AD-783 392

DEVELOPMENT OF A METHOD FOR THE ANALYSIS OF IMPROVED HELICOPTER DESIGN CRITERIA

Ross F. Metzger, et al

Kaman Aerospace Corporation

Prepared for:

Army Air Mobility Research and Development Laboratory

July 1974

DISTRIBUTED BY:

NTIS

National Technical Information Service
U. S. DEPARTMENT OF COMMERCE
5285 Port Royal Road, Springfield Va. 22151

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

AD-783392

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: USAAMRDL-TR-74-30

2. GOVT ACCESSION NO.

3. RECIPIENT'S CATALOG NUMBER

4. TITLE (and Subtitle): DEVELOPMENT OF A METHOD FOR THE ANALYSIS OF IMPROVED HELICOPTER DESIGN CRITERIA

5. TYPE OF REPORT & PERIOD COVERED: Final

6. PERFORMING ORG. REPORT NUMBER: R-1172

7. AUTHOR(s): Ross F. Metzger, Richard C. Meier, Arved Plaks, Alex Berman

8. CONTRACT OR GRANT NUMBER(s): DAAJ02-72-C-0064

9. PERFORMING ORGANIZATION NAME AND ADDRESS: Kaman Aerospace Corporation, Old Windsor Road, Bloomfield, Conn. 06002

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: Task 1F162208AA8201

11. CONTROLLING OFFICE NAME AND ADDRESS: Eustis Directorate, U.S. Army Air Mobility R&D Laboratory, Fort Eustis, Va. 23604

12. REPORT DATE: July 1974

13. NUMBER OF PAGES: 216

14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)

15. SECURITY CLASS. (of this report): Unclassified

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report): Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number): Helicopter, Cost Effectiveness, Model

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

A method has been developed to allow the evaluation of helicopter design criteria. Analytical models have been developed which include mission analysis, performance relationships, statistical and analytical weight predictions, mission effectiveness (including payload utilization data and environmental statistics), and fixed and operational cost estimations. The models are specifically designed to determine the cost effectiveness of a

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

Block 20

"two-point design criterion". A computer program known as ZODIAC II was developed to implement these and other analytical models. A user's guide for the program and illustrative computations are presented.

PREFACE

The work presented in this report was performed by Kaman Aerospace Corporation under Contract DAAJ02-72-C-0064 (DA Task 1F162208AA8201), for the Eustis Directorate, U. S. Army Air Mobility Research and Development Laboratory, Fort Eustis, Virginia. The program was under the technical direction of Mr. H. I. MacDonald of the Technology Applications Division. The authors wish to express their appreciation to Mr. MacDonald and also to Mr. J. P. Whitman and Mr. J. P. Trant, Jr., all of the Eustis Directorate for their helpful suggestions.

This project required efforts from a number of individuals in the Engineering Departments of Kaman Aerospace. Those making major contributions are as follows: R. Metzger, A. Plaks, R. C. Meier, A. Berman, H. C. Freeman, C. P. Hardegsen, A. Rodolakis, M. J. Tarricone.

TABLE OF CONTENTS

	PAGE
PREFACE	iii
LIST OF ILLUSTRATIONS	vii
LIST OF TABLES	x
INTRODUCTION	1
ANALYTICAL MODEL	2
GENERAL ORGANIZATION	2
MISSION DEFINITIONS	7
PERFORMANCE RELATIONSHIPS	10
STATISTICAL WEIGHTS	29
ANALYTICAL WEIGHTS	30
EFFECTIVENESS	41
COST MODEL	53
COMPUTATIONAL METHOD - USER'S GUIDE TO ZODIAC II	61
PROGRAM FEATURES	61
USERS RULES	64
SYNTAX RULES FOR EXPRESSIONS	72
PROGRAM OPERATION	74
PROGRAM LIMITATIONS	75
ERROR CODES	75
SUGGESTIONS FOR THE NEW USER	76
METHOD APPLICATIONS	81
UTILITY MISSION	81
WEIGHT SENSITIVITY	88
PAYLOAD UTILIZATION EFFECTS	97
GUNSHIP	100
CRANE	100
TRANSPORT	100
OBSERVATION	106

Preceding page blank

TABLE OF CONTENTS (Continued)

PAGE

CONCLUSIONS. 108

LITERATURE CITED 109

APPENDIXES

I. ZODIAC II PROGRAM LISTING 111

II. MODEL LISTINGS. 163

LIST OF SYMBOLS. 201

LIST OF ILLUSTRATIONS

Figure		Page
1	Major Units of Model and Major Input Data.	3
2	SFC Data Compiled for Aircraft Sizing Studies.	13
3	SFC Variation at Part Power	15
4	Empirical OGE Hover Power Required.	17
5	Speed-Power-Weight-Drag Relationship for DLN = 4, S = .1	19
6	Speed-Power-Weight-Drag Relationship for DLN = 6, S = .1	20
7	Speed-Power-Weight-Drag Relationship for DLN = 8, S = .1	21
8	Speed-Power-Weight-Drag Relationship for DLN = 10, S = .1.	22
9	Minimum Power in Forward Flight and Corresponding Airspeed.	23
10	Stall Boundary as a Function of Disc Loading, Airspeed, and Drag	27
11	Drag Trends for Different Types of Helicopters	28
12	Main Transmission, Rate of Change of Weight Per Unit Change in Torque.	35
13	Main Rotor, Rate of Change of Weight Per Unit Change in Torque	40
14	Typical Altitude - Temperature Data	44
15	Combined Altitude - Temperature Data.	49

LIST OF ILLUSTRATIONS (Continued)

Figure		Page
16	Sample Utilization Data	50
17	Cost Model	54
18	Helicopter Initial Costs	57
19	Two-Point Design Gross Weights for Utility Mission	83
20	Two-Point Design Payloads for Utility Mission	85
21	Engine and Transmission Ratings for Two-Point Design Helicopters	86
22	Typical Cost Per Flight Hour Results for Utility Mission	87
23	Probability of Hover and VRC for Utility Mission	89
24	Overall Cost Effectiveness for Illustrative Utility Mission	93
25	Effect of 20% Increase in Empty Weight Model on Payload - Utility Mission	94
26	Effect of 20% Increase in Empty Weight Model on Cost Per Hour - Utility Mission	95
27	Effect of 20% Increase in Empty Weight Model on Probability of Hover - Utility Mission	96
28	Effect of 20% Increase in Empty Weight Model on Overall Cost Effectiveness of Utility Mission	98
29	Overall Cost Effectiveness for Fixed and Variable Payload for Utility Mission	99

LIST OF ILLUSTRATIONS (Continued)

Figure		Page
30	Gross Weights for Gunship Mission for 35°C	101
31	Overall Co.t Effectiveness for Gunship Mission for 35°C	102
32	Payloads and Gross Weights for the Crane	103
33	Overall Cost Effectiveness for Crane Mission	104
34	Overall Cost Effectiveness for Transport Model	105
35	Overall Cost Effectiveness for Observation Model	107

LIST OF TABLES

Table		Page
I	MISSION PROFILE	8
II	PRIMARY MISSION DEFINITIONS	11
III	INPUT DATA FOR PARAMETRIC WEIGHT ANALYSIS.	31
IV	TYPICAL PARAMETERS.	37
V	BLADE MATERIALS	38
VI	WEIGHT ANALYSIS - BLADES AND HUB.	39
VII	AREAS USED IN HOVER PROBABILITY CALCULATIONS.	46
VIII	SAMPLE COST EFFECTIVENESS CALCULATION . . .	52
IX	ERROR CODES	76

INTRODUCTION

A "single-point" design is defined as a helicopter which satisfies a specification defining a single altitude, temperature, payload combination, in addition to some prescribed mission. While it is admittedly an over-simplification to describe present helicopters as pure single-point design vehicles, this criterion is certainly a major factor in their final configuration.

A "two-point" design is defined as one which, in addition to the single-point criterion, has been sized so as to be able to make full use of the additional power, torque, and lifting capability available at some other altitude and temperature condition (usually lower and cooler).

A single-point design has the advantage of being the lightest weight vehicle capable of performing the stated mission. The single-point design vehicle will also have lower initial costs than any other vehicle capable of performing the same mission. However, since the engines in a single-point design vehicle must be capable of producing enough power to hover at high density - altitude, usually 4000 ft, 95°F, it will have power available at lower density altitudes which cannot be handled by its power transmission system and lifting capability which cannot be handled by the fuselage structure. In actual operation, the helicopter will either be performing below its engine capabilities or exceeding torque and structural limits.

The two-point design will be heavier and have higher initial costs. It will, however, be able to take advantage of (some of) its additional capability. It will be able to safely carry greater loads and will have greater endurance. Because it is less susceptible to abuse, its maintenance costs will be reduced. Depending on what is selected as the second design point, the particular operating conditions, and the appropriate effectiveness criteria, this vehicle can be significantly more cost effective.

It is the purpose of this project to develop a method for the selection of the most cost-effective second design point and to develop a better understanding of the concept and the factors which affect the selection.

ANALYTICAL MODEL

The analytical model used to describe the interrelationships between the various factors affecting the cost and the effectiveness is crucial to the success of the project. Because of the wide variety of possible models reflecting missions, vehicle concepts, technology levels, available data, and the approach preferences of the agency performing the evaluation, a rather general computer program was developed to handle models of varying complexity and organization. All the details of the program are given in later sections. However, it is necessary to point out that the program is based on the implementation of analytical models in the form of logic diagrams. The diagrams contain computational units which are called "modules". The discussion of the models developed in this study will be presented in this framework, starting from the overall logic and progressing to the details of the individual computational units.

The analytical models developed in this report are considered to be adequate representations of the five types of helicopters included in this project, i.e., (1) utility, (2) cargo, (3) crane, (4) observation, and (5) gunship. These models, however, have been developed primarily as foundations to be built upon and modified as our knowledge increases and for the analysis of specific mission requirements. The computer implementation has been specifically developed to have the capability to easily handle minor or major changes in details or overall logic of the model.

GENERAL ORGANIZATION

The model is logically separated into four major units as illustrated in Figure 1. The first block indicates the preliminary design of the baseline single-point design. The major input items here are the single altitude and temperature condition and the basic mission definition.

The second block to be entered after the single-point design has been achieved indicates the computation of the changes in the aircraft based on a specified second altitude and temperature condition.

The third block computes the cost of the helicopter including production and operating costs but excluding maintenance which will be a function of the actual missions flown.

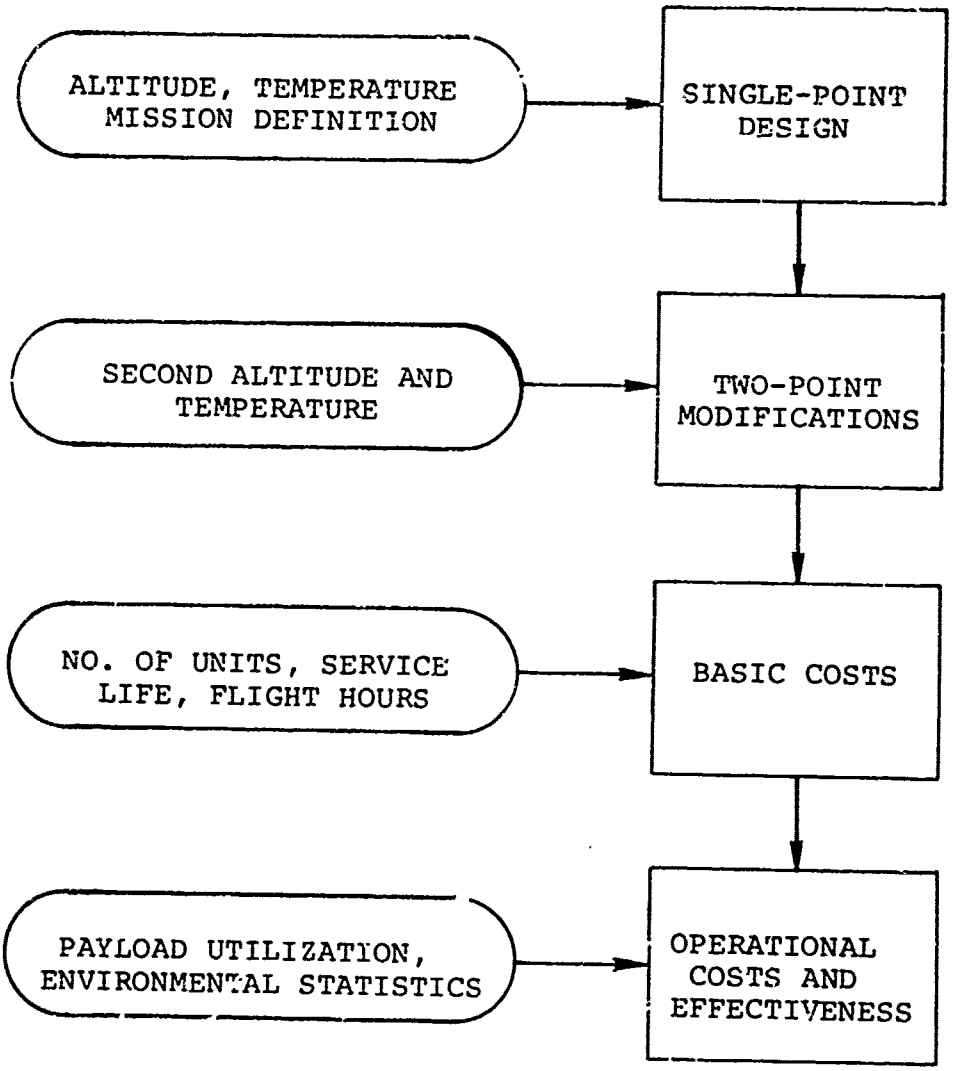


Figure 1. Major Units of Model and Major Input Data.

