible in a computer integrated manufacturing environment [1]; however, the scope of this thesis is limited, so the implementation of this process must be left for others.

Typical Interaction:

(1) Project Documents, Scheduling, Design Revisions/Proposals.
(2) Project Documents, Estimates, Strategic Planning, Design Evaluation.
(3) Process Data, Shop Drawings, Material Bills.
(4) Progress Reports, Manpower Utilization, Scheduling, Inventory.

Figure 3. Current Links to Design and Fabrication Functions.
Assuming that the fabricator has resolved the internal problems mentioned above, attention should be redirected to another important link: the one between the designer and the fabricator. Presently, several structural steel fabricators receive design drawings from the consultant with the invitation to tender. A detailed fabrication cost estimate is made from these drawings and submitted back to the consultant in the form of a bid. At this time the fabricator has identified some areas where savings can be made by suggesting some structurally equivalent, but more cost efficient designs for certain components of the structure. Quite often the designer may receive notice from the fabricator at the time of the bid that there may be some means of cutting costs, since this is an effective means of making his bid more attractive. But after this time, these changes are usually not suggested by the fabricator due to the inability of the design process to easily accept changes.

The characteristics of each fabrication plant depends on the size, and type of projects that the fabricator is involved with. This means that every fabrication plant will have its own set of cost variables which make it more suitable for the fabrication of certain types of structures than for others. Thus, interaction between the designer and the fabricator is essential for all projects, and must become an integral part of the construction process.

2.5 Design/Fabrication Interaction

The proposed implementation of a new design/fabrication system as opposed to that currently in existence is shown in Figures 4a and 4b. Under the current system there is no realistic method for design optimization, since there is basically no direct interaction between the designer and the
fabricator. The unidirectional flow of information in the construction of most projects is marked by the inability of the design process to accept changes (whether beneficial or not) as suggested by the fabricator.

Experience gained by the fabricator during the course of his work is often prevented by the design process from being put to its best use. The designer is unable to accommodate the fabricator's desires once the job is awarded because of an inflexible design process, yet the fabricator is not ready to "tell all" before this time since this may give away some competitive advantage.

This dilemma cannot be completely overcome in conventional practice, but a compromise can be made. Mutually acceptable revision procedures could accelerate the flow of information between the two parties so that improved designs can be put forward at the moment they are envisioned. Further cooperation involving sharing of savings, and bonuses for innovative advances, could lead to established optimization procedures.

The proposed design/fabrication process (Figure 4b) provides a mechanism for fabrication analysis, which offers continuously improved estimation and design procedures. This may remove the heavy dependence upon experience found in the current process (Figure 4a), but it arouses another large research topic. The use of such a system in practice requires further investigation and analysis in terms of the effects that this may have on the contractual obligations of the parties involved in the construction process. Direct application of the research contained in this thesis is currently possible for design-build contracts, and once the legal aspects mentioned above are resolved, application will also be made in conventional practice.
Figure 4(a). Current Design/Fabrication Process.
Figure 4(b). Proposed Design/Fabrication Process.
3. The Design Evaluation Program

"Flexibility" was the primary design criterion for a program which I called ESTImate. The use of conventional programming techniques was found to provide too many restrictions and too much coding to accommodate the arbitrary project and process data found in the fabrication industry. Recent advances in compiler construction provided a mechanism for generating an arbitrary set of FORTRAN equations, and then compiling and linking them to the rest of the program, all from within the program. This technique is referred to as a fourth generation approach to programming [13], and is one of the key elements of this thesis.

The construction of an estimation routine, which can process large quantities of data using arbitrary estimation equations, could be approached in many ways. Several computer software firms have developed programs for similar kinds of usage. Most of these programs (VISICALC, CALC, etc.) are geared towards the interactive user, and are based on the translation of screen commands into mathematical operations. The individual equations as input by the user, whether in batch mode or by direct means, must be individually converted to their mathematical equivalent. Both the logic of the expressions, in terms of the order of the operations, and the mechanism for storage during calculation must be internally controlled. This means that the entire syntax and control already available in higher level languages (FORTRAN, Pascal, C, etc.) are duplicated by these programs. Some software in this area has partially overcome this redundancy by programming in an assembly language, where the user must control all functions of data
storage and manipulation. This approach has been shown to be successful for its intended interactive application; however, if the program is to process large quantities of data using the same set of equations, then a great deal of time and storage will be wasted by redundant operations.

The solution to this difficulty in programming comes from assessing the needs of the user. The program will be used on a regular basis by designers and fabricators for cost estimation, and will only be updated when it has been shown to be in error. This will likely be done at regular intervals (weeks, months, years) depending on the quantity and type of work being done by the user. From this description of user needs, it is apparent that the previously discussed techniques are much more flexible than required for our purpose, and that another approach is needed. If the program was never to be updated by the user, then one could simply incorporate the equations right into the code. This would be acceptable if the user was familiar with the organization of the program and was able to properly link his work to the rest of the coding. In most circumstances, the personnel involved with this program will change over time and there will be some loss of familiarity associated with this transition. To avoid this problem, a system was developed to allow the construction of an estimate subroutine file (which contains the estimation equations) by means of a program. This routine, depicted in Figure 5, does the same kind of interpretation and translation that the interactive calculator programs do, but rather than compute the results it assembles a program which will be used for that operation. "READACT" is a subroutine which reads estimation functions from the activity database, while "VLE" reads a list of variables and their descriptions from a variable database. Subroutine "CREATE" assembles the estimation functions
into equivalent FORTRAN expressions, checking the logic of the input data. The result is an estimate subroutine file which contains a complete FORTRAN subroutine which will be used for estimation.

Figure 5. Construction of the Estimate Subroutine File.

The estimate subroutine file must be compiled and linked to the rest of the program to complete the program construction. The preliminary version of ESTimate uses a VAX 11 run-time library function (invoked by subroutine "GETSUB") to do this from within the program; however, later versions running on UNIX could do this by other means. The use of the estimation routine is then done as in Figure 6. A quantity database, containing all physical properties and activity methods is read by subroutine "READQUAN", while "VLE" is used to read a control file which defines the content and format of the output data. Subroutine "ESTIMATE" computes all of the estimated times/costs for each activity and writes this information to the estimate database.
Figure 6. Using the Estimation Subroutine.

Information contained in the estimate database can be used in a variety of manners. Consultants may utilize this information as a means of design evaluation in an optimization routine. Fabricators can use it to summarize their bid proposals, to analyze their fabrication plant, or to assess the accuracy of their estimation functions. The preliminary version of ESTImate has a multiple linear regression routine (Figure 7, which reads from the estimate and information system databases, using subroutines "READESTI" and "READIS", and computes regression coefficients which can be used to update the activity database.

Input mechanisms (QINPUT, ISINPUT) are not included in the existing program, but are shown in Figure 8 to demonstrate their potential use. The assembly of the quantity database can be automated using a tablet or menu approach, as shown by Leung [9], depending on the preference of the user.
Input to the information system database can be done by cross-referencing the estimate database and the actual times/costs experienced in the shop. Bid and schedule output is dependent upon both the user's preference and the hardware to be used, so they are not installed. Use of program ESTImate is explained in detail by the user's guide in the appendix.

3.2 Multiple Linear Regression and the Information System

3.2.1 Motivation for MLR

With a primary concern of assessing the accuracy of individual cost estimation parameters, solely from information which has been collected on a global basis, a systematic method of analysis is needed. This system must be
ACTIVITY DATABASE
- Estimation functions

VARIABLE DATABASE
- Variable list
- Descriptions

VARIABLE INPUT CONTROL
- Input variables
- Input descriptions

CISC DATABASE
- Section properties

QUANTITY INPUT CONTROL
- Input format

SUBROUTINE CREATE

SUBROUTINE QINPUT

INTERACTIVE INPUT
(by Estimator)

VARIABLE OUTPUT CONTROL
- Output variables

ESTIMATE SUBROUTINE FILE

QUANTITY DATABASE
- Physical properties
- Activity methods

INFORMATION SYSTEM INPUT CONTROL
- Input format

INFO

QUANTITY DATABASE
- Physical properties
- Activity methods

INFORMATION SYSTEM DATABASE
- Actual Times/Costs

SUBROUTINE ISINPUT

INTERACTIVE INPUT
(by Shop)

ESTIMATE DATABASE
- Estimated Times/Costs for each activity

INFO

SUBROUTINE BID

BID OUTPUT
- Spreadsheets
- Bid summary

SUBROUTINE MLR

MLR Coefficients
- Used to update Activity Database

SUBROUTINE SCHEDULE

SCHEDULE OUTPUT
- Progress reports
- Schedules

Completed

Not Completed

* The portion completed by this thesis is enclosed by the dotted line.

Figure 8. The Design Evaluation Program.
able to break down times (or other characteristic parameters) measured at the entrance and exit from a shop to determine how much effort was spent on various operations during the fabrication process. Multiple linear regression is a technique that can perform this kind of analysis.

![Diagram](image)

**Figure 9. The Conversion from Global to Detailed Information.**

The mechanism for collecting this global data, and the best locations for establishment of "checkstations" in a particular plant, can be determined by analyzing the combination of existing processes and the needs of the fabricator in terms of scheduling, controlling, and estimating. The collection of data for MLR is intended to take advantage of current procedures and control methods rather than to impose a new system in place of the old. The existing control system may collect all of the information required by MLR, or it may need some discrete modifications to upgrade it to that level. The individualistic nature of fabrication plants, complimented with the need to monitor and control the fabrication process, accentuates the utility of multiple linear regression analysis.

The concept of an information system based on a checkstation approach arises for psychological and economical reasons. Monitoring of individual
activities in the shop is unwanted by the workers, since this would mean that the productivity of each worker could be investigated, nor is it wanted by management since the cost and the time associated with keeping track of operations would be prohibitive. The ideal situation offered by this system would be to establish a few key locations in the fabrication plant from which global data pertaining to item progress would be extracted. The data gathered at these checkstations can then be converted to offer information which would evaluate the accuracy of the estimation process, to provide scheduling and controlling data for plant operations, and to assess the fabrication plant in terms of its productivity and economy.

This approach to information collection, organization, and analysis can provide both the short-term improvements that the fabricators are obligated to meet, as well as the long-term technological competitiveness that the North American industry is lacking [8].

3.2.2 The Information System

An information system is a planning and control process which is based on the collection, organization, and analysis of data utilizing a computer.
This system, as described by Berry [2], will gather information from the shop on a regular basis so that it may be used to analyse the fabrication process.

The programs developed in this research project are able to accommodate an arbitrary WBS so that they can be applied to existing fabrication operations.

The implementation of an information system in a non-computerized environment is likely to be done over some period of time; hence, it must be compatible with the present manual system so that regular operation can be maintained during the transition period. The requirement of compatibility is to have the ability to accept and transmit information that is normally handled by the present system without requiring extensive changes at the base level. This can be achieved by doing the replacement in a modular nature. Care must be taken to avoid simply duplicating the present system in a computerized form, since this may accentuate the limitations and inefficiencies inherent in the manual system while ignoring the newly available computer graphic aids (tablet and menu I/O mechanisms).

The operation and efficiency of an information system is dependent on the "language" of the data communication. A work breakdown structure, which represents the fabrication process in code form, must be implemented to facilitate effective communication between users of the system. The design of a work breakdown structure must be independent of the organization-responsibility structure, so that information pertaining to one item has the same meaning at various levels of production. Information which has been translated into code should be meaningful to the users, so that the work breakdown structure can be associated with physical operations in the fabrication process.
The data collected using the work breakdown structure can be organized to provide two types of information: project data, and process data. Project data pertains to a particular project or structure, and will be organized in a structure database. Each item (consisting of N identical pieces) in the structure database will be tracked through the plant by monitoring item progress at key operation checkpoints. A time/progress record for each item will be available to the fabricator, so that a deterministic analysis of the costs incurred during the fabrication of various structural components may be performed.

Process data pertains to all plant processes (cleaning, cutting, drilling/punching, welding, material handling, etc.). This data is assembled from the same monitoring operation as project data, but is organized to provide information about the efficiency of the plant itself. Simulation of the fabrication plant operation is required in order to evaluate the efficiency of the plant configuration, and with the use of the process data examination of expansion or replacement investments are made possible. Thus, an interactive plant optimization can be done by investigating each activity in the fabrication process. Monitoring the job progress from the component level provides information which would be used by engineering, drafting, purchasing, shop, shipping, and managerial operations. This information could then be used to interactively optimize the processing of material through the plant.

3.2.3 Example Use of MLR

The use of a multiple linear regression model could be applied in a variety of manners; however, in the following case study a "material bill
approach" was adopted. The actual data used for this example is presented in the ESTImate User's Guide on pages 12-24 (see Appendix). Each item on a material bill was considered independently. Five different operations were investigated (shear, saw, burn, drill, punch), and the information was collected only at the entrance and exit from this particular part of the shop. This choice of example was purely for the sake of simplicity, and would likely have to be expanded for commercial use.

The schematic drawing of a portion of a fabrication plant shown in Figure 11 represents the process under consideration. Incoming items are given pre-designated paths which they will follow through this area. Again some simplifications have been made to clarify the example.

![Diagram of Example Shop Area](image)

Figure 11. Example Shop Area.

Before the MLR analysis can be done, data contained in this material bill will be used by the estimation subroutine to calculate the estimated times to perform the fabrication operations. Then the information system subroutine receives an input file from the estimation subroutine, which
contains the estimated times to perform the operations required for each item on the material bill. Through interactive input from the user it compiles a datafile which contains the actual times measured in the shop.

Input to the MLR program includes the estimated times for each item to complete a set of fabrication operations ([X]), and the times measured by the information system that were actually experienced in the shop (Y). Within the MLR program the regression coefficients (b) and the residual differences (U) will be determined. Choosing this formulation provides output which is easily understood, since a regression coefficient of \( b_i = 1.00 \) indicates that the actual time experienced in performance of operation i is exactly the same as that which was estimated. This provides information which allows for easy adjustment of the estimation parameters, since the corrected value is simply obtained by multiplying the regression coefficient by the old value.

To test the MLR program the IS data was arbitrarily adjusted so that the actual times required to perform activities 1, 3, and 5 (shearing, burning, and punching) were all increased by 10%. Variations of this sort may appear to be semi-random to the untrained eye, yet they should be easily recognized by the MLR program. This is indicated by the MLR program since it returned values of \( b_1 = b_3 = b_5 = 1.10 \), and \( b_2 = b_4 = 1.00 \).

The steps used in the following example were:

(1) Input data is summarized in a material bill (all physical properties used by the estimation program).

(2) Estimated times [X] for each activity are calculated by the estimation program for each item on the material bill.

(3) Actual global times (Y) are measured by the information system for each item on the material bill.
ESTIMATION FUNCTIONS

Used in the construction of the Estimation Program.

Activity Database
SA = 15.0 + (1 + 0.15*LENGTH)*N
SP = 4.6 + 4.5*N
SS = (6.85 + 0.43*WTPL + 0.1*LENGTH)*N

FEEDBACK TO ESTIMATION FUNCTIONS
Not developed in this thesis.

REGRESSION COEFFICIENTS

These coefficients show how the estimation program performed on the latest project by evaluating the accuracy of each individual estimation function.

B(0) = 0.0
B(1) = 1.10
B(2) = 1.10
B(3) = 1.00
B(4) = 1.10
B(5) = 1.10
B(6) = 1.00

Figure 12. Example Use of Multiple Linear Regression (MLR).
(4) Multiple linear regression is performed on the two sets of times to identify where the estimate has been in error.

(5) The regression coefficients (b) indicate which fabrication operations were consistently in error and by how much.

While the above example has demonstrated the tremendous utility of the MLR program one must recognize that real fabrication data is not likely to be as consistent as was used there. Variations in productivity are to be expected at all times. The nature of MLR modelling is such that, if given enough data, these variations can be essentially "ironed out" so that the best values for the estimation parameters can be obtained. The degree of certainty associated with a set of regression coefficients can be measured by examining the "goodness of fit" provided by the given data. Utilization of statistics can provide an effective means for identifying data which is inconsistent with previous experience, and for locating problem areas in the fabrication process.

3.2.4 MLR - Design and Theory

The theory behind multiple linear regression is similar to least squares fitting for two dimensional sets of data. The same concepts of minimization of the sum of squared residuals apply except that the number of dimensions over which the minimization is performed is not limited. Formulation of the problem in matrix form can be done as follows (see Chatterjee (1977)) [2]:
\[ [X] = \begin{bmatrix} x_{01} & x_{11} & \cdots & x_{p1} \\ x_{02} & x_{12} & \cdots & x_{p2} \\ \vdots & \vdots & \ddots & \vdots \\ x_{0n} & x_{1n} & \cdots & x_{pn} \end{bmatrix} \]

\[ Y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}, \quad U = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_n \end{bmatrix}, \quad b = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_p \end{bmatrix} \]

(3)

where \([x]\) are the independent variables

\(Y\) are the measured data values

\(b\) are regression coefficients

\(U\) are random disturbances

\(x_{0i} = 1\) for all \(i\) (first vector simply provides for a constant value)

\(n\) is the number of data values

\(p\) is the number of contributing variables

Estimate \(b\) by minimization of the sum of squared residuals.

\[ S(b_0, b_1, \ldots, b_p) = \sum_{i=1}^{n} u_i^2 \]

(4)

\[ = \sum_{i=1}^{n} (y_i - b_0 - b_1 x_{1i} - b_2 x_{2i} - \cdots - b_p x_{pi})^2 \]

(5)

Differentiation yields the following set of linear equations to be solved:

\[ S_{11} b_1 + S_{12} b_2 + S_{13} b_3 + \cdots + S_{1p} b_p = S_{y1} \]

\[ S_{21} b_1 + S_{22} b_2 + S_{23} b_3 + \cdots + S_{2p} b_p = S_{y2} \]

\[ \vdots \]

\[ S_{p1} b_1 + S_{p2} b_2 + S_{p3} b_3 + \cdots + S_{pp} b_p = S_{yp} \]

(6)

where

\[ S_{ij} = \sum_{k=1}^{n} (x_{ik} - \bar{x}_i)(x_{jk} - \bar{x}_j) \quad i, j = 1, 2, \ldots, p \]