Primary data includes the production quantity, the attrition rate and service life and flight hours.

The last logical block makes use of payload utilization data, environmental data and results in the determination of total costs and effectiveness.

These four blocks will now be described in more detail in terms of the modules included in them. These modules represent the basic computation required. In this discussion, the function of each module will be indicated but the actual relationships used will be presented in later sections. The model described here is for a utility helicopter but is typical of all the models. When details are discussed, the differences between the models will be indicated.

Single-Point Design

The modules in the first major block of operations are computed in the following order:

SIZE
TO ALLOWANCE AND CRUISE
MAX RANGE
TOTAL FUEL
STAT WEIGHT
GROSS WEIGHT

The functions of each of these modules are given in the following paragraphs.

SIZE - This module computes the power required, based on the temperature, altitude, rate of climb, gross weight, and disc loading. In addition, such data as flat plate area, main rotor radius, and takeoff power are also computed.

TO ALLOWANCE AND CRUISE - This module computes the fuel consumed during the takeoff and cruise at V_{max} segments of the mission.

MAX RANGE - The speed for maximum range is determined, and cruise at this speed is carried out with a fuel used computation.

TOTAL FUEL - In this module the required fuel reserve is added to the two previously determined increments to obtain total fuel requirement.

STAT WEIGHT - Based on a statistical weights model appropriate to the type of helicopter being studied, the weight of each component is computed.

GROSS WEIGHT - This module simply adds the empty weight, fuel weight, crew and payload.

Notice that the gross weight is required in the first module and is recalculated in the last. In operati[illegible] gross weight is estimated and the entire process [illegible]ated until a converged gross weight is obtained. At [illegible]s point all the design parameters for the single-point design have been determined.

Two-Point Modifications

The second block of Figure 1 which determines the changes due to the additional design point contains the following modules in this order:

SIZE (Two-point data)
TO ALLOWANCE AND CRUISE
MAX RANGE
TOTAL FUEL
ANAL WEIGHT
GROSS WEIGHT TWO
TO ALLOWANCE AND CRUISE
MAX RANGE
TOTAL FUEL
PAYLOAD
ERROR

The first four modules are the same as used previously except they have as data the altitude and temperature of the two-point design.

ANAL WEIGHT - This module corrects the previously obtained weight by analytically taking into account the changes in torque and gross weight. These effects are used to modify the weight of the drive system, rotor, and fuselage.

GROSS WEIGHT TWO - The gross weight of the two-point design helicopter is obtained.

The next three modules determine the mission fuel with the new gross weight.

PAYLOAD - This module determines the payload capability at the first design point.

ERROR - The payload just determined is compared with the requirements. If the payload is deficient, the gross weight is increased by the deficiency and the entire block is repeated until convergence.

Basic Costs

The computation of the basic costs is accomplished with a single module:

PAPC COSTS

This module computes the production, attrition, crew, and miscellaneous costs on a per-hour, per-production helicopter basis.

Operational Costs and Effectiveness

PAYLOADS

OGWS
TO ALLOWANCE AND CRUISE
MAX RANGE
TOTAL FUEL

MAINT COSTS

HOVER PROB

MEI

PAYLOADS - This module refers to input tables to obtain discrete payloads and their respective utilization frequency.

OGWS - The gross weight with one of the above payloads is obtained.

The next three modules then, as previously, determine the total fuel for the standard mission with the specified payload. The process is iterated from the module OGWS until the fuel load (and gross weight) converges.

MAINT COSTS - The maintenance costs are computed based on the ratios of gross weights, computed MTBF's, and overload effects.

HOVER PROB - This module uses environmental data to determine the integrated probability that the helicopter can perform the required mission.

MEI - The mission effectiveness index and the overall cost effectiveness are computed.

MISSION DEFINITIONS

One of the objectives of this study is to develop models that have the capability to treat a mission profile consisting of a number of different types of mission segments. These are: (a) ground operation, e.g., engine start, warmup, and check-out; (b) takeoff; (c) climb; (d) cruise at given airspeed; (e) cruise at airspeed for maximum range; (f) dash at V_{max}; (g) loiter; (h) hover, e.g., loading or unloading of cargo; (i) descent; and (j) landing with a fuel reserve. The five mission profiles compiled for the five types of helicopters were compiled in such a way that all of the above specified mission segments are represented. The one exception is the descent segment, which is only implied by the altitude change between the preceding and following mission segments, but for which no calculations are performed because past specifications reviewed do not allow distance credit for the descent segment. Additional segments were defined using recent RFQ's as samples to permit construction of any recently used mission profile.

Each mission segment is calculated for a given altitude and temperature. Depending on the type of segment, time, distance or airspeed is specified. In the case of climb and loiter, the airspeed is not directly specified but is determined as the speed for minimum power. For cruise mission segments, the airspeed is either specified or calculated to meet specified criteria as will be explained later. In any case, the calculation procedure includes checks that no applicable stall or power or torque limit is exceeded.

Performance at each mission segment is calculated for the gross weight at the start of the mission segment. Weight reduction due to fuel consumption is accounted for by subtracting used fuel weight for each succeeding mission segment. Payload changes, as due to unloading or loading of cargo, rescuees, troops, or armament, may be accounted for in determination of the initial gross weight of any segment.

The drag changes due to off-loading or picking up of external cargo, or disposing of weapons during the mission can be accounted for similarly to payload changes discussed above.

Table I is a listing of segments included in the five mission profiles. The approaches to the analysis of these segments are given in the following paragraphs.

TABLE I. MISSION PROFILE

Mission Segment	TAS	Distance	Time	Power
Ground Operation	0	0	Spec.	Flight Idle
T.O.	0	0	Spec.	Spec.
Climb	Look up speed for min. power	Calc	Calc	Spec.
Cruise @ Given A/S	Spec.	Calc Spec.	Spec. Calc	Calc
Cruise @ Given A/S for Maximum Range	Calc	Calc Spec.	Spec. Calc	Calc
V-max, Dash	Calc	Calc Spec.	Calc Calc	Spec.
Loiter	Look up speed for min. power	Calc	Spec.	Calc
Hover	0	0	Spec.	Calc
Reserve	Any mission segment above or percent initial fuel or both			

Calc = Calculated Spec. = Specify

Ground operation fuel flow is determined directly from engine statistics. The engine is assumed to be at flight idle setting as would be the case for preflight checking of aircraft systems. Review of engine data showed that fuel flow is a function of engine size and pressure altitude resulting in:

$$FL = .15(PRA)(PR)(TIME)/60$$

Fuel allowance for the takeoff mission segment is calculated per military specifications as fuel used at a given power and in a given time period. The power usually specified is the maximum continuous power rating. The calculations procedure selects the lesser of the two - engine or transmission power level - for the specified rating and atmospheric conditions and determines the fuel flow for that using parametric part power SFC vs referred power variation data.

The climb segment requires that both initial and final altitudes and temperatures are specified. Power rating is specified as either intermediate or maximum continuous, and the calculation procedure determines average altitude and temperature. The engine power available at the specified rating and average atmospheric conditions is compared with the transmission limit, and the lesser of the two is used. The average fuel flow is determined now for this power. Time-to-climb determination involves determination of minimum level flight power from the appropriate table. Then,

$$R/C = \frac{33{,}000\ (\text{HP available} - \text{minimum HP required})}{\text{Gross Weight}}$$

and

$$\text{Time} = \frac{\Delta\ \text{Altitude}}{R/C}$$

With known time and fuel flow, the fuel used is now calculated.

Range for cruise at a given airspeed is determined in a very straightforward manner. Power is determined at the specified airspeed from appropriate parametric power required tables, for which fuel flow may now be determined using part power SFC data. Either time or distance may be specified and the other is calculated. The fuel used is simply the fuel flow times the time.

Cruise at airspeed for maximum range is similar to the preceding segment except that the airspeed is first determined by a procedure where specific air range is calculated over a range of airspeeds. Peak value is determined, and the corresponding airspeed is then compared to stall, transmission or power limited airspeed. The least of these becomes the cruise speed for maximum range and is used for subsequent calculations, as described above.

Range at maximum airspeed is specified for either intermediate or maximum continuous power rating. If intermediate power is specified, then the airspeed is defined as the dash airspeed. In either case, engine power available under the given atmospheric conditions is checked against transmission limit, and the smaller of these is used for airspeed determination. [illegible] airspeed is checked so that it does not exceed stall limi[illegible] airspeed. Power, the fuel flow, and fuel load are calcul[illegible]d as described before.

For all of the range segments, the distance may be specified as the total required distance less distance covered during the preceding climb segment, if any. This accounts for these range or radius missions where mission profile requires "climb on course to cruise altitude...".

Loiter airspeed and power are determined from tables of performance at minimum power in forward flight. Determination of resulting fuel flow and fuel is straightforward.

Hover is determined using the parametric hover power equation. Fuel flow and fuel load determination is straightforward.

Reserve may be specified as a percentage of initial fuel or it may be specified as any other mission segment; i.e., a specified time or distance at some specified airspeed or power level. Most commonly used percentage is 10 percent.

Table II lists the primary mission (first point) definitions used in this study. The intention is to cover as many types of segments as possible and to specify representative missions.

PERFORMANCE RELATIONSHIPS

The performance and weight models consist of relationships to permit specified requirements to be combined with a given technology level to result in defining aircraft general characteristics. The requirements include ability to hover or to have a maximum speed capability, ability to fly specified mission profiles with a specified payload. Technology level is defined by characteristics of helicopter components that will be attainable at a given time, such as component weight/size relationships, engine SFC characteristics, aircraft drag, power-weight-speed relationships, etc. The application of the performance and weight models results in characteristics such as rotor dimensions, weight buildup and engine ratings.

TABLE II. PRIMARY MISSION DEFINITIONS

	Pressure Altitude (ft)	OAT (°C)	TAS (kn)	Time (min)	Distance (n mi)	Change in Payload (lb)	Change in Drag (ft^2)
UTILITY - PL = 2640 lb							
Ground Operation	4000	35	0	8	0		
V_{max}, MCP	4000	35		20			
Best Cruise Speed	4000	35		80			
Reserve: Best Cruise	4000	35		30			
GUN SHIP - PL = 2000 lb							
Cruise @ Given A/S	4000	35	100	38			
Hover	4000	35	0	32	0		
V_{max}, MCP	4000	35		8		-1340*	-5
Cruise @ Best Cruise A/S	4000	35		6			
Reserve: Cruise @ Best Cruise A/S	4000	35		30			
CRANE - PL = 45,000 lb							
Ground Operation	0	35	0	10	0		
T.O. Allowance (MCP)	0	35	0	2	0		
Hover	0	35	0	10	0		
Max. Range Cruise	0	35			50		
Hover w/o Cargo	0	35	0	10	0	-45,000	-100
Max. Range Cruise	0	35			50		
Reserve: Max. Range Cruise	0	35		30			
OBSERVATION - PL = 300 lb							
T.O. Allowance (MCP)	SL	15	0	3	0	0	
Endurance (Loiter)	SL	15		180		0	
Reserve - 10% of Initial Fuel							
TRANSPORT - PL = 25,000 lb							
T.O. Allowance (MCP)	SL	15	0	3	0	0	
Climb							
Cruise (MCP)	7000	1			100		
(Descend)				0	0		
Ground Operation	SL	15	0	5	0	-25,000	
Climb							
Cruise (Best Range)	7000	1			100		
Reserve - 10% of Initial Fuel							

* Expendable Ordnance

In this section, performance related technology levels will be discussed. It should be noted that in some cases the values chosen only serve as samples to illustrate the methodology for determining an optimum two-point design and do not necessarily represent the actual values.