(7)
The final estimate is then:

\[
S_{yi} = \sum_{k=1}^{n} (y_k - \bar{y})(x_{ik} - \bar{x}_i) \quad i = 1, 2, \ldots, p
\]  \hspace{1cm} (8)

\[
\bar{x}_i = \frac{\sum_{k=1}^{n} x_{ik}}{n}
\]  \hspace{1cm} (9)

\[
\bar{y} = \frac{\sum_{k=1}^{n} y_k}{n}
\]  \hspace{1cm} (10)

Solution of these equations is done by inversion of \([S]\).

\[
[S] \cdot b = S_y
\]  \hspace{1cm} (11)

(proxp)(px1) = (px1)

providing the regression coefficients \(b_0\).

\[
b = [S]^{-1} \cdot S_y
\]  \hspace{1cm} (12)

(proxl) = (proxp)(px1)

\[
b_0 = \bar{y} - b_1 \bar{x}_1 - b_2 \bar{x}_2 - b_3 \bar{x}_3 - \cdots - b_p \bar{x}_p
\]  \hspace{1cm} (13)

The final estimate is then:

\[
y_i = b_0 + b_1 x_{1i} + b_2 x_{2i} + b_3 x_{3i} + \cdots + b_p x_{pi}
\]  \hspace{1cm} (14)

With observed residuals

\[
u_i = y_i - y_i
\]  \hspace{1cm} (15)
Application of MLR theory to the problem of assessing estimation parameters requires an examination of the fabrication process. Standard estimation equations, which come from experience, usually relate the time required to perform an act to some physical characteristic of the material (weight, length, thickness, etc.). Since the original estimation parameters were obtained through case studies, it is likely that the investigators tested similar dependent characteristics for all of the operations. For example, one property that is used in almost every case is the weight of the material. This implies that the estimation of different fabrication operations are somewhat dependent on each other, even though in reality they may be very independent processes. From MLR theory one cannot distinguish between deviations in global data for concurrently measured dependent variables; however, this problem can be solved by ensuring that some measurements are taken when one or the other variable is not present. Thus, the application of MLR is valid only if it is performed by one who is familiar with the fabrication process, and if sufficient data has been collected.

Development of extensive software for use in this area will likely be able to overcome this barrier, so that a userfriendly program can evaluate individual applications of MLR in terms of validity and logic. This kind of program would eventually provide user interface (I/O) that requires little knowledge of the processes involved in multiple linear regression, so that data can be analyzed efficiently without having to perform a set of tedious mathematical operations.
3.3 Applications of Program ESTImate

3.3.1 Design Optimization

As previously explained, structural cost optimization has not been performed in the past due to a lack of cost information. During this time, the design profession has used some other design tools, combined with some good common sense, to attain near optimum cost designs. One not very successful approach has been to calculate a minimum weight design in order to minimize the material costs. Experienced designers recognize that cost savings due to the peculiarities of the fabrication process could be attained through design simplification and repetition of elements. The problem facing designers, is still to determine the tradeoff point between these philosophies which produces the minimum cost design.

The ESTImate program can be used as one module in the optimization routine of Figure 13. Both interactive or automated alteration of components of the structure is possible since the design evaluation program is independent from the selection procedure. Optimization modules can perform various functions. One has been developed by Leung [9] to provide minimum cost designs for single large components (webs of plate girders). Another routine by Moses and Goble [11] takes a different approach by converting a minimum weight design to one of minimum cost using dynamic programming and learning curve concepts. Monte Carlo techniques could also prove to be useful in this area. Adaptations of the above techniques can all be inserted in the optimization module of Figure 13, to make a variety of improvements available to the designer.
Figure 13. Design Optimization.
3.3.2 Improvement of the Fabrication and Estimation Processes

The current estimation procedures used by most designers and fabricators are based on a few independent studies made in the past of the fabrication operations, with slight modifications reflecting their overall performances on constructed projects. The measurement of times or costs actually experienced in the performance of various operations is usually done by instigating a time study program for the process in question. This is in addition to the usual monitoring procedures used in conventional control, and it necessarily causes additional expenses which prevent it from becoming regular practice.

Implementation of an information system, which collects data on a regular basis from a global level, coupled with a multiple linear regression analysis can provide a continuous basis for updating estimation parameters. The flowchart of the ESTImate program shown in Figure 8 outlines the mechanism described above. Estimation functions contained in the activity database can be updated by the MLR coefficients obtained from each application of the MLR analysis, without requiring the additional expenses experienced by time study techniques of the past.

Additional benefits provided by the ESTImate program are: automatic bid summaries and spreadsheet analysis of the bid for pre-construction use, and progress reports and schedules for use during the construction. Potential modules can be added to the program to provide complete integration of all aspects of construction from accounting to shipping. Since real-time analysis of the fabrication process is now available, investigations may reveal where improvement is needed and is possible when there is still time to make corrections.
3.3.3 Research and Education

A major benefit provided by program ESTImate is access to "experience" which may alleviate some of the preference for analytical detachment now prevalent in the education of engineers [9]. The application of ESTImate could be done in the final years of an engineering student's education in a design project. Rather than designing to simply meet code requirements, as is so often done in education and in professional practice, designs can now be evaluated in terms of economy. Recursive design will provide students with hands on experience with code requirements and some ideas of how to improve or optimize their designs. Optimization routines such as that of Figure 13 can become common knowledge among students, and hopefully will be adopted by the firms which they will be employed by in the future.
4. SUMMARY

4.1 Conclusions

This thesis provides two new tools for designers, fabricators, and educators. The first is a flexible estimation routine, able to accept arbitrary process and project data. The second is a multiple linear regression model coupled with an information system which will provide the necessary data for the estimation routine.

The estimation program can be used as a design evaluation module for both professional practice and research. Costs can be associated with an arbitrary structural form by examining the specific capabilities of the fabrication plant. This mechanism for comparison leads to potential applications in design improvement and heuristic optimization.

Potential improvements in the fabrication and estimation processes are made possible with the use of a multiple linear regression analysis. Data required for the estimation process can now be obtained by simply analyzing data measured by a "checkstation" approach, so that the costs associated with the fabrication process can finally be monitored in an efficient manner.

Integration of scheduling, controlling, and estimating as offered by this system provides a mechanism that can assess current practice, and can identify areas of potential improvement. Designers and fabricators, who have been heavily dependent on their past experience, could take advantage of these new tools by utilizing them to improve the quality and competitiveness of structural steel designs.
4.2 Extensions of This Research

Possible extensions of the research contained in this thesis come from the application of program ESTImate to a variety of problems. Three specific areas are noted:

(1) Optimization of steel structures, and some design guidelines showing how designers could approach this problem, could realistically be attained by adopting the algorithm of Figure 13. This problem also involves the development of modules which tie in both structural analysis and code requirements, utilizing a format which is compatible with the rest of the program.

(2) A more detailed analysis of the fabrication process, along with a mathematical investigation into the statistical accuracy provided by various formulations of the multiple linear regression model, is needed to assess the implementation of an information system in an existing fabrication plant.

(3) Commercial use of the programs developed thus far is not possible without input and output routines that are specifically designed to suit the needs of the user. The programs must be transported to a microcomputer where specific routines can be developed that are efficient for use with that machine.
REFERENCES


# Table of Contents

1. INTRODUCTION .................................................................................................................. 44

2. APPLICATIONS of ESTImate .............................................................................................. 45

3. USE OF THE PROGRAM
   a) Program execution ........................................................................................................... 46
   b) ESTImate Operations ...................................................................................................... 46
      i. Editing with the Variable List Editor (VLE)
      ii. Creating the Estimate Subroutine File
      iii. Estimating
      iv. Multiple Linear Regression
   c) Data Files ....................................................................................................................... 53
      i. Descriptions
      ii. Examples

4. ESTImate COMMAND LANGUAGE .................................................................................... 66

5. SUMMARIES of ESTImate SUBROUTINES
   a) COMPARE ......................................................................................................................... 67
   b) CREATE ............................................................................................................................. 67
   c) DISPLAY ........................................................................................................................... 67
   d) EQUATION ......................................................................................................................... 68
   e) ESTIMATE ......................................................................................................................... 69
   f) FILESTAT ......................................................................................................................... 69
   g) GETSUB ........................................................................................................................... 69
   h) MLR .................................................................................................................................. 70
   i) READ .................................................................................................................................. 70
   j) STEP .................................................................................................................................. 71
   k) VLE ................................................................................................................................... 72

6. FILE NAME I/O .................................................................................................................... 73

7. ERROR MESSAGES .............................................................................................................. 74

8. DATA FORMAT .................................................................................................................... 78
   a) Activity Database (AD) ..................................................................................................... 79
   b) Estimate Database (ED) ................................................................................................... 80
   c) Information System Database (ISD) ............................................................................... 81
   d) MLR Output (MLR) ......................................................................................................... 82
   e) Quantity Database (QD) ................................................................................................. 83
   f) Variable List Editor Files (VLE) ..................................................................................... 84
1. INTRODUCTION

This program was developed at the University of British Columbia by Bruce Forde during 1984. The intent of the program was to provide a flexible estimation routine which could be used by both structural designers and fabricators in the estimation of construction costs.