Engine Performance

The performance of a "rubberized" turboshaft engine is characterized by how power available varies with atmospheric conditions and how specific fuel consumption is related to rated power and to part power condition.

Turboshaft engine rated power available varies with temperature, altitude and forward speed. The last effect, ram, is ignored as negligible for the relatively low-speed regime of helicopters treated in this study. The effect of atmospheric condition was determined empirically from current engine data as:

$$PA/PRA = PR[1 - 2.08(TR - 1)]$$

where PRA is the highest power rating given at 15°C, SL and V = 0, which in this study was equated to the 30-minute, intermediate power rating. Similarly, the ratio of maximum continuous power to the highest engine rating was determined empirically from current engine data resulting in:

$$P/PA = RA = .9$$

In this study the fuel consumption rate is determined for the specified power level using the "rubberized" SFC versus power curve keyed to an SFC at rated power. The SFC at rated power may vary from .55 to .7 for current engines for .36 to .49 for 1980's engines. This data is presented in Figure 2.

For this study the current engine technology data was put into an equation form as:

$$SFO = 1.136(PRA/NEN)^{-.105}$$

where NEN is the number of engines. Thus, PRA/NEN is the rating of one engine. SFO applies to the highest power rating at SL, 15°C, V = 0.

For advanced engines for 1980 and beyond, a 20-percent reduction of SFL is projected.

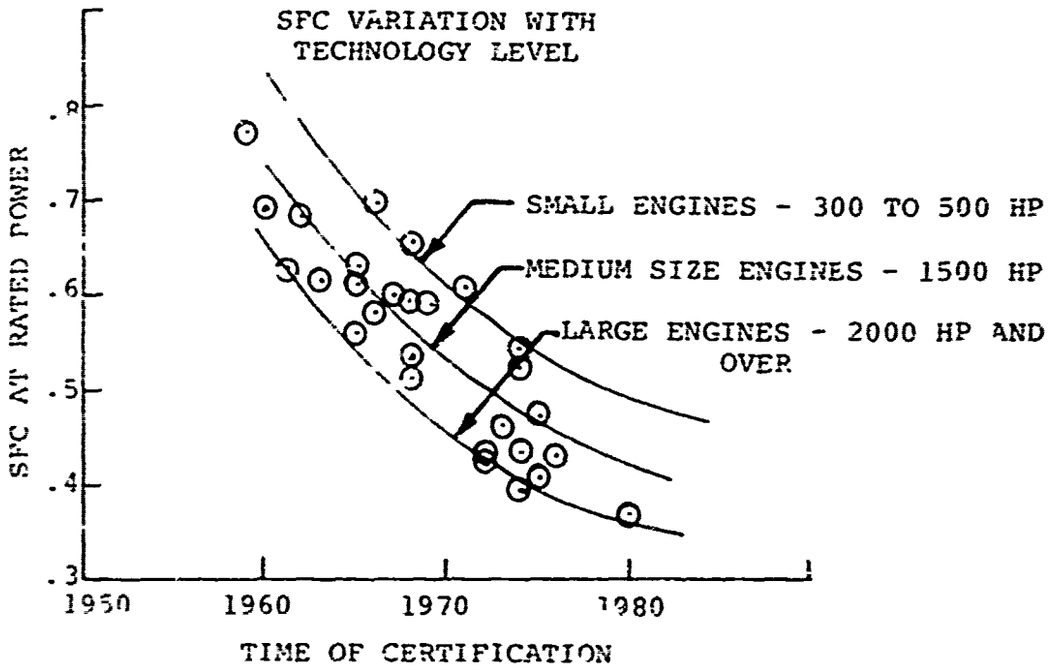


Figure 2. SFC Data Compiled for Aircraft Sizing Studies.

The SFC at rated power is determined by technology level, engine size, tradeoffs of engine simplicity versus engine efficiency, and development history of the engine. The smaller engines have higher SFC than bigger engines because bigger engines can be designed with lower percentage losses. Simplicity, such as the use of a fixed shaft, is achieved at the expense of efficiency. During the development cycle of an engine, SFC usually improves; however, growth usually occurs to obtain more power within a given engine size envelope, and SFC improvement is considered of secondary importance.

The technology level is identified mainly by turbine inlet temperatures used and pressure ratios employed. These factors seem to go up hand in hand and result in lower SFC's and lower engine specific weight. The temperatures with current values from 1600° to 2000°F will increase to 2400°F in the 1980-1990 period requiring advanced materials and turbine cooling in various degrees. Pressure ratios similarly will increase from the current 6:1 to 14:1 range to up to 20:1.

In treating engine statistical data, the rated power was selected as the highest thermodynamic, i.e., turbine inlet temperature limited, rating given, excluding any "emergency" ratings. Any derating due to gearbox limits was ignored. The highest, i.e., T.O. or maximum (10 minutes), ratings were used, if given, instead of the 30-minute limits, assuming that the time ratings only express limits imposed to achieve certain TBO. In effect, it was assumed that given engine would have the same SFC at the highest rating regardless whether it is a 5-, 10-, or 30-minute rating.

For fuel consumption determination at part power operation, the SFC versus referred power was "rubberized" by generalizing it in terms of SFC and power at rated conditions, i.e., SFO and PRA resulting in SFC/SFO versus PRF/PRA. From the study, two shapes appeared as shown in Figure 3. The flatter of the two curves appeared associated with some of the proposed (mostly paper) engines, indicating a possible new trend in turbine design optimization in which the engine manufacturer's seem to take into account the fact that a good portion of the time engines are operated at part power and that it is here where low SFC's pay off. However, it has been noted that some prototype engines, which were initially designed to have this flat trend, seem to have moved back toward the current trend, throwing some question as to the achievability (or the will to achieve) of the projected flat trends. Either curve may be used to determine SFC at any power level once rated power and the corresponding SFC have been selected.

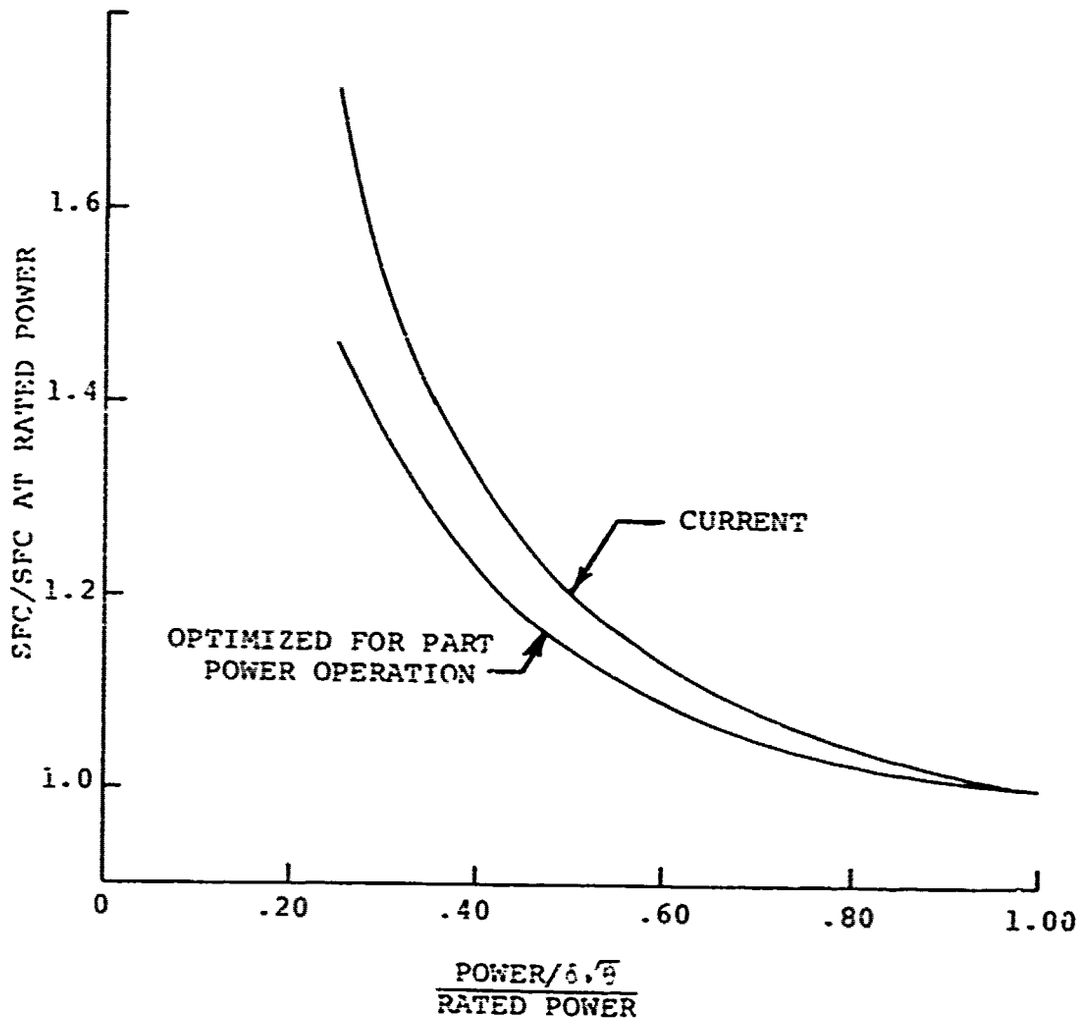


Figure 3. SFC Variation at Part Power.

The ram effect due to forward flight reduces the SFC slightly. Due to the relatively low forward speed of the helicopters studied, the ram may be ignored. The expression by which fuel consumption can be calculated is then:

$$W_{FL} = (P)(SFO)(SSF)KFF$$

where P is the given power at the desired airspeed, SFO is the SFC at rated power determined as a function of technology level and engine size, SSF is the SFC increase factor for part power operation determined as a function of PRF/PRA and KFF is any loss or allowance factor applied. Since in this study relative fuel loads are sought, application of a loss or allowance would not affect the conclusions of the study, and thus, for simplicity, none are applied.

Hover and Vertical Flight

A substantial body of statistics exists for various helicopters which establish a power required. Blockage losses, drive system losses, tail rotor and accessories power were not extracted from the statistics. Thus, the variables are gross weight and (total) power at the engine output shaft. Data form is GW/SHP vs DL/DR as is shown in Figure 4.

This is consistent with the traditional $C_P/\sigma = f(C_T/\sigma, \mu)$ presentation if the latter is expanded for a specified R and DL. The curve may be expressed in equation form as $SHP = (.051)(GW)(DL/DR)^{.41}$ at a specified rate, VRC.

Power to climb without forward speed can be calculated as power required to hover OGE plus an increment for the potential energy change. Assuming a climb efficiency of 1.25 results in a hover power adjustment as:

$$\Delta SHP = (VRC)(GW)/(33000)(1.25) = .00002424\ (VRC)(GW)$$

If it should be desirable to size the engine or transmission to meet a given vertical rate of climb requirement with a specified margin of power, then the above may be combined to result in:

$$SHP = [(.051)(GW)(DL/DR)^{.41} + \Delta SHP]/(1 - PM/100)$$

where PM is the desired power margin in percent.

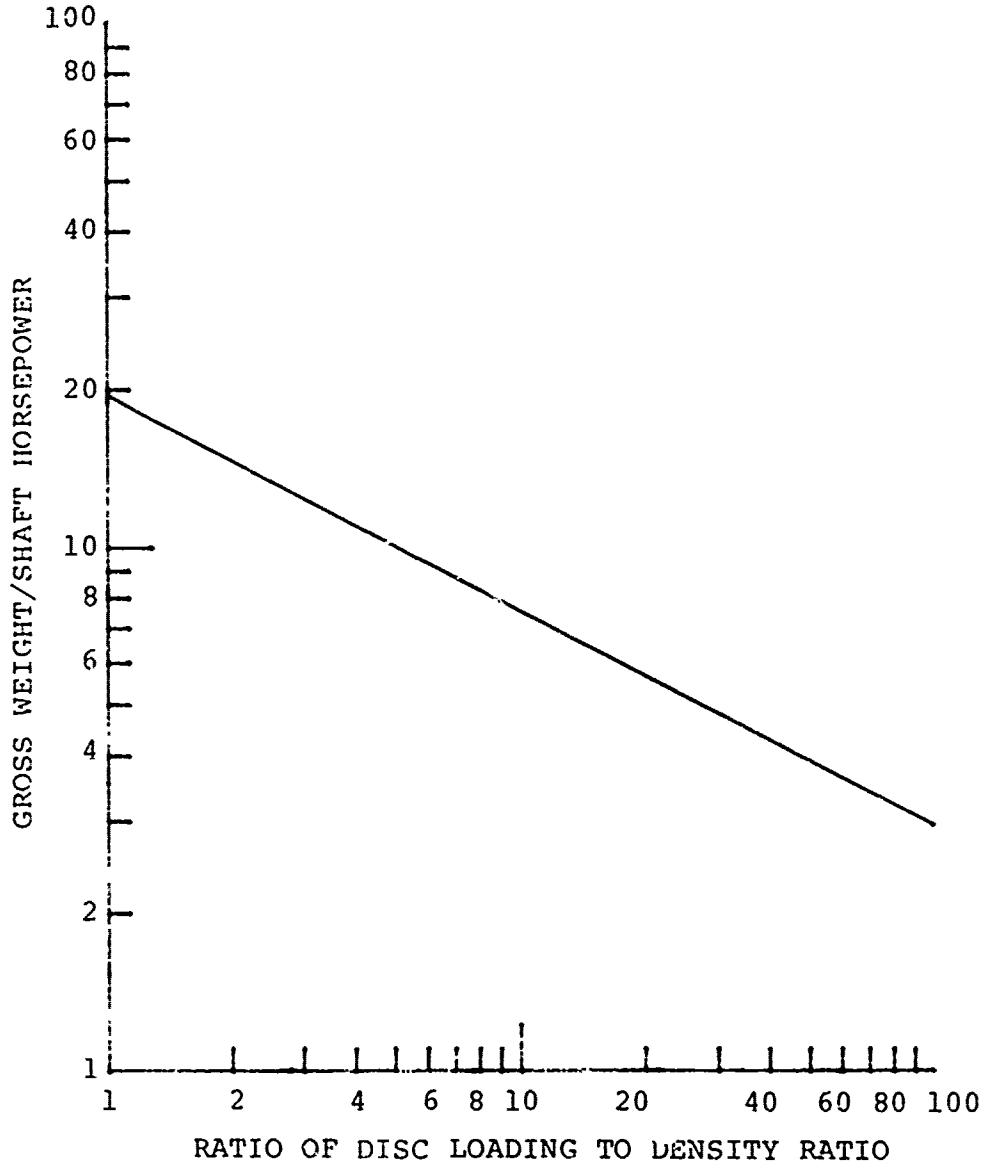


Figure 4. Empirical OGE Hover Power Required.

Speed-Power

The aircraft speed-power-weight-drag relationship is a function of rotor characteristics, such as airfoil sections used, taper, twist, disc and blade loadings (and thus solidity), tip speed schedule (and thus tip Mach number), etc.

For this study, data were generated for the following characteristics:

Airfoil:	NACA 23012 (constant)
Number of Blades:	4
Solidity, S	0.1
Tip Speed	700 fps (constant)
Taper	None

The airfoil selected is for a current state-of-the-art airfoil with good stall characteristics. The drag-lift characteristics are well established throughout the Mach number range. The other listed characteristics were chosen as typical for helicopters currently in use or in development. The data were obtained by a standard rotor performance program involving an iterative numerical solution of the blade flapping equations of motion.

The results are used in the model in a parametric form as illustrated in Figures 5-8. Data are entered into the program in two ways to allow determination of airspeed for a given power and determination of power at a given airspeed.

$$TAS = f(DLN, POW, FOW)$$

$$POW = f(DLN, TAS, FOW)$$

These tables apply for speeds above speed for minimum power with accuracy improving as speed is increasing. Accuracy of the tables reduces at reduced airspeeds due to the shallowness of the slope of power vs airspeed. For this reason, separate data, as shown in Figure 9, are entered for speeds for minimum power to accommodate performance related to this speed such as loiter and maximum rate of climb. This data again are in two forms:

$$TAS = f(DLN, FOW)$$

$$MRHP = f(DLN, FOW)$$

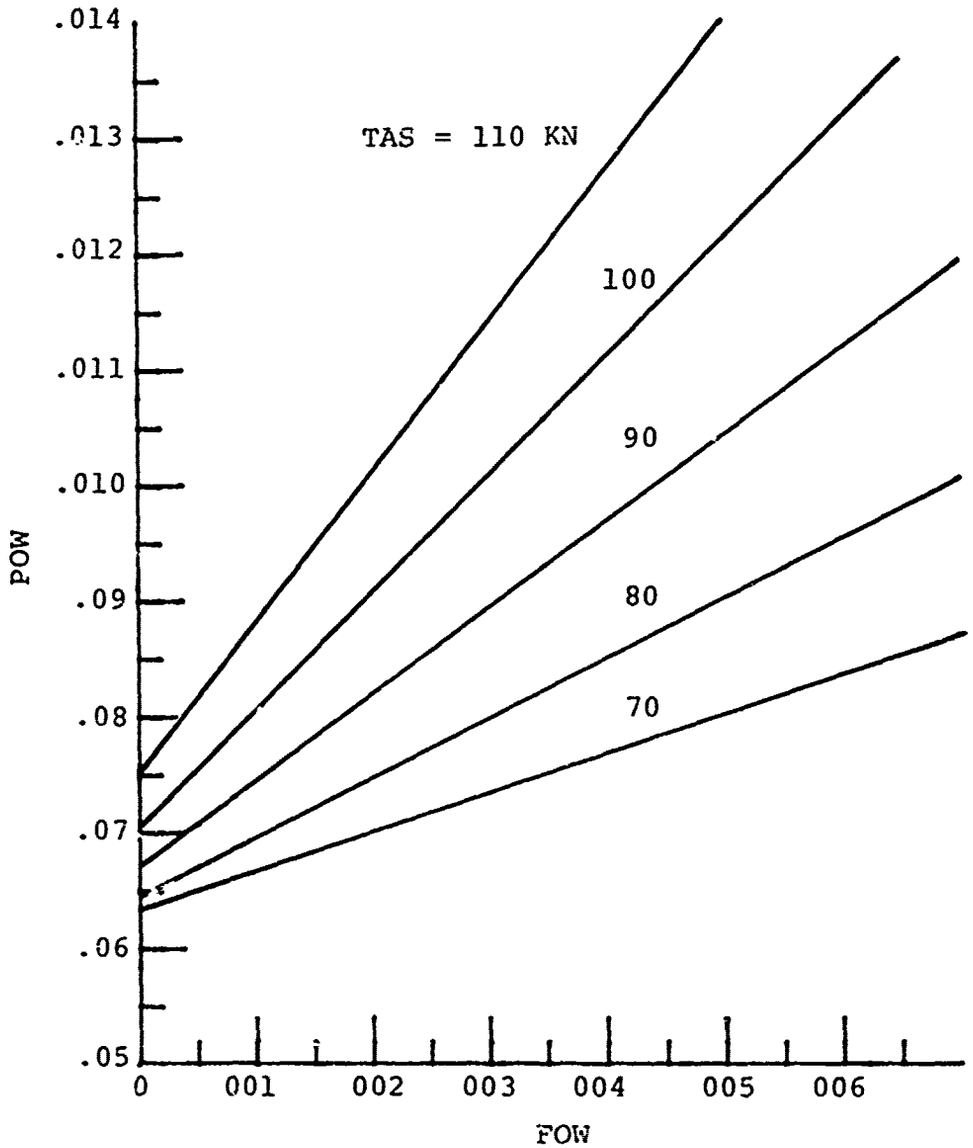


Figure 5. Speed-Power-Weight-Drag Relationship for DLN = 4, S = .1.

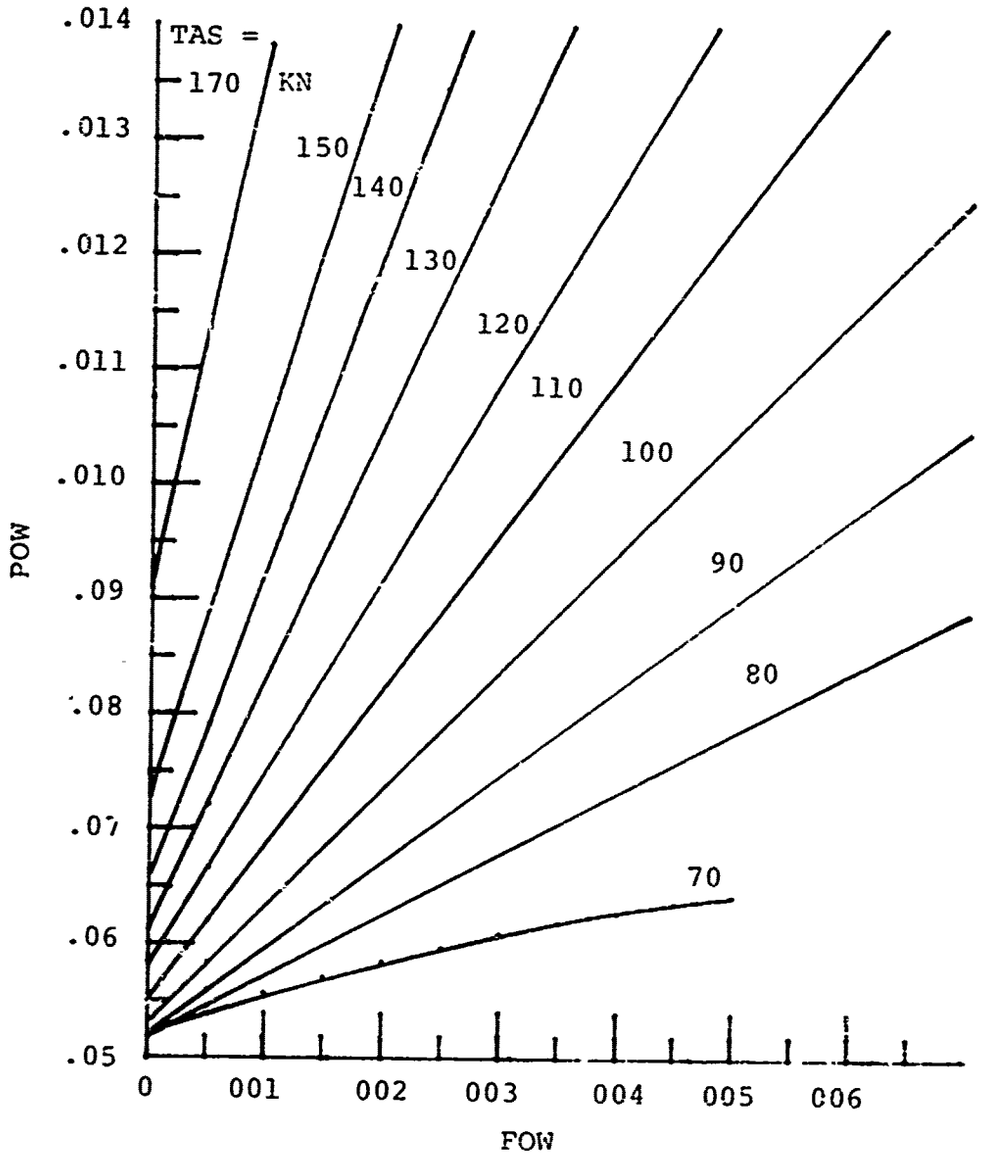


Figure 6. Speed-Power-Weight-Drag Relationship for DLN = 6, S = .1.