Use of the program is explained in detail in the following user's guide, and comments regarding the program development and construction are placed strategically throughout the code. The language used is FORTRAN 77 with some minor adaptations for use in a Digital Equipment - VAX 11 environment.

The suggested procedure for use of this guide is to:

1. Read section 3 regarding use of the program. Then read section 4 on the ESTImate command language.

2. Try to duplicate the results of the examples given in section 3c using input data of your own (refer to section 8 on data format if necessary).

3. Sections 5 to 8 are primarily for those who wish to make alterations to the source code of ESTImate, and are included to provide a basis for program expansion and error diagnosis.

This approach ensures that the user will become familiar with both the logic and the operation of program ESTImate.
2. APPLICATIONS OF ESTImate

There are two intended users of ESTImate:

(a) Contractors  - project estimating
    - project scheduling
    - process improvement

(b) Designers  - project estimating
    - design evaluation/optimization

The largest benefits offered by ESTImate are those available to contractors, who would use it as a tool in estimating and scheduling projects, since it provides a mechanism to monitor and evaluate the efficiency of their operations. Design evaluation by contractors may also provide an effective means of reducing costs and improving profit margins so that both owners and contractors can benefit from the resulting design improvements.

Designers have a potential use of ESTImate in the evaluation/optimization of their projects; however, design/construction interaction must first be established.
3. USE OF THE PROGRAM

(a) Program Execution

To invoke program ESTImate on a Digital Equipment VAX, which is running under the VMS operating system, simply type:

RUN ESTI

This can be abbreviated to a user specified command by making use of a VMS command file (see the VMS operating system guide). Once ESTImate is running, a set of interactive commands are available (see ESTImate command language). The user can create and use an estimation program, or can evaluate the accuracy of a set of estimation equations by analysing data measured by an information system with multiple linear regression.

(b) ESTImate Operations

The flowchart of Figure 1 outlines the structure of program ESTImate. Rectangular boxes indicate datafiles, while the ovular shapes refer to self contained subroutines. Further subroutines used for I/O are indicated using arrows. Detailed summaries of the ESTImate operations and the individual subroutines are given in sections 3 and 5.

i. Editing with the Variable List Editor (VLE)

The construction of lists with accompanying descriptions can be done with the VLE subroutine. Organization of the list is by default alphabetically in terms of the variable name. In addition to interactive use of VLE, program ESTImate utilizes VLE as an I/O mechanism for several internal operations. Names of the variables should consist of only upper case alphanumeric characters (since EQUATION has support for these) with a maximum length of 10. The description can be any characters up to a length of 40. The available operations in VLE are: ADD, DEL, DESC, LIST, SAVE, SORT, STOP, and HELP. All of these commands, with the exception of DESC can be abbreviated to one letter including the lower case equivalent. DESC can be abbreviated to DES or des to prevent
Estimate USER'S GUIDE

Figure 1. Program ESTimate Flowchart.
confusion with the DEL command. DELETE is the only VLE command that includes a lower level of commands, which in this case are used to facilitate deletion by name, number, or range of numbers.

VLE should be used to establish the Variable Database and the Variable Output Control files. Further descriptions of the commands are given in the command language summary.

ii. Creating the Estimate Subroutine File

Program ESTImate requires the user to create a subroutine which contains the set of equations to be employed in the estimation process. This subroutine could be written by one who is familiar with FORTRAN, if it was ensured that all the correct statements were included to make it compatible with the rest of ESTImate. Rather than forcing the user to become very familiar with the FORTRAN coding used in the rest of the program (which may require extensive research). Program ESTImate simply writes the subroutine for the user. This way the user must only ensure that an input file is prepared with a set of equations, in a defined format, rather than having to understand the inner workings of the ESTImate program.

To create the estimate subroutine file the user should be in the first level of the ESTImate command language (the prompt must be "ESTI:"). The word "CREATE" (or an abbreviation) will instigate the CREATE subroutine prompts as shown below:

ESTI: CREATE

Activity Database Filename? <enter your filename>

Variable Database Filename? <enter your filename>

Further prompts appear to determine the direction of output:

Do you want the output sent to a file? (Y/N) <Y or N>

If the answer is "Y" then the following prompt appears:

Estimate Subroutine Filename? <enter your filename>
The filename chosen must be non-existing since ESTImate will not overwrite. It is advisable to first display the result to the screen (by choosing this option rather than to a file) to see if the input files are processed without any errors. If an error occurs, reference will be made as to where it originated (the most likely errors are in the activity database). See the section on error handling to identify where the error occurred (READACT, EQUATION, and STEP are likely candidates).

![Diagram](image)

**Figure 2. Creating the Estimate Subroutine File.**
iii. Estimating

Before the estimation routine can be invoked the GETSUB command must be used to compile and link the new estimate subroutine file to the main program. This requires about 30 seconds on a VAX 11/730; however, this procedure only has to be done once for each new activity database, not every time you want to estimate. The terminal input will be:

ESTI:GETSUB

Estimate Subroutine File? <enter your filename>

Figure 3. Estimating.
Once the estimate subroutine file has been linked, using the ESTIMATE command will result in prompts for a quantity database and an output file. Reading from this data may result in some errors as described in the section on error messages, with the correct format being given in the section on data formats. Output is controlled by the VOC (Variable Output Control) file as described in the previous section. The terminal input will be:

**ESTI:ESTIMATE**

**Quantity Database Filename ?**  <enter your filename>

Further Prompts appear to determine the direction of output:

**Do you want the output sent to a file ? (Y/N)  \(<Y\ or\ N>\)**

If the answer is "Y" then a further prompt appears:

**Estimate Database Filename ?**  <enter your filename>

The output file should be a non-existing file since the ESTImate program will not overwrite an existing file.

iv. Multiple Linear Regression

To examine the accuracy of a set of estimation equations, an estimate database (containing the estimated times calculated by the estimation equations) combined with the actual times as measured by an information system are analysed by multiple linear regression. To invoke this analysis, input will be:

**ESTI:MLR**

**Estimate Database Filename ?**  <enter your filename>

**Information System Database Filename ?**  <enter your filename>

Further prompts follow to direct the flow of output:

**Do you want the output sent to a file ? (Y/N)  \(<Y\ or\ N>\)**

If the answer is "Y" then the following prompt appears:

**MLR Output Filename  <enter your filename>**
The resulting output is a datafile which contains a set of regression coefficients which could be used to update the current set of estimation functions. This routine calls the COMPARE function which checks to see if the two input files are compatible. The definition of compatible is that every item name and description must be identical to its counterpart on the other list, and both lists must have the same number of items.

---

**ESTIMATE DATABASE**
- Estimated Times / Costs for each activity

**INFORMATION SYSTEM DATABASE**
- Actual Times / Costs

**SUBROUTINE MLR**

**MLR COEFFICIENTS**
- Used to update Activity Database

---

Figure 4. Multiple Linear Regression.
(c) Data Files

Presented on the following pages are: first a list of brief descriptions of the datafiles used in ESTImate, and then a list of examples.

i. Descriptions

Activity Database

The activity database is composed of a set of estimation functions which are to be used by the estimation routine. The format of this file is described in the data format section under the heading of AD FORMAT. The file is at present to be constructed using the system editor, but can accept new data through optimization routines in the future. The order of activities is arbitrary, but must match the code given in the activity database. Future input routines could make use of a control file which organizes the order matching the two files.

Variable Database

The variables used in the estimation process are stored alphabetically here in a VLE format. All I/O is done by the VLE routine and is described in the section on editing with the VLE editor.

Estimate Subroutine File

The ESF is a FORTRAN 77 equivalent of the activity database, describing the estimation functions in a standard program with IF THEN and assignment statements.

Quantity Database

The quantity database is composed of all physical properties and activity methods pertaining to a particular project (or portion of a project). This is similar to the material bills used by most contractors. Input of this database is not done by ESTImate version 1.00, since an efficient input mechanism has not yet been devised. The format of the data must follow that given by QD FORMAT in the data format section.
Variable Output Control File

The variables to be output by the estimation routine are stored here along with the format they are to be displayed in. This is done using the VLE editor by simply replacing the variable description with a FORTRAN F or I format statement. To get a F6.1 output, the description should be "6.1". This means a total field of 6 digits (including the decimal), with 1 digit after the decimal. To get an I3 output simply use "3" to produce an integer with a maximum of 3 digits.

Estimate Database

Estimated times/costs for each activity are summarized here with the corresponding mark numbers and descriptions. Choice of the variables in this file is controlled by VOC (below), and should be such that MLR can be performed if desired (if all estimated times are output, then if times are recorded by the IS, MLR can be performed on the combination). The format of this file is given by ED FORMAT in the data format section.

Information System Database

The times/costs measured by the information system in the shop or field are stored here with the corresponding item mark numbers and descriptions. The format of the datafile must be that given by ISD FORMAT in the data format section.