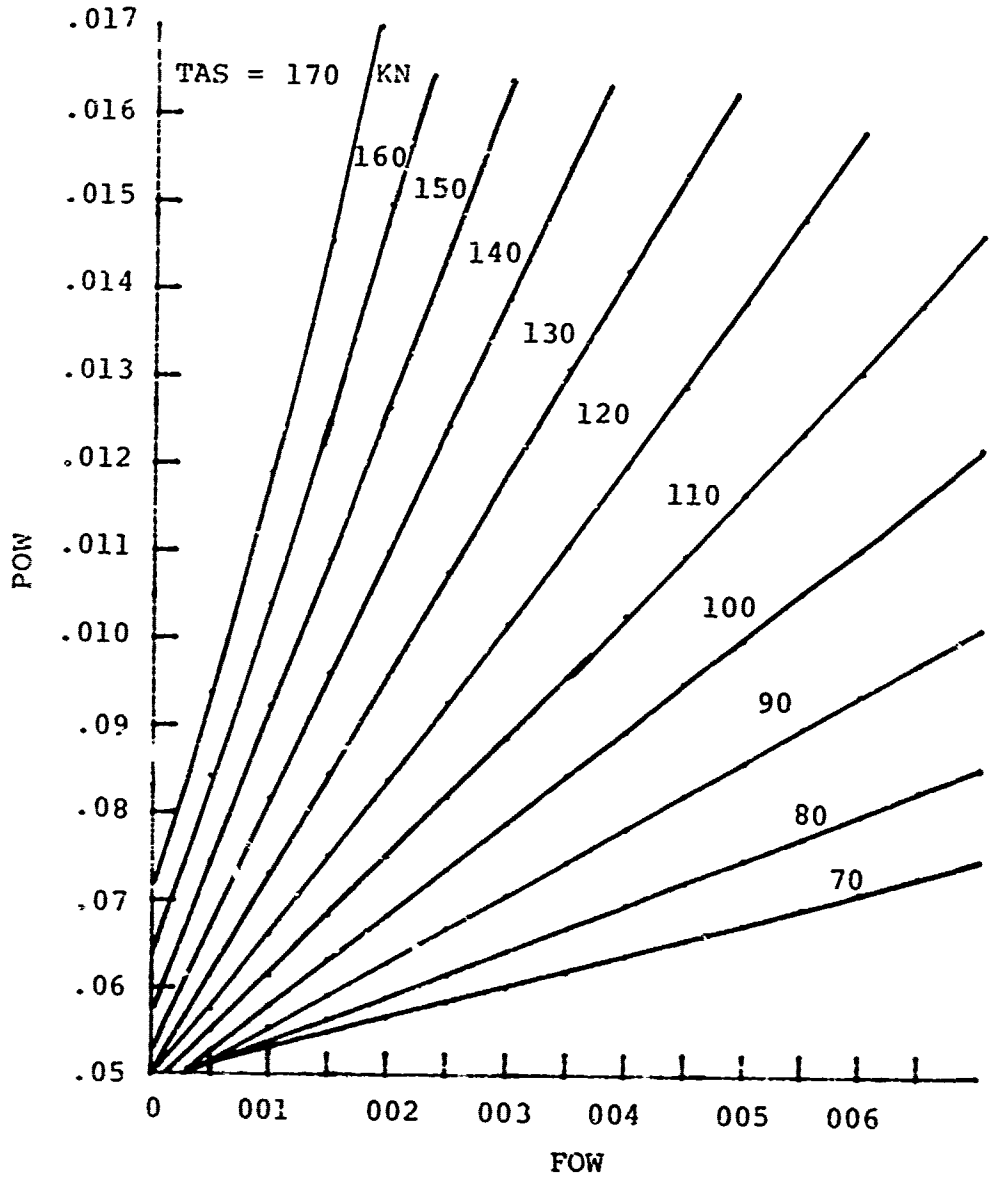


Figure 7. Speed-Power-Weight-Drag Relationship for DLN = 8, S = .1.

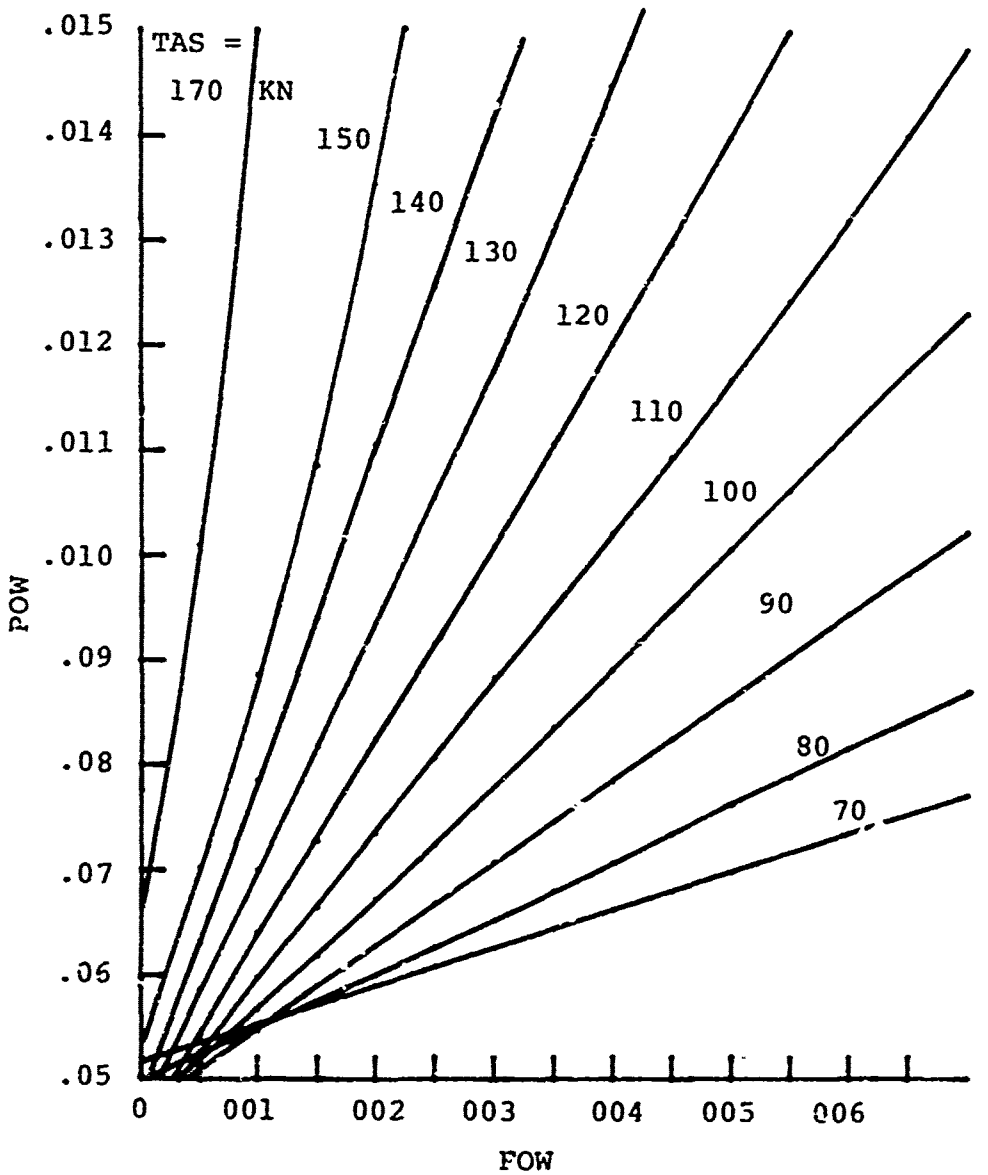


Figure 8. Speed-Power-Weight-Drag Relationship for DLN = 10, S = .1.

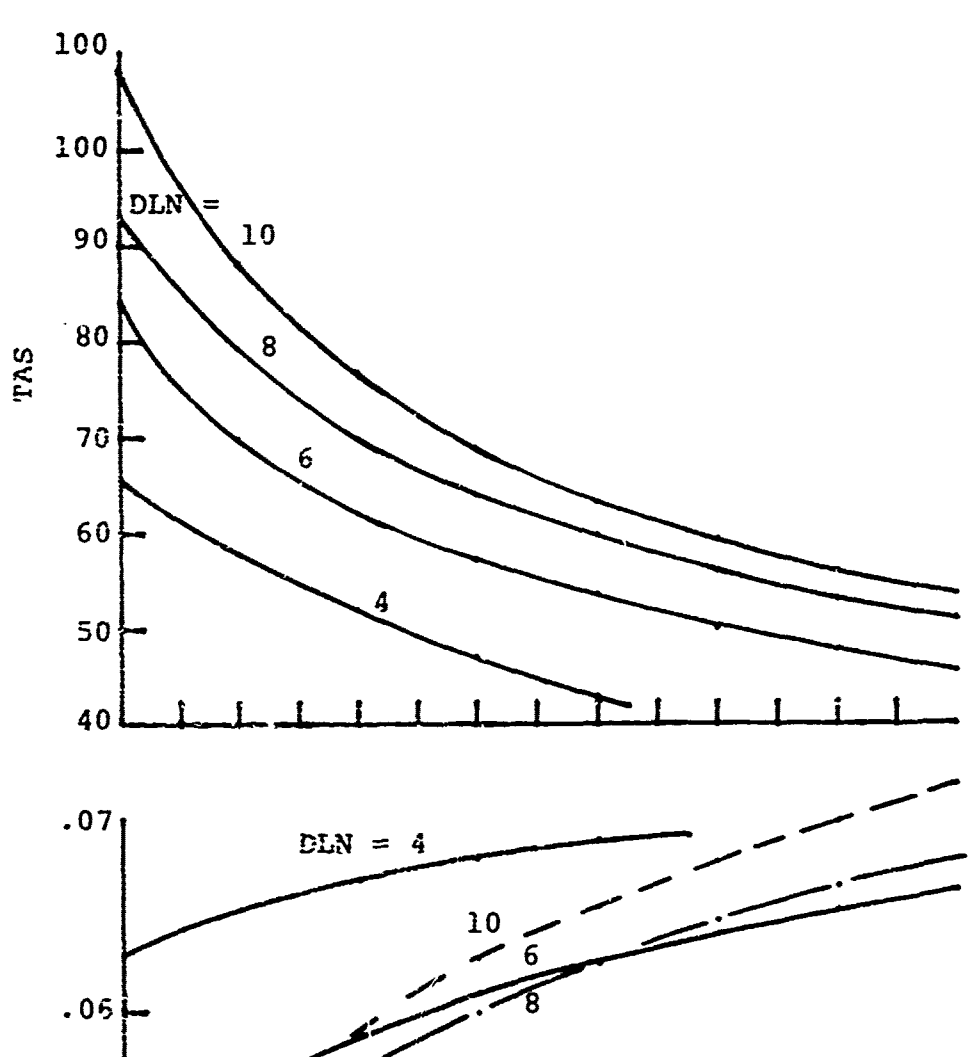


Figure 9. Minimum Power in Forward Flight and Corresponding Airspeed.

The parametric form used in the table is the result of determining the most meaningful way of presenting the power-speed-lift-drag relationship for a given rotor geometry. The early forms of C_p - C_T - μ types of presentation have lost some of their usefulness ever since Mach number effects have been taken into account. The form used, however, is consistent with C_p/σ - C_x/σ - C_L/σ - $V/\Omega R$ - σ - M relationship and has the advantage over the latter that it is closer to the dimensional form. In this semi-dimensional form, rapid comparisons can be made of data used with other estimates or test data and corrections made incrementally or percentage-wise as desired.

The relationship between the traditional coefficient type formulation and the used semi-dimensional form is illustrated by the following. C_p/σ is defined when C_x/σ, C_L/σ, σ, $V/\Omega R$ and M are given.

There is no increase in number of variables to define a condition if instead of $V/\Omega R$ and M, one would specify V and ΩR and assume a nominal speed of sound to relate these as

$$\Omega R = Mc/(1 + V/\Omega R)$$

$$V = (V/\Omega R)\Omega R$$

The remaining coefficients can be rewritten as:

$$POW = \frac{(C_p/\sigma)A\sigma\rho(\Omega R)^3/550}{(C_L/\sigma)A\sigma\rho(\Omega R)^2} = \frac{C_p/\sigma}{C_L/\sigma}\frac{\Omega R}{550}$$

$$q = .5\rho V^2 = .5(DR)\rho_o(V/\Omega R)^2(\Omega R)^2$$

$$FOW = \frac{F_x(DR)}{GWq} = \frac{(C_x/\sigma)A\sigma\rho(\Omega R)^2(DR)}{(C_L/\sigma)A\sigma\rho(\Omega R)^2 .5(DR)\rho_o(V/\Omega R)^2(\Omega R)^2}$$

$$= \frac{C_x/\sigma}{C_L/\sigma}\frac{}{.5\rho_o(V/\Omega R)^2(\Omega R)^2}$$

$$DLN = \frac{GW}{(A)(DR)} = \frac{(C_L/\sigma)A(DR)\rho_o(\Omega R)^2}{(A)(DR)} = (C_L/\sigma)\cdot\rho_o(\Omega R)^2$$

In the above, the use of a nominal speed of sound may, for some off-design condition, introduce a small error. With a 20°C change in the temperature, a 3-1/2-percent error in resulting Mach number is introduced. This, however, should be an acceptable penalty for a study such as this, and thus the data presented is generated for C = 1117 fps only.