MLR Output File

This file contains the regression coefficients from MLR analysis. These coefficients can be used to update the activity database so that it reflects the current times or costs actually measured by the information system. Since this requires some judgement as to the method of alteration, it is left up to the user to devise this subroutine.
ii. Example Data Files

The following output is from the example mentioned in the M.A.Sc. Thesis of Bruce Forde: Five operations were investigated, and for clarity of use in the ESTImate program these were broken down into seven groups (due to the nature of the equations used). Information was collected at the entrance and exit from the part of shop where these operations take place. Refer to the above reference for more information on the choice of the example and the use of the results.
Activity Database

The following is a datafile which represents the estimation functions used for the estimation of seven fabrication operations.

The format is of type AD (section 8a).

shearing of plates
1
STEP/PCWT/3
0,25
SP=4.6+4.5*N
25,75
SP=6.32+7.2*N
75,10000
SP=6.59+9.7*N
RETURN

shearing of angles
2
STEP/LENGTH/2
0,15
SA=15.0+(1+0.15*LENGTH)*N
15,1000
SA=(52+0.3*LENGTH)*N
RETURN

sawing of shapes
3
SS=(6.85+0.43*WTPL+0.1*LENGTH)*N
RETURN

burning of shapes
4
BS=(1.23+0.045*WTPL)*N
RETURN

burning of plates
5
BP=12.0+(4+(2.0+0.03*THICK)*BLENGTH/1000)*N
RETURN

drilling
6
DR=(5+(1.0+0.05*THICK)*NHOLES)*N
RETURN

punching
7
PU=5.78+(0.42+0.9*NHOLES)*N
RETURN

END

<activity description>
<activity number>
<a step function in terms of "PCWT"
<range (units are those of "PCWT"
<equation applied for the above
range of the variable "PCWT">

<end of activity>

<end of all activities>
Variable Database (input using VLE)

This is all set up for you by the VLE editor by simply running ESTImate and invoking the editor. The list is internally alphabetized so the input order does not matter.

NUMVAR  14
1  BLENGTH  burnlength (mm)
2  BP  time for burning plates
3  BS  time for burning shapes
4  DR  time for drilling
5  LENGTH  member length
6  N  number of identical members
7  NHOLES  number of holes
8  PCWT  pieceweight
9  PU  time for punching
10  SA  time for shearing angles
11  SP  time for shearing plates
12  SS  time for sawing shapes
13  THICK  member thickness
14  WTPL  weight per length
The following is the program written by the create subroutine using the activity database and the variable list. The names of the variables in the equations (as input by the user) are translated into an equivalent element in the V-vector, having the corresponding number from the variable list. The step functions are converted to FORTRAN IF THEN statements containing the values of the range given by the user. The lower number is contained in the range while the upper number is the upper bound.

```fortran
REAL V(14)
INTEGER NVAR, ACTCODE(*), NACT
CHARACTER*10 NAME(*)
NVAR = 14
NAME( 1) = 'BLENGTH '
NAME( 2) = 'BP '
NAME( 3) = 'BS '
NAME( 4) = 'DR '
NAME( 5) = 'LENGTH '
NAME( 6) = 'N '
NAME( 7) = 'NHOLES '
NAME( 8) = 'PCWT '
NAME( 9) = 'PU '
NAME(10) = 'SA '
NAME(11) = 'SP '
NAME(12) = 'SS '
NAME(13) = 'THICK '
NAME(14) = 'WTPL '

! Shearing of plates

IF(ACTCODE( 1).EQ.1) THEN
  IF(V( 8).GE.0.AND.V( 8).LT.25) THEN
    V(11) = 4.6 + 4.5*V( 6)
  END IF
  IF(V( 8).GE.25.AND.V( 8).LT.75) THEN
    V(11) = 6.32 + 7.2*V( 6)
  END IF
  IF(V( 8).GE.75.AND.V( 8).LT.10000) THEN
    V(11) = 6.59 + 9.7*V( 6)
  END IF
END IF
```
Cshearing of angles

IF(ACTCODE(2).EQ.1)THEN
  IF(V(5).GE.0.AND.V(5).LT.15)
    V(10)=15.0+(1+0.15*V(5))*V(6)
  IF(V(5).GE.15.AND.V(5).LT.1000)
    V(10)=(52+0.3*V(5))*V(6)
END IF

Csawing of shapes

IF(ACTCODE(3).EQ.1)THEN
  V(12)=(6.85+0.43*V(14)+0.1*V(5))*V(6)
END IF

Cburning of shapes

IF(ACTCODE(4).EQ.1)THEN
  V(3)=(1.23+0.045*V(14))*V(6)
END IF

Cburning of plates

IF(ACTCODE(5).EQ.1)THEN
  V(2)=12.0+(4+(2.0+0.03*V(13))*V(1)/1000)*V(6)
END IF

Cdrilling

IF(ACTCODE(6).EQ.1)THEN
  V(4)=(5+(1.0+0.05*V(13))*V(7))*V(6)
END IF

Cpunching

IF(ACTCODE(7).EQ.1)THEN
  V(9)=5.78+(0.42+0.9*V(7))*V(6)
END IF

NACT=7
RETURN
END
Quantity Database (input using the VMS editor)

The following is the quantity database (structure database, material bill, etc.) to be estimated using the chosen estimation functions and program given above. The format follows that given by the QD format of section 8e.

PROJECT 501 <project name>
Material bill 12 <quantity database description>

(A3,X,A13) <format of mark,description>

7 - number of potential activities

(X,7I1) <format of activity code>

7 - number of input variables

6,5,14,8,1,7,13 - order of input variables

* <format of variables>

14 - number of variables in database

(T10,A10,T25,A40) <format of variable list>

1 BLENGTH burnlength (mm)
2 BP time for burning plates
3 BS time for burning shapes
4 DR time for drilling
5 LENGTH member length
6 N number of identical members
7 NHOLES number of holes
8 PCWT pieceweight
9 PU time for punching
10 SA time for shearing angles
11 SP time for shearing plates
12 SS time for sawing shapes
13 THICK member thickness
14 WTPL weight per length

1 W920X253 <mark,description>
0001001 <activity code>
15.0 12.0 253.0 0.0 920.0 20.0 28.0
2 W760X147
0001001
30 8.2 147.0 0.0 760.0 16 17.0
3 W610X140
0010010
20 9.1 140.0 0.0 0.0 16 22.0
<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>W530X109</td>
<td>0001001</td>
<td>25</td>
<td>8.0</td>
<td>109.0</td>
<td>0.0</td>
</tr>
<tr>
<td>5</td>
<td>W460X74</td>
<td>0010010</td>
<td>10</td>
<td>6.7</td>
<td>74.0</td>
<td>0.0</td>
</tr>
<tr>
<td>6</td>
<td>W360X79</td>
<td>0010001</td>
<td>28</td>
<td>4.6</td>
<td>79.0</td>
<td>0.0</td>
</tr>
<tr>
<td>7</td>
<td>W250X89</td>
<td>0010100</td>
<td>40</td>
<td>6.0</td>
<td>89.0</td>
<td>0.0</td>
</tr>
<tr>
<td>8</td>
<td>L125X125X10</td>
<td>010001</td>
<td>30</td>
<td>4.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>9</td>
<td>L90X90X10</td>
<td>0100001</td>
<td>18</td>
<td>3.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>10</td>
<td>L90X90X10</td>
<td>0100001</td>
<td>24</td>
<td>2.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>11</td>
<td>L75X75X10</td>
<td>0100001</td>
<td>50</td>
<td>1.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>12</td>
<td>PL50X720</td>
<td>0001110</td>
<td>4</td>
<td>15.0</td>
<td>0.0</td>
<td>4212.0</td>
</tr>
<tr>
<td>13</td>
<td>PL14X3860</td>
<td>0001110</td>
<td>2</td>
<td>15.0</td>
<td>0.0</td>
<td>6322.7</td>
</tr>
<tr>
<td>14</td>
<td>PL30X600</td>
<td>0001110</td>
<td>8</td>
<td>10.0</td>
<td>0.0</td>
<td>1404.0</td>
</tr>
<tr>
<td>15</td>
<td>PL12X1500</td>
<td>0001110</td>
<td>4</td>
<td>10.0</td>
<td>0.0</td>
<td>1404.0</td>
</tr>
</tbody>
</table>

END
Variable Output Control  (input using VLE)

The list of output variables is contained in the VOC file as well as their output formats. In this case, all variables to do with times were output so that MLR could be performed on the operations in question. The formats used are F8.1 as discussed in the file description for VOC.

<table>
<thead>
<tr>
<th>NUMVAR</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>BP</td>
<td>8.1</td>
</tr>
<tr>
<td>2</td>
<td>BS</td>
<td>8.1</td>
</tr>
<tr>
<td>3</td>
<td>DR</td>
<td>8.1</td>
</tr>
<tr>
<td>4</td>
<td>PU</td>
<td>8.1</td>
</tr>
<tr>
<td>5</td>
<td>SA</td>
<td>8.1</td>
</tr>
<tr>
<td>6</td>
<td>SS</td>
<td>8.1</td>
</tr>
</tbody>
</table>
Estimate Database  (output by subroutine ESTIMATE)

This is the times or costs as estimated by the estimation subroutine. The input format is as given in the ED FORMAT in section 8b and is very similar to the ISD FORMAT of the next page.

PROJECT 501

Material bill 12
6 Output variables

\((T10,A10,T25,A40)\)

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>BP</td>
<td>time for burning plates</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>BS</td>
<td>time for burning shapes</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>DR</td>
<td>time for drilling</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PU</td>
<td>time for punching</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SA</td>
<td>time for shearing angles</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SS</td>
<td>time for sawing shapes</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>


<table>
<thead>
<tr>
<th></th>
<th>B</th>
<th>B</th>
<th>D</th>
<th>P</th>
<th>S</th>
<th>S</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>P</td>
<td>S</td>
<td>R</td>
<td>U</td>
<td>A</td>
<td>S</td>
</tr>
<tr>
<td>1</td>
<td>W920X253</td>
<td>0.0</td>
<td>189.2</td>
<td>0.0</td>
<td>282.1</td>
<td>0.0</td>
</tr>
<tr>
<td>2</td>
<td>W760X147</td>
<td>0.0</td>
<td>235.4</td>
<td>0.0</td>
<td>450.4</td>
<td>0.0</td>
</tr>
<tr>
<td>3</td>
<td>W610X140</td>
<td>0.0</td>
<td>0.0</td>
<td>772.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>4</td>
<td>W530X109</td>
<td>0.0</td>
<td>153.4</td>
<td>0.0</td>
<td>241.3</td>
<td>0.0</td>
</tr>
<tr>
<td>5</td>
<td>W460X74</td>
<td>0.0</td>
<td>0.0</td>
<td>254.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>6</td>
<td>W360X79</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>269.5</td>
<td>0.0</td>
</tr>
<tr>
<td>7</td>
<td>W250X89</td>
<td>0.0</td>
<td>209.4</td>
<td>1236.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>8</td>
<td>L125X125X10</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>180.4</td>
<td>63.0</td>
</tr>
<tr>
<td>9</td>
<td>L90X90X10</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>110.5</td>
<td>41.1</td>
</tr>
<tr>
<td>10</td>
<td>L90X90X10</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>102.3</td>
<td>46.2</td>
</tr>
<tr>
<td>11</td>
<td>L75X75X10</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>116.8</td>
<td>72.5</td>
</tr>
<tr>
<td>12</td>
<td>PL50X720</td>
<td>258.2</td>
<td>0.0</td>
<td>188.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>13</td>
<td>PL14X3860</td>
<td>130.0</td>
<td>0.0</td>
<td>132.4</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>14</td>
<td>PL30X600</td>
<td>303.8</td>
<td>0.0</td>
<td>120.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>15</td>
<td>PL12X1500</td>
<td>150.7</td>
<td>0.0</td>
<td>109.6</td>
<td>0.0</td>
<td>0.0</td>
</tr>
</tbody>
</table>
Information System Database (input using the VMS editor)

This is the times or costs as measured by the information system. The input format is as given in the ISD FORMAT of section 8c and is very similar to the ED FORMAT shown on the previous page.

PROJECT 501
Material bill 12
6 Output variables
(T10,A10,T25,A40)

1 BP time for burning plates
2 BS time for burning shapes
3 DR time for drilling
4 PU time for punching
5 SA time for shearing angles
6 SS time for sawing shapes

(X,A3,X,A13,F8.1)

1 W920X253 518.4
2 W760X147 754.4
3 W610X140 2131.2
4 W530X109 434.2
5 W460X74 647.4
6 W360X79 1452.3
7 W250X89 1466.3
8 L125X125X10 267.7
9 L90X90X10 166.8
10 L90X90X10 163.4
11 L75X75X10 208.2
12 PL50X720 472.0
13 PL14X3860 275.4
14 PL30X600 454.2
15 PL12X1500 275.4
Multiple Linear Regression  (output by subroutine MLR)

This is the set of regression coefficients that best fit the combination of times or costs estimated by the estimation subroutine and those measured by the information system. The results show the expected result that items 1, 2, 4, and 5 should all be multiplied by a factor of 1.10 while the others remain unchanged.

The matrix of times shown below are the reestimated times, (not the initial input from the information system) and are in good agreement with the information system.

PROJECT 501
Material bill 12

\[(T_{10}, A_{10}, T_{25}, A_{40})\]

<table>
<thead>
<tr>
<th></th>
<th>BP</th>
<th>time for burning plates</th>
<th>0.0</th>
<th>189.2</th>
<th>0.0</th>
<th>282.1</th>
<th>0.0</th>
<th>518.4</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td>B( 0) = 0.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>BS</td>
<td>B( 1) = 1.10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>DR</td>
<td>B( 2) = 1.10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>PU</td>
<td>B( 3) = 1.00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>SA</td>
<td>B( 4) = 1.10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>SS</td>
<td>B( 5) = 1.10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>B( 6) = 1.00</td>
<td></td>
<td>0.0</td>
<td>235.4</td>
<td>0.0</td>
<td>450.4</td>
<td>0.0</td>
<td>754.4</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>0.0</td>
<td>0.0</td>
<td>772.0</td>
<td>0.0</td>
<td>241.3</td>
<td>0.0</td>
<td>434.2</td>
</tr>
<tr>
<td>4</td>
<td>PU</td>
<td>B( 6) = 1.00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| 0.0 | 0.0 | 0.0 | 393.4 | 647.4 | 0.0 | 0.0 | 1155.8 | 1452.3 |
| 6 |    | 0.0 | 0.0 | 269.5 | 0.0 | 1155.8 | 1452.3 | 1466.3 |
| 7 |    | 0.0 | 0.0 | 209.4 | 1236.0 | 0.0 | 0.0 | 1466.3 |
| 8 |    | 0.0 | 0.0 | 180.4 | 0.0 | 1359.2 | 2131.2 | 267.7 |
| 9 |    | 0.0 | 0.0 | 102.3 | 0.0 | 2131.2 | 267.7 |
| 10 |   | 0.0 | 0.0 | 116.8 | 0.0 | 472.0 | 0.0 | 275.4 |
| 11 |   | 0.0 | 0.0 | 120.0 | 0.0 | 0.0 | 0.0 | 472.0 |
| 12 |   | 0.0 | 0.0 | 109.6 | 0.0 | 0.0 | 0.0 | 275.4 |
4. ESTImate COMMAND LANGUAGE

The ESTImate program is controlled by a command language which is set up in a hierarchical manner. Level 1 is accessed through the main program. Level 2 is accessed through the current main subroutines. Level 3 is accessed through the current lower level subroutines. In version 1.00 of ESTImate there is only one set of level 2 and 3 commands. This will be expanded in later versions where I/O commands will be essential.

LEVEL 1

ESTImate Commands are:
CREATE - creates a set of ESTImate subroutines
GETSUB - select an estimate subroutine file
ESTIMATE - use estimate subroutine
ISINPUT - add input to the information system database
QINPUT - add input to the quantity database
MLR - perform Multiple Linear Regression
VLE - edit variable lists
DISPLAY - display a file
STOP - return to VMS
HELP - print this message

LEVEL 2.1

VARIABLE LIST EDITOR Commands are:
ADD - add a variable and its description
DEL - delete one or more variables
DESC - change a variable description
LIST - display the variable list
SAVE - save the new variable list
SORT - sort alphabetically
STOP - return to active mode
HELP - print this message

LEVEL 3.1

DELETE Commands are:
NAME - deletes one variable by name
NUM - deletes variable number i
RANGE - deletes variables numbered i to j
STOP - return to VLE mode
HELP - print this message
SUBROUTINE COMPARE(VARLIST1, VARLIST2, NUMVAR1, NUMVAR2, COMMAND)

VARLIST1 - the first list of variables.
VARLIST2 - the second list of variables.
NUMVAR1 - the number of variables (from VARLIST1).
NUMVAR2 - the number of variables (from VARLIST2).
COMMAND - control phrase:
          "SAME" - lists are identical.
          "ERR" - lists are not identical.

This subroutine compares two variable lists. Every item must match identically on the two lists including the number of items.

SUBROUTINE CREATE

Analyzing estimation functions from the activity database, with the list of potential variables from the variable database, CREATE constructs a FORTRAN program which essentially matches the logic of the estimation functions. The estimation functions must be input according to the AD format initially by hand, but can potentially be updated automatically using the MLR output. Subroutines EQUATION and STEP are used to perform the data analysis, and subroutines READACT and VLE are called to retrieve data from storage.

SUBROUTINE DISPLAY

This subroutine prompts the user for a file to be displayed, accepting the default of .DAT as the data type. An error may occur if the user specifies a type other than a text file.
SUBROUTINE EQUATION(VARLIST, VARDESC, NUMVAR, INPUT, OUTPUT)

VARLIST - the list of variables.
VARDESC - the list of variable descriptions.
NUMVAR - the number of variables.
COMMAND - control phrase (if set to "STOP" then VLE returns to MAIN or can be set to "GO" to become an interactive editor).
INPUT - a character string containing the input equation.
OUTPUT - a character string containing the output equation.

This subroutine was developed in a test program called INT and acts as the interpreter of equations used in ESTImate. Input equations are decomposed into variables, constants, and operators and the logic of the expression is tested. The result is an equivalent FORTRAN equation written to OUTPUT. Each equation must consist of a variable name and an equals sign followed by a string of variables, constants and operators. Variable names can only be made up of non-numeric capital letters.