Variation of solidity does not require separate charts but can be accommodated similarly to Reference 1. This reference substantiates the equivalence of a drag change to inflow change resulting from a solidity change. In this study the data was compiled for S = .1; performance for any other S can be obtained by entering the data with an adjusted equivalent flat plate drag area:

$$F = F_{actual} + \frac{(GW)(BL)(S - .1)}{4q^2}$$

The adjusted F is now used to calculate FOW as before. POW is determined from the data at the desired normalized blade loading BLN. The chart values in the presentation of this report must be used as DLN/S. For example, values shown for DLN = 6 lb/ft^2 and S = .1 are really for a BLN = 6/.1 = 60 lb/ft^2.

The tip speed in this study was chosen as 700 fps representing the current state-of-art average. The outcome of this study will not be significantly different if run at any other tip speed. However, if it should be desired to repeat the study for any other tip speed, new power required must be calculated. From this, either a new data table must be generated or a suitable adjustment found for the existing data.

Geometry of the rotor blade naturally affects the power levels and is an inherent part of any performance presentation. The characteristics used here, such as airfoil and twist, are current technology and should yield representative results. However, as in the case of tip speed, should it be desired to investigate some different geometry, then corresponding power data must be calculated, from which a new table can be generated or a suitable adjustment found for the data now used.

The tables yield main rotor power. To obtain total power, i.e., power at the engine output, a factor was determined which includes the effects of tail rotor power, accessories power, and drive system losses. The factor applies only to forward flight and is:

$$MRHP/SHP = KPP = .9174$$

Stall

The ultimate limiting factor on airspeed is stall of the retreating blade. For the purposes of this study, the stall boundary was determined consistent with the data generated for forward flight. Thus, they contain the same assumptions as the power required. Stall was defined at the airspeed when the angle of attack reached 12 degrees at the outboard section of the blade. The angle was found to be just below the stall caused break in power vs lift curves. The data is shown in Figure 10. The data are entered into the program in the form of an equation:

$$TAS = 70 + 15.7(15 - DLN)/(1000\ POW)^{.2598}$$

For operational use, helicopter maximum airspeeds are limited to airspeeds below stall. The margin, SM, specified for this study is 10 knots.

Drag

Drag of aircraft is most conveniently represented as a function of gross weight to the two-thirds power, a relationship which assumes that a characteristic dimension exists which, when cubed, gives a volume proportional to gross weight, and when squared, gives an area proportional to the equivalent flat plate drag area. Statistics from various current and projected helicopters have led to the trends shown on Figure 11.

The statistical drag data is treated as a function of maximum gross weight of the helicopter. There is no assurance that helicopter cargo spaces are designed to carry the maximum structural weight limited payload. In fact, for many helicopters the maximum gross weight has grown by a third, without a corresponding increase in cargo space. However, since the cargo-containing fuselage, as a rule, makes up less than a quarter of the total drag, the cargo space is not a significant factor. It was felt that the total drag is more related to the maximum gross weight of the aircraft which dictates rotor size so that gross weight adequately represents the aircraft. In the case of helicopters with considerable development and growth behind them, the latest gross weight was used for correlation purposes. Then, helicopters at their maximum GW exhibit higher disc loadings than originally designed for. Thus, the statistics reflect the tendency for new aircraft to be designed for higher disc and power loadings.

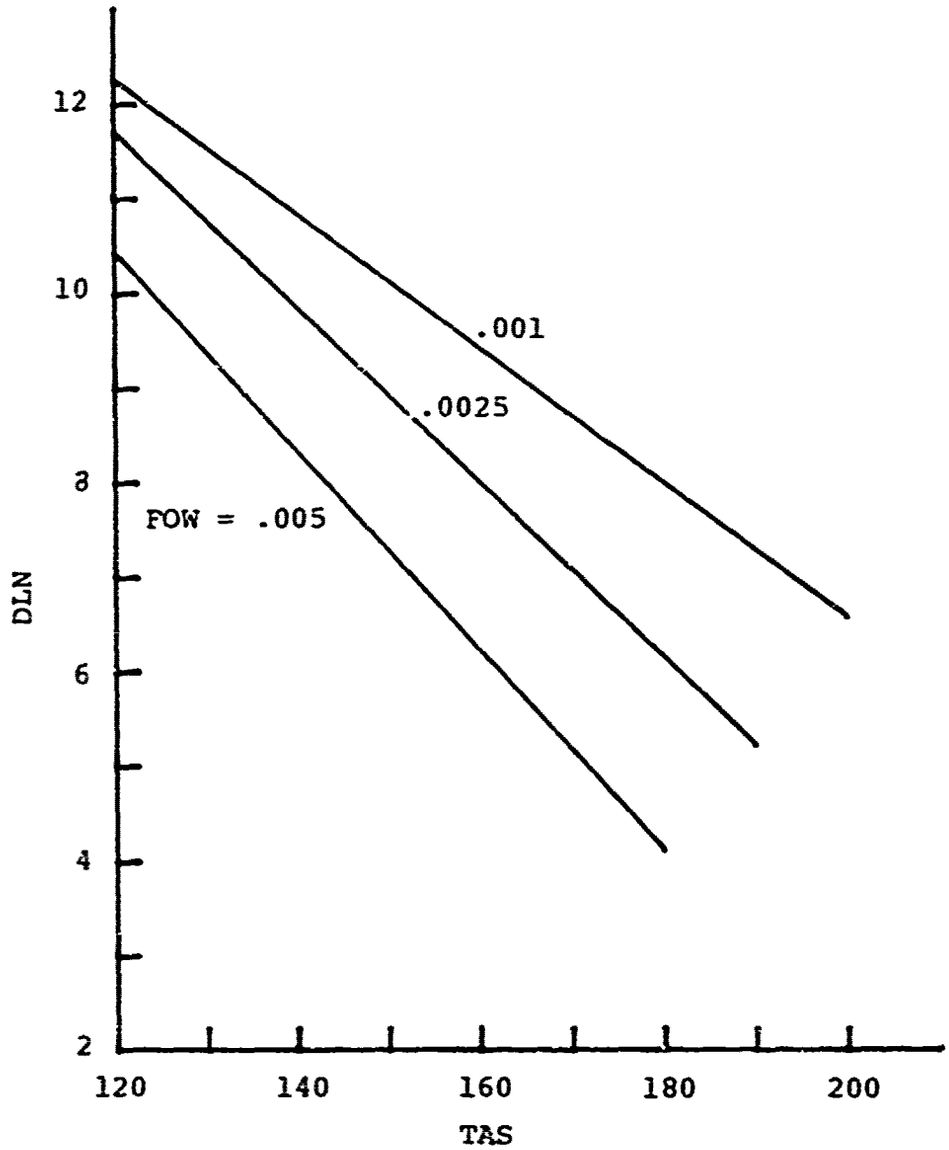


Figure 10. Stall Boundary as a Function of Disc Loading, Airspeed, and Drag.

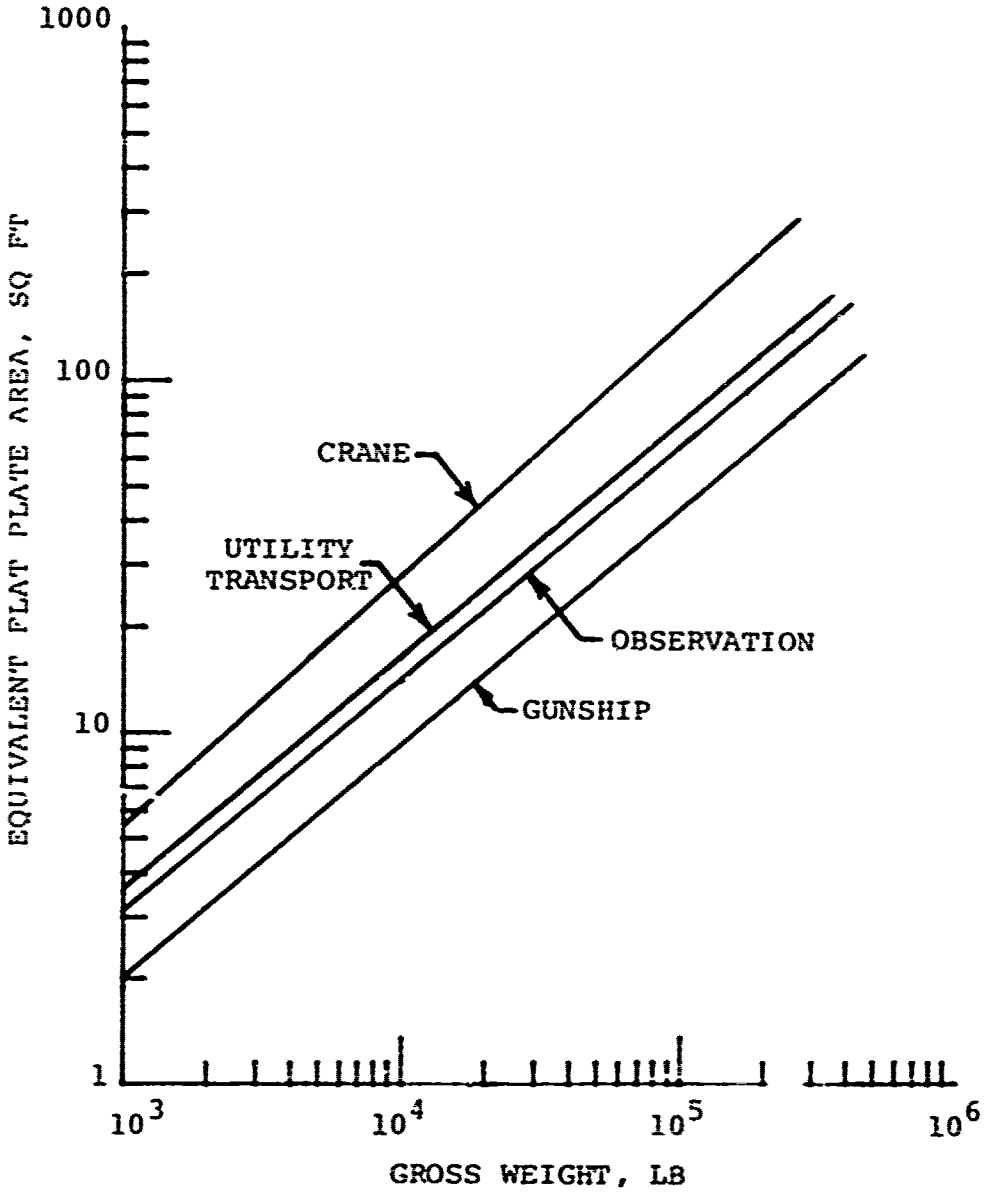


Figure 11. Drag Trends for Different Types of Helicopters.

Depending on the type of helicopter, the loading density varies and thus the volume-gross weight relationship. Therefore, different fairings were drawn for the various types of helicopters.

These expressions cover only the basic aircraft without external stores or cargo. These items have a characteristic of themselves and are treated as an add-on to the basic drag. In this study, it is assumed that all fuel is carried internally, i.e., no external fuel tanks are used. Furthermore, it is assumed that no external stores are carried by the observation-utility-transport categories of aircraft. The weapons drag of the gunship is assumed at an equivalent flat plate drag area of 5 ft^2, and the cargo drag area for cranes is assumed to be 100 ft^2. If desired by the user of the program, these add-ons can also be rubberized by use of suitable statistics.

The above resulted in the following relationships to be used in this study.

Utility, transport	$f = .035\ GW^{2/3}$
Observation	$f = .03\ GW^{2/3}$
Gunship	$f = .02\ GW^{2/3} + 5$
Crane	$f = .07\ GW^{2/3} + 100$

STATISTICAL WEIGHTS

The statistical weight model used for the estimation of conventional (single point) designs was developed to represent the five types of helicopters under consideration and to use relatively limited data such as would be available during a typical predesign situation. This model is a further development of that given and fully documented in Reference 2. The statistical trends were based on analyses of the following helicopters:

CH-3C	HH-2D
CH-34	HH-52
CH-37	UH-2A/B
CH-53A	UH-19
SH-3A	UH-1B
S-52	UH-1D
S-61	UH-1N

The equations are listed in the STAT WEIGHT module and are not repeated here. There are three types of data used in the parametric equations, as follows:

Basic Input Data - These data either define characteristics of the vehicle or are a result of performance calculations. Examples are: Number of engines and rotor radius. The required input data are listed in Table III.