EXAMPLE: TIME = 10.1 * (WEIGHT + LENGTH/1.62)

NOTATION:
L = letter A, B, C, ..., Z
D = digit 1, 2, 3, ..., 0
O = operator (+, -, *, /)
LSIG = last significant character
NDECIMAL = number of decimal places per constant (maximum of 1)

LOGIC:
L can follow L O ( = or blank if LSIG is O, ( or =
D can follow D O ( . = or blank if LSIG is O, ( or =
O can follow L D ) = or blank if LSIG is L, D or )
( can follow O ( = or blank if LSIG is O or ( ) can follow L D ) = or blank if LSIG is L, D or )
. can follow D O ( = or blank if LSIG is O or ( BUT only if NDECIMAL=0 (reset)
= can follow L BUT MUST FOLLOW FIRST VARIABLE
SUBROUTINE ESTIMATE

Once linked to the estimate subroutine file, this subroutine estimates the values of all variables in the variable output control file VOC.DAT, using the quantity database of the user's choice. Subroutines VLE and READQUAN are called to retrieve data.

SUBROUTINE FILESTAT(NAME, TYPE, STAT)

NAME - the name of the file
TYPE - the data type
STAT - a logical variable

The call to this subroutine should supply the name of a file to be tested, and a value of STAT (a logical variable).

INPUT VALUE of STAT   TRUE   TRUE   FALSE   FALSE
FILESTATUS           EXIST  NONEXIST  EXIST  NONEXIST
OUTPUT VALUE of STAT  TRUE   FALSE   TRUE   FALSE
ERROR MESSAGE        NO     YES     YES    NO

The input of a new filename must be done by the caller.

SUBROUTINE GETSUB(SUBFILE)

SUBFILE - name of the estimate subroutine file

This subroutine makes use of the VAX Run Time Library function which accepts a command to compile and link the selected estimation subroutine file while using ESTImate. A command file GETSUB.COM is called to perform this action passing the name of the file as a parameter. In ESTImate version 1.00 this requires about 30 seconds on a VAX 11/730.
SUBROUTINE MLR

This subroutine was developed to investigate the potential use of a multiple linear regression model in the evaluation of the estimation of structural steel fabrication costs. The preliminary version specifically dealt with a case study where the progress of specific member types had been monitored by an information system, which records the time at which the members pass key checkstations in the fabrication plant. The estimation process was simplified so that a working program could be written over a short period of time.

Subsequently the ESTImate program was written, and was done in such a manner so that MLR could be attached to it. The original limitations of the case study were eliminated so that the use of MLR is now unrestricted.

ESTImate version 1.00 has no input mechanism for information system data; however, it will accept data of ISD format for use in the MLR routine (see section 8c).

READ subroutine group

SUBROUTINE READNAME(NUMBER,NAME)
SUBROUTINE READNUM(NUMBER,NAME)
SUBROUTINE READACT(UNIT,ACTNUM,ACTDESC,ACTDAT,NEQUA)
SUBROUTINE READQUAN(UNIT,NAME,DESC,NUMVAR,FMT,
                      VARLIST,VARDESC,MARK,QDESC,AECTCODE,NACT,V,COMMAND)
SUBROUTINE READESTI(UNIT,NAME,DESC,NUMOUT,OUTLIST,OUTDESC,
                      NUMQUAN,MARK,QDESC,X,I,J)
SUBROUTINE READIS(UNIT,NAME,DESC,NUMOUT,OUTLIST,OUTDESC,
                   NUMQUAN,MARK,QDESC,Y,I)

Each of the above routines are used as mechanisms for collecting data from the available databases. See sections on error messages and file organization for more information.
SUBROUTINE STEP (ACTDAT, POINTER, VARLIST, NUMVAR, STEPDAT, N)

ACTDAT - the activity data list.
POINTER - the location in the activity data.
VARLIST - the list of variables.
NUMVAR - the number of variables.
STPDAT - the step function data list.
N - the number of step function equations.

The organization of a step function is as follows:

STEP/<variable name>/<number of equations>
<value1>,<value2>
  . . . equation . .
<value3>,<value4>
  . . . equation . .
  .
<valueN-1>,<valueN>
  . . . equation . .

The word "STEP" is read by the calling routine, but the entire set of data is still passed to this subroutine in vector ACTDAT, with the position in the vector marked by POINTER. The variable <varname> is first drawn from VARLIST to see if it exists. If it does, then V(i) is used in the IF THEN statements in the subroutines, else an error message is printed. Following this the bounds for <varname> which apply to the first equation are read. Subroutine EQUATION is called to read the following line returning the FORTRAN equation which is then written with the IF THEN statement to vector STEPDAT. This procedure continues until the last equation is processed. The number of spaces used in STEPDAT is returned in N.
SUBROUTINE VLE(VARLIST, VARDESC, NUMVAR, COMMAND, VARFILE)

VARLIST - the list of variables.
VARDESC - the list of variable descriptions.
NUMVAR - the number of variables.
COMMAND - control phrase:
  "STOP" - returns to MAIN with VARLIST and VARDESC.
  "GO" - VLE becomes an interactive editor.
  "NEW" - a new file can be opened in the editor.
  "ERR" - means that an error occurred in VLE.
VARFILE - the name of the datafile which contains the variables.

This subroutine performs two functions:

1. VARIABLE LIST EDITOR
   - sorts, adds, and deletes variables from a list.
   - displays the list

2. I/O of LIST TO FILE STORAGE
   - retrieves variable lists from storage files.
   The file is an internally alphabetized sequential type.
6. FILE NAME I/O

NOTE: later versions of ESTImate will provide access to bypass these prompts by stacking of the commands issued by the user. The following is a list of the prompts issued by ESTImate:

CREATE

Activity Database Filename ?
Variable Database Filename ?
Do you want the output sent to a file? (Y/N)
Estimate Subroutine Filename ?

DISPLAY

File to be displayed?
The default file type is FILE.DAT
If you want to display some other type then you have to write the full name.

ESTIMATE

Quantity Database Filename ?
Do you want the output sent to a file? (Y/N)
Estimate Database Filename ?

GETSUB

Estimate Subroutine Filename ?

MLR

Estimate Database Filename ?
Information System Database Filename ?
Do you want the output sent to a file? (Y/N)
MLR Output Filename ?
7. ERROR MESSAGES

The following list shows the subroutines where a particular error message originates so that efficient diagnosis of any non-recoverable problem is possible.

COMPARE

Number of variables in list1:
Number of variables in list2:
DATAFILES SELECTED ARE INCOMPATIBLE
• • • • • • ERROR • • • • • •
DATAFILES SELECTED ARE INCOMPATIBLE

CREATE

The names of the two files must be different
• • • • • • • • • • • • TRY AGAIN • • • • • • • • • • • •
"VOC" is reserved for the OUTPUT CONTROL FILE
• • • • • • • • • • • • TRY AGAIN • • • • • • • • • • • •

DISPLAY

An error occurred when data was being read
An error occurred when the file was OPENED
The most likely causes of this are:
1. Incorrect file specification for type.
2. Data with unusual attributes.

EQUATION

• • • • • • • • • • • • ERROR IN VARIABLE NAME • • • • • • • • • • • •
• • • • • • • • • • • • ERROR IN CONSTANT DESIGNATION • • • • • • • • • • • •
• • • • • • • • • • • • ERROR IN OPERATOR PLACEMENT • • • • • • • • • • • •
• • • • • • • • • • • • ERROR IN LEFT BRACKET PLACEMENT • • • • • • • • • • • •
• • • • • • • • • • • • ERROR IN RIGHT BRACKET PLACEMENT • • • • • • • • • • • •
• • • • • • • • • • • • ERROR IN DECIMAL PLACE • • • • • • • • • • • •
• • • • • • • • • • • • TOO MANY DECIMAL PLACES • • • • • • • • • • • •
• • • • • • • • • • • • ERROR IN_EQUALS SIGN PLACEMENT • • • • • • • • • • • •
• • • • • • • • • • • • BRACKETS NOT BALANCED • • • • • • • • • • • •
• • • • • • • • • • • • TOO MANY_EQUALS SIGNS • • • • • • • • • • • •
• • • • • • • • • • • • VARIABLE "",VARNAME,"," DOES NOT EXIST • • • • • • • • • • • •

ESTIMATE

You have not linked the estimate subroutine file to the ESTIMATE subroutine (use the GETSUB command).
"VOC" is reserved for VARIABLE OUTPUT CONTROL
• • • • • • • • • • • • TRY AGAIN • • • • • • • • • • • •
ERROR TOO MANY_VARIABLES IN OUTPUT LIST
solution: use several output files
or reduce the size of the variables.
REVISE THE DATAFILE
MLR

•••••• ERROR *•••••
DATAFILES SELECTED ARE INCOMPATIBLE

READACT

•••••• ERROR (OR END IN READ) •••••
The most common errors found are:
1. Activity description missing.
2. Activity number missing.
3. RETURN or END missing.
See the ESTImate user guide regarding
"Organization of the Activity File".