Intermediate Data - Actual values may be used if known from design definition. If an actual value is not available, an intermediate data value can be calculated using input data and the appropriate equations. An example is Tail Rotor Radius. If not known from design definition, the tail rotor radius (in feet) is estimated, using input data only, by .087 (rotor radius)$^{1.22}$.

Calculated Weight Data - The weight calculated by a weight equation is used in a subsequent weight equation. An example is Blade Weight, which is subsequently used to calculate Hub Weight.

ANALYTICAL WEIGHTS

The statistical weight model is appropriate for the single-point baseline vehicle. The modifications resulting from the additional design point will result in changes in the structure, drive system, and rotor. These changes are due to the changes in gross weight and engine and rotor torque. The other major elements (engine, fuel system, and fixed weights) of the vehicle are assumed to change only insignificantly since the basic mission is unchanged.

Structure

The various elements of the structure of a typical helicopter were examined from the point of view of whether the weight of the element was significant and what kinds of loads were the major design consideration. It is assumed that limit load factors, sink speed, and crash criteria are not changed with the changes in gross weight due to the second design point criteria.

The elements of the structure considered include the forward fuselage, center fuselage, tail cone, vertical tail (tail rotor pylon), horizontal tail, landing gear, engine and transmission mounts and carry through structure. In general, the items which contribute significant weight can be lumped together as the fuselage and tail and the landing gear. Some of the components are designed by steady flight loads, some by vibratory loads, some by landing loads, some by cargo loads, etc. The steady loads can be expected to be proportional to the gross weight. Landing loads also are proportional to gross weight. The vibratory loads are the

TABLE III. INPUT DATA FOR PARAMETRIC WEIGHT ANALYSIS

Symbol		Data
AG	=	Number of Auxiliary Landing Gears
BF	=	Blade Folding Option (= 1 if used, = 0 otherwise)
BRK	=	Main Rotor Brake Option (= 1 if used, = 0 otherwise)
CAP	=	Gallons of Fuel - Gal.
CB	=	Blade Chord - Feet
EDS	=	Engine Drive Shaft Option (= 1 if used, = 0 otherwise)
EN	=	Number of Engines
HP1	=	Rotor Horsepower - hp
HP2	=	Installed Horsepower - hp
ITR	=	Intermediate Tail Rotor Gearbox Option (= 1 if used, = 0 otherwise)
KLG	=	Landing Gear Geometry - Values/Configuration .0157 - Skid Gear .0247 - Sponson Mounted .0280 - Quadricycle .0329 - Tricycle - Fuselage Mounted .0405 - Crane - Straddle Type
KNAC	=	Nacelle Arrangement - Values/Configuration .96 - Twin Engines Mounted to Transmission Forward or Aft of Main Rotor 1.19 - Single Engine Mounted to Fuselage Forward or Aft of Main Rotor 1.23 - Twin Engines With Combining Gearbox 2.26 - Twin Engines Outboard of Main Fuselage Add Factors for More Than Two Engines
MOW	=	Maximum Operating Weight - Lb
NMR	=	Number of Main Rotor Blades

TABLE III - Continued

Symbol		Data
NR	=	Number of Main Rotors
NULT	=	Ultimate Load Factor
P	=	Number of Passengers
RM	=	Main Rotor Radius - Feet
S	=	Main Rotor Solidity
TAF	=	Type of Aft Fuselage - Values/Configuration 8 - Full Fuselage Depth at Splice of Main Fuselage to Aft Fuselage. Example: SH3A 9 - Tailboom Configured for Rear Ramp. Example: CH53 10 - Tailboom Without Rear Ramp. Example: UH19 13 - Full Fuselage Depth at Splice of Main Fuselage to Aft Fuselage and With a Tail Wheel Full Aft. Example: HH2D 15 - Tailcone Upswept From Fuselage Splice. Example: UH1D
SW	=	Total Wing Area
TAG	=	Type of Auxiliary Gear - Values/Configuration 0 - Observation 0 - Gunship 1.0 - Utility 2.5 - Transport 15.5 - Crane
TAR	=	Armament Provision and Plating - Values/Configuration 600 - Gunship (0 otherwise)
TEL	=	Type of Electronics - Values/Configuration (Depends on A/C Designation) .42 - Observation .75 - Crane 1.00 - Utility 1.16 - Transport 1.25 - Gunship

TABLE III - Continued		
Symbol		Data
TPU	=	Auxiliary Power Unit Option (= 1 if used, 0 otherwise)
TPY	=	Type of Pylon Configuration - Values/Configuration (Depends on Type of Aft Fuselage) 14 - Tailcone Upswept From Fuselage Splice. Example: UH1D 25 - Tailboom Without Rear Ramp. Example: UH19 45 - Tailboom Configured for Rear Ramp. Example: CH53 48 - Full Fuselage Depth at Splice of Main Fuselage to Aft Fuselage. Example: SH3A 62 - Full Fuselage Depth at Splice of Main Fuselage to Aft Fuselage and With a Tail Wheel Full Aft. Example: HH2D
VM	=	Main Rotor Tip Speed - FPS
WPL	=	Desired Weight of Payload

most difficult to generalize upon. The transition flight regime is when the highest vibratory loads can be usually expected. Experience indicates that the vibratory loads are proportional to GW, increasing by about 4 to 8 times the ratio of delta GW to GW.

Simple models of typical structures have indicated that the weight of the structure will be approximately proportional to ratios of the design loads. This is, of course, most true when relatively small changes are considered.

The types of loads which design the separate components will vary between types of helicopters and even between different helicopters of the same type. Some elements are not strictly designed by loads, for example, by the use of minimum gage skins and landing gear mechanisms.

Because of the great uncertainty in generalizing the structural weight changes a simple but reasonable approximation is made in the model used in this study. It is assumed that the change in weight of the fuselage, tail, and landing gear of the second design point vehicle is directly proportional to the change in gross weight from the first point design. Typically, the structure involved here represents about 10 percent of the gross weight, and a 50-percent increase in gross weight will result in a structural weight increase of about 5 percent of the gross weight.

The computer program allows easy modification of this approximation when it is considered necessary and when specific aircraft are being studied.

Drive System

The methods developed in Reference 2 were used to determine the rate-of-weight change with torque for the main transmission. The analysis includes the optimized weights of shafts, bearings, case, and all gears. For this study, the following gross weight helicopters were studied: 3000, 15,000, 25,000, 100,000, 200,000. These are considered typical of observation, utility, gunship, cargo, and crane helicopters.

Figure 12 illustrates the results of the analysis. While the data has been specifically obtained for the main transmission, the percentage change is assumed to apply to the entire drive system.

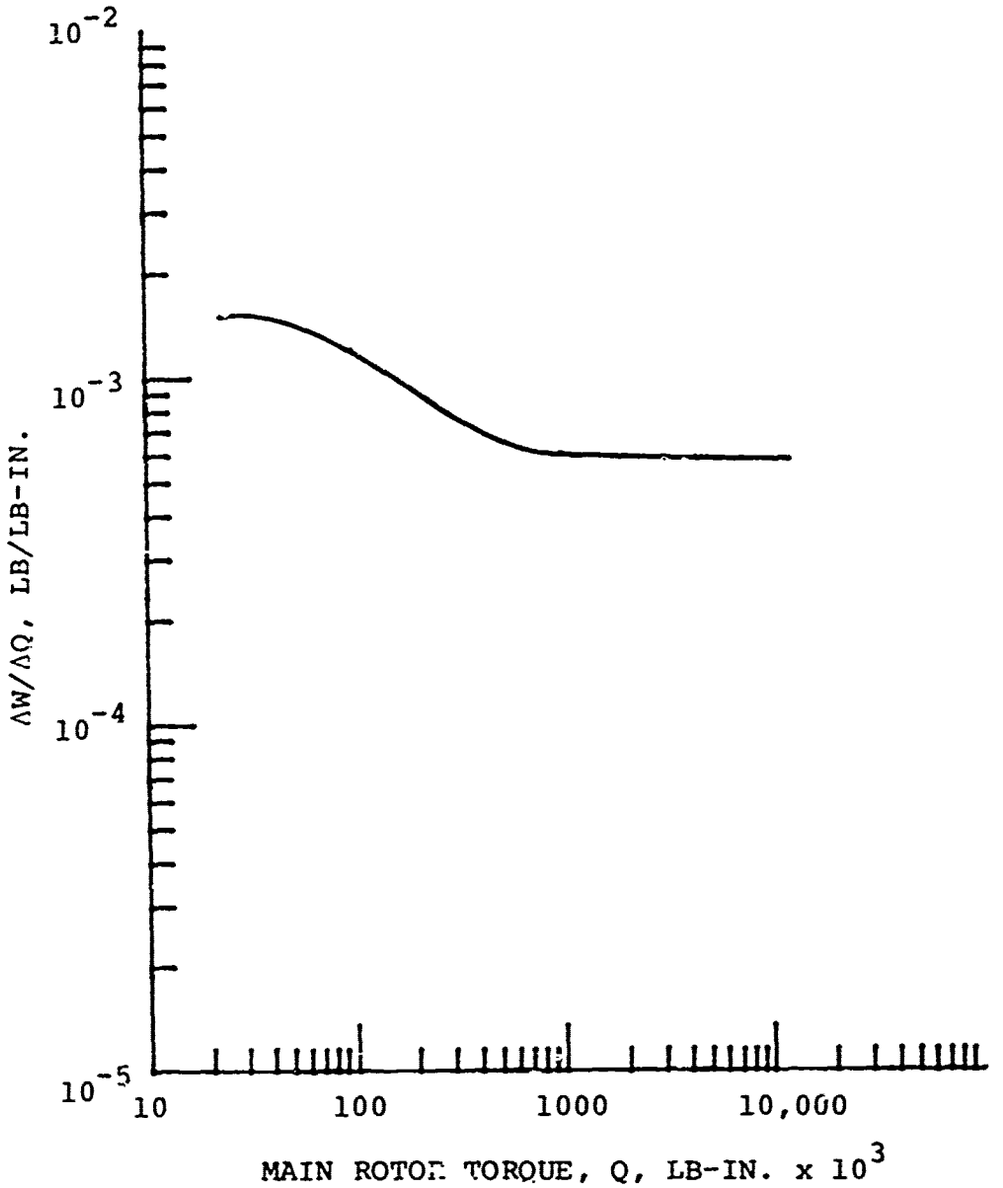


Figure 12. Main Transmission, Rate of Change of Weight Per Unit Change in Torque.

Rotor

The relationship between rotor torque and weight-change-with torque is determined. To facilitate this, it was decided to limit the study to the components which are more directly related to the imposed loads. These consist of the blades and the hub. The retentions, controls, bearings, etc., can be idealized structurally; however, because these items have so many mechanical configurations, the calculated component weights would vary directly from actual cases. It is assumed that the weights of these items vary identically as the analytically determined blades and hub.

The approach used here is as follows.

1. Assemble data providing design parameters (disc loading, tip speed, etc.) for various gross weight helicopters. Fit equations to these data.

2. Choose materials and appropriate working properties.

3. Define configuration to be modeled. Write equations defining structure.

4. Define load conditions.

5. Determine minimum weight configuration for each point chosen for analysis.

6. For each point, increase and decrease gross weight by five percent and determine optimum weights for each new point.

7. Analyze the hub in a similar manner.

Design Equations - The design equations were derived from the data of Table IV which is considered to be typical of the five types of helicopters.

TABLE IV. TYPICAL PARAMETERS					
Category	Observ	Utility	Gunship	Transport	Crane
Nominal G.W.	3000	15,000	15,000	70,000	150,000
Rotor					
Disc Loading	4	8	9	9	9
Tip Speed	650	700	700	700	750
Solidity	.05	.1	.1	.1125	.1
Blade Loading	80	80	90	80	90
Number of Blades	4	4	4	5	6
Aspect Ratio	25.4	12.7	12.7	14.14	19.1

From the foregoing data, the following equations are derived for rotor blades.

$$R = .564(GW/DL)^{.5}$$

$$CB = R/AR$$

$$DL = 9.27515 - 15680.1/GW$$

$$QMR = .58296(GW)^{3/2}/(DL)^{1/2}$$

$$\Omega = 11948(DL/GW)^{.5}$$

Materials - Blade section materials were chosen as given in Table V.