READESTI

•••••• END IN READ OF NAME ••••••
•••••• ERROR IN READ OF NAME ••••••
•••••• END IN READ OF DESC ••••••
•••••• ERROR IN READ OF DESC ••••••
•••••• END IN READ OF NUMOUT ••••••
•••••• ERROR IN READ OF NUMOUT ••••••
•••••• END IN READ OF FMT(1) ••••••
•••••• ERROR IN READ OF FMT(1) ••••••
•••••• END IN READ OF BLANK ••••••
•••••• ERROR IN READ OF BLANK ••••••
•••••• END IN READ OF OUTLIST ••••••
•••••• ERROR IN READ OF OUTLIST ••••••
•••••• END IN READ OF FMT(2) ••••••
•••••• ERROR IN READ OF FMT(2) ••••••
•••••• ERROR IN READ OF X VECTOR ••••••
See the ESTImate user guide regarding
"Organization of the Estimate File".
READIS

***** END IN READ OF NAME *****
***** ERROR IN READ OF NAME *****
***** END IN READ OF DESC *****
***** ERROR IN READ OF DESC *****
***** END IN READ OF NUMOUT *****
***** ERROR IN READ OF NUMOUT *****
***** END IN READ OF FMT(1) *****
***** ERROR IN READ OF FMT(1) *****
***** END IN READ OF BLANK *****
***** ERROR IN READ OF BLANK *****
***** END IN READ OF OUTLIST *****
***** ERROR IN READ OF OUTLIST *****
***** END IN READ OF FMT(2) *****
***** ERROR IN READ OF FMT(2) *****
***** ERROR IN READ OF Y VECTOR *****

See the ESTATmate user guide regarding
"Organization of the Information System File".

READQUAN

FREE FORMAT IS NOT ALLOWED FOR MARK or DESC
***** END IN READ OF NAME *****
***** ERROR IN READ OF NAME *****
***** END IN READ OF DESC *****
***** ERROR IN READ OF DESC *****
***** END IN READ OF NACT *****
***** ERROR IN READ OF NACT *****
***** END IN READ OF NVAR *****
***** ERROR IN READ OF NVAR *****
***** END IN READ OF ORDER *****
***** ERROR IN READ OF ORDER *****
***** END IN READ OF NUMVAR *****
***** ERROR IN READ OF NUMVAR *****
***** END IN READ OF FMT(1) *****
***** ERROR IN READ OF FMT(2) *****
***** ERROR IN READ OF FMT(3) *****
***** ERROR IN READ OF FMT(4) *****
***** END IN READ OF VARIABLE(I) *****
***** END IN READ OF VARIABLE(I) *****
***** END IN READ OF MARK *****
***** ERROR IN READ OF MARK *****
***** END IN READ OF ACTCODE(I) *****
***** ERROR IN READ OF ACTCODE(I) *****
***** END IN READ OF V(ORDER(I)) *****
***** ERROR IN READ OF V(ORDER(I)) *****

See the ESTATmate user guide regarding
"Organization of the Quantity File".
STEP

****** VARIABLE "VARNAME" DOES NOT EXIST ******
****** ERROR IN NUMBER ******
ERROR IN READ OF ACTDAT
**TOO MANY DECIMAL PLACES**
****** ERROR IN NUMBER ******

VLE

Your data file has a mistake in
the number of variables "NUMVAR"
THE ACTUAL NUMBER IS . . . .
NOT . . . . . . . . . . . . .
If you SAVE the file, and start again
this will be corrected for you.
Number of blank variables encountered =
****** ERROR BAD DATA FILE ******
NAME is already on the variable list
NAME is not on the variable list
NO VARIABLES WERE DELETED
ERROR WAS:
No variable of name
NO VARIABLES WERE DELETED
ERROR WAS:
No variable of number
NOTE: If J is less than I, then only
only VARLIST(I) is deleted.
****** ERROR IN RANGE NUMBER ******
NO VARIABLES WERE DELETED
ERROR WAS:
No variables in range
8. DATA FORMAT

The flowchart showing the organization of the ESTImate program (Fig 1) displays all datafiles as being enclosed by rectangles. The following list identifies the type of format used by each datafile. The format types are given in alphabetical order following this page.

ESTImate Datafiles:

A. PROCESS DATA
- ACTIVITY DATABASE AD
- VARIABLE DATABASE VLE
- MLR OUTPUT MLR

B. PROJECT DATA
- QUANTITY DATABASE QD
- ESTIMATE DATABASE ED
- INFORMATION SYSTEM DATABASE IS
- BID OUTPUT N/A
- SCHEDULE OUTPUT N/A

C. ESTImate CONTROL DATA
- VARIABLE INPUT CONTROL VLE
- QUANTITY INPUT CONTROL QIC
- INFORMATION SYSTEM INPUT CONTROL ISIC
- VARIABLE OUTPUT CONTROL VLE
AD FORMAT

The organization of the activity file is as follows:

activity description
ACTIVITY NUMBER
  ...
  . . . equations . . .
  ...
RETURN

activity description
ACTIVITY NUMBER
  ...
  . . . equations . . .
  ...
RETURN
END

For more information on the equations, see EQUATION and STEP subroutine summaries. Also see the example
ED FORMAT

The organization of the estimate file is as follows:

NAME - Name of quantity file
DESC - Quantity data description
NUMOUT - Number of output variables
FMT(1) - Format of variable numbers, names and descriptions
  1  OUTLIST(1)  OUTDESC(1)
  2  OUTLIST(2)  OUTDESC(2)
  3  OUTLIST(3)  OUTDESC(3)
    :    :    :    :
    :    :    :    :
    :    :    :    :
NUMOUT  OUTLIST(NUMOUT)  OUTDESC(NUMOUT)

FMT(2) - Format of estimate output
  :  :  :  :  :  :  :
  :  :  :  :  :  :
MARK QDESC V(1) V(2) V(3) .. V(NUMOUT)
  :  :  :  :  :  :
  :  :  :  :  :
  :  :  :  :

See the example in section 3cii.
**ISD FORMAT**

The organization of the information system file is as follows:

- **NAME** - Name of quantity file
- **DESC** - Quantity data description
- **NUMOUT** - Number of output variables
- **FMT(1)** - Format of variable numbers, names and descriptions
  1. `OUTLIST(1)` `OUTDESC(1)`
  2. `OUTLIST(2)` `OUTDESC(2)`
  3. `OUTLIST(3)` `OUTDESC(3)`
  .
  .
  .
- **NUMOUT** `OUTLIST(NUMOUT)` `OUTDESC(NUMOUT)`
- **FMT(2)** - Format of information system output
  .
  .
  .
- **MARK** `QDESC` `Y(I)`
  .
  .
  .

See the example in section 3cii.
MLR FORMAT

The organization of the MLR output file is as follows:

NAME - Name of quantity file
DESC - Quantity data description
NUMOUT - Number of output variables
FMT(1) - Format of variable numbers, names and descriptions

<table>
<thead>
<tr>
<th></th>
<th>OUTLIST(1)</th>
<th>OUTDESC(1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>OUTLIST(2)</td>
<td>OUTDESC(2)</td>
</tr>
<tr>
<td>3</td>
<td>OUTLIST(3)</td>
<td>OUTDESC(3)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

NUMOUT OUTLIST(NUMOUT) OUTDESC(NUMOUT)

B(0)
B(1)
B(2)

See the example in section 3cii.
QD FORMAT

The organization of the quantity file is as follows:

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NAME</td>
<td>QUANTITY DATABASE NAME</td>
</tr>
<tr>
<td>DESC</td>
<td>quantity database description</td>
</tr>
<tr>
<td>FMT(1)</td>
<td>format of first data line (member mark &amp; description)</td>
</tr>
<tr>
<td>NACT</td>
<td>number of activities</td>
</tr>
<tr>
<td>FMT(2)</td>
<td>format of second data line (the activity code)</td>
</tr>
<tr>
<td>NVAR</td>
<td>number of input variables V</td>
</tr>
<tr>
<td>ORDER(nvar)</td>
<td>order of input variables in quantity database</td>
</tr>
<tr>
<td>FMT(3)</td>
<td>format of third data line (the input variables)</td>
</tr>
<tr>
<td>NUMVAR</td>
<td>number of variables V</td>
</tr>
<tr>
<td>FMT(4)</td>
<td>format of variable list</td>
</tr>
</tbody>
</table>

(1) VARLIST(1)     VARDESC(1)
(2) VARLIST(2)     VARDESC(2)
(3) VARLIST(3)     VARDESC(3)

\[
\begin{align*}
\text{data line 1} & : & \text{data line 1 contains:} \\
\text{data line 2} & : & \text{data line 2 contains:} \\
\text{data line 3} & : & \text{data line 3 contains:}
\end{align*}
\]

- Member mark, Member description
- Activity method code
- Vector V(NVAR) for the member

1 data line 1       data line 1 contains:
1 data line 2       - Member mark, Member description
1 data line 3       data line 2 contains:
1 data line 1       - Activity method code
1 data line 2       data line 3 contains:
1 data line 3       - Vector V(NVAR) for the member

END (control for looping) must be in the first column!

See the example in section 3cii.
VLE FORMAT

The organization of VLE files is as follows:

**FORTRAN format**

NUMBER OF VARIABLES
1 NAME(1) DESC(1)
2 NAME(2) DESC(2) (T10,A10, T25,A40)
3 NAME(3) DESC(3)
...
NUMVAR NAME(NUMVAR) DESC(NUMVAR)

See the example in section 3cii.