Configuration - A typical configuration was selected as follows:

(a) Symmetrical Airfoil 00 Series
(b) Mass Balanced Blade at 1/4 Chord
(c) Inboard Extension of Mass Balance to be a Variable
(d) Thickness Ratio = .12
(e) Trailing Edge Cap Length = .05 CB
(f) Spar Wall = .08 x .12 CB = .0096 CB
(g) Spar Width "x" is Variable

TABLE V. BLADE MATERIALS

Element	Material	Lb-In.^{3} Density	Design Static Strength	Design Fatigue Strength	Comments
Spar	2014 Al	.101	68,000	6000	Structural Member
Mass Balance	Lead	.4	-	-	-
Spline Cap	Al	.101	-	-	-
TE Cap	Al	.101	-	-	-
Spline	Honeycomb	.0018	-	-	-
Bond	Glue	.0745 (Lb-Ft^2)	-	-	-

The unit weights of the separate elements are found to be of the following form: (Weights in lb/in. of length, where CB is blade chord and X is spar width)

Spar: $W_1 = .00097\ CB\ (2X + .17\ CB)$

Core: $W_2 = .000108\ (CB(CB-X) - .0025(CB^3/(CB-X))$

Core Cap: $W_3 = .000024\ CB(.0036\ CB^2 + (CB-X)^2)$

Core Glue: $W_4 = .00103\ (.0036\ CB^2 + (CB-X)^2)$

Trailing Edge Cap: $W_5 = .0000152\ CB^3/(CB-X)$

Mass Balance: $W_6 = .0453(CB)\ x - .000435\ CB^2$

The area of the spar:

$$A = .0192\ CB(X + .085\ CB)$$

Load Conditions - The following load conditions were considered: Centrifugal loading, edgewise moment (starting torque with limit torque factor of 2), and droop bending moment (limit factor of 2.67). Because no general fatigue criteria due to flatwise bending could be developed, this condition was excluded.

Minimum Weight Configuration

The minimum weight blades meeting the strength requirements was obtained. This data is given in Table VI.

TABLE VI. WEIGHT ANALYSIS - BLADES AND HUB

G.W. (lb)	Total Blade Weight (lb)	Hub Weight (lb)	Hub & Blade Weight (lb)
3,150	147.56	15.00	162.56
3,000	130.33	14.31	144.64
2,850	120.67	13.73	134.40
15,750	870.96	70.16	941.12
15,000	796.51	65.16	861.67
14,250	724.16	61.24	785.40
26,250	1930.56	138.80	2069.36
25,000	1811.43	130.62	1942.05
23,750	1690.25	122.51	1812.76
73,500	8397.16	636.22	9033.38
70,000	7945.01	600.14	8545.15
66,500	7498.49	563.84	8062.33
157,500	22948.80	1970.38	24919.20
150,000	21967.50	1849.78	23817.30
142,500	21050.20	1739.41	22789.60

Vary Gross Weight - The gross weight was varied by ±5 percent, and the blade weights were obtained as above and are also given in Table VI.

Analyze Hub - A similar analysis was performed for the hub. The results are given in Table VI.

The resulting data is given on Figure 13.

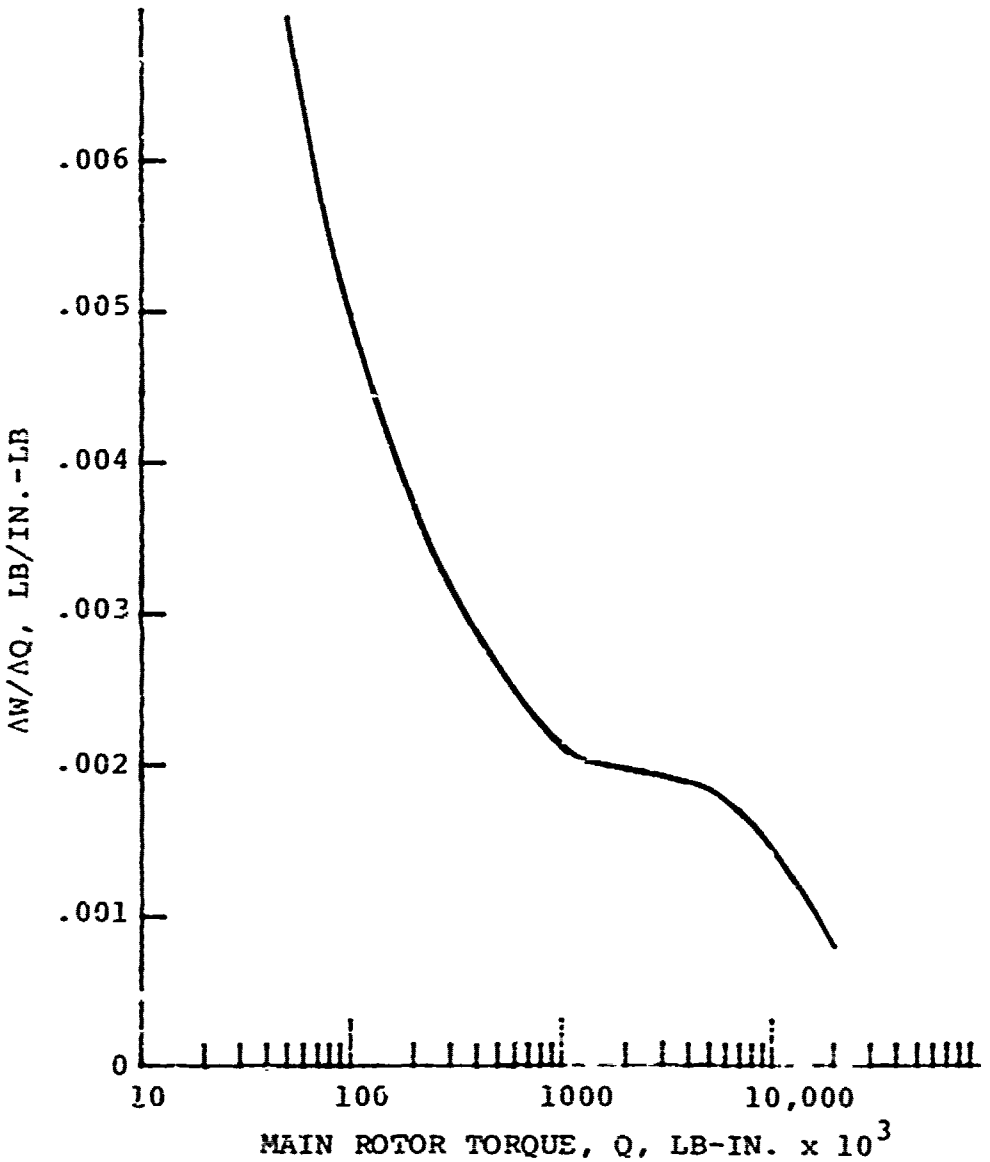


Figure 13. Main Rotor, Rate of Change of Weight Per Unit Change in Torque.

EFFECTIVENESS

It is the basic premise of the study that helicopters designed structurally for operation at off-design conditions will be heavier and costlier than those whose structure is designed at the same point at its powerplant. However, it is believed that when the entire operating spectrum of a helicopter is examined, it will be found that the extra capacity of the overdesigned helicopter will more than compensate for its additional size, weight, and cost. Thus, each of the helicopter types analyzed - utility, cargo, crane, observation, and gunship - is analyzed for its overall effectiveness in performing its intended functions at both design and off-design conditions.

Mission Parameters

As in other parts of the model, the primary emphasis of submodels used to calculate mission effectiveness is on determining the differences in effectiveness between a helicopter designed with a single, arbitrary design point and a helicopter designed with an additional, more structurally demanding design point. Hence, the mission effectiveness measures are those which concentrate on evaluating these differences, once a suitable base is established. The base or reference level mission effectiveness for each type of helicopter is established by its design point requirements. These requirements are the performance parameters and associated data derived from mission analyses. The mission analyses translate operational requirements (numbers of troops to be airlifted, distance from troop bases to the forward edge of the battle area (FEBA), weapons compliments, supply levels and resupply rates, etc.) into design requirements for the helicopters (payload, range, speed, OGE hover requirements, etc.). And these design requirements are specified such that under the most severe, or nearly most severe, conditions, the helicopter will be able to perform its required mission.

In the case of a transport helicopter for example, the payload may be specified as so many troops and/or equipment and cargo, the range as that necessary to move the payload from staging areas to the FEBA, the speed as that required to achieve the required reaction time or resupply rate, and so forth. To permit sizing of the powerplant, it is necessary to specify a stringent combination of hover altitude, temperature, climb rate, and power margins that the aircraft must be able to perform with its design payload. For either a single-point design helicopter, or a helicopter with two design points, this requirement would be the same.

Similarly, for either a single- or two-point design helicopter, the required cruise speed would be the same. When the transport helicopter is operating at off-design conditions, its cruise speed could be modified, or its range, or any other pertinent parameter within its capabilities. In general, however, it is more appropriate to fix these values at their required levels and to examine the change in the helicopter's "essential" operational parameters. In the case of a transport helicopter this would be payload. Thus, the appropriate measure of mission effectiveness to be analyzed in this study for the transport helicopter would be its payload.

For the purposes of comparing helicopters with two design points to those with a single-point design, payload is an excellent mission measure of effectiveness (MOE) for all of the helicopter types to be analyzed by the model except for the observation helicopter. Both utility and crane helicopters have payload as their primary variable requirement, with range, speed, etc., set by operational requirements that must be met (but not necessarily exceeded) under any payload conditions.

For the armed helicopter, performance at high speeds is a primary design requirement, along with weapons load. But again, payload has been selected as the variable design parameter for the mission MOE; the assumption has been made that the design value of speed (and the associated maneuver load criteria) was selected after trading off payload versus rapidly increasing component weights, particularly the rotor, at higher speeds. At these conditions, there is much more leverage in increasing payload than there is in increasing speed, because of the design limitations of the helicopter. In addition, only a small portion of the helicopters' flight time will be spent at the maximum conditions, thus arguing that increasing capability (payload) at lower speeds is the preferred way to increase overall mission effectiveness.

For the observation helicopter, the primary performance parameter is time-on-station or time on patrol which can be expressed simply as endurance at stated conditions (best speed for range, best speed for endurance, etc.). Since payload is relatively fixed (observer, tracking equipment, etc.), and other performance parameters like speed are not dominating requirements, endurance at a nominal patrol speed has been selected as the appropriate mission MOE for the observation helicopter.

Operational Parameters

Operational parameters are those parameters which define the total environment, both natural and man-made, within which the helicopter will operate. These parameters are included in the analyses defining the helicopter design requirements and can significantly influence the intended operation of the helicopter if they are different from those designed for.

In this study, the only environmental parameter which must be included in the two-point design analysis is the altitude-temperature profile that the helicopter must operate in. However, this combined parameter is extremely important since it determines the off-design capability of the power system of a helicopter, and hence, its overall mission effectiveness parameter (payload or endurance).

A helicopter designed to hover with a specific payload at a specific altitude and temperature has a specific design horsepower powerplant requirement. At different altitudes and temperatures, its powerplant will deliver different power, and hence, its payload (or fuel load) capacity will also vary. Thus it will achieve different levels of mission effectiveness (payload or endurance) depending on the conditions under which it must hover during the course of a mission. The most concise and unambiguous method to analyze this problem is presented in Reference 3, and is the method used in this study. In this method, a joint temperature - altitude probability distribution for a given geographic area is prepared first. An example of such a distribution is shown in Figure 14. The data for both altitude and temperature for this study were taken from the above reference, and an explanation of how the data are obtained and processed is given there.

The data shown in Figure 14 are used to calculate the overall probability that a given helicopter carrying a given payload and fuel load can hover in the specified environment. This is done by determining the maximum temperature that the aircraft can hover at for a given gross weight at a given altitude. This point is plotted on the graph, and after several more points are calculated, a line, such as that labeled "GW", can be drawn. The process can be repeated for any other gross weight as well (GW_2, GW_3). This line represents the combination of altitudes and temperatures above which (higher altitudes and/or higher temperatures) the helicopter cannot hover, and hence, cannot perform its mission. The proportion of area of the chart below this line, compared to the total area of the chart, is then numerically equal to the probability that the helicopter can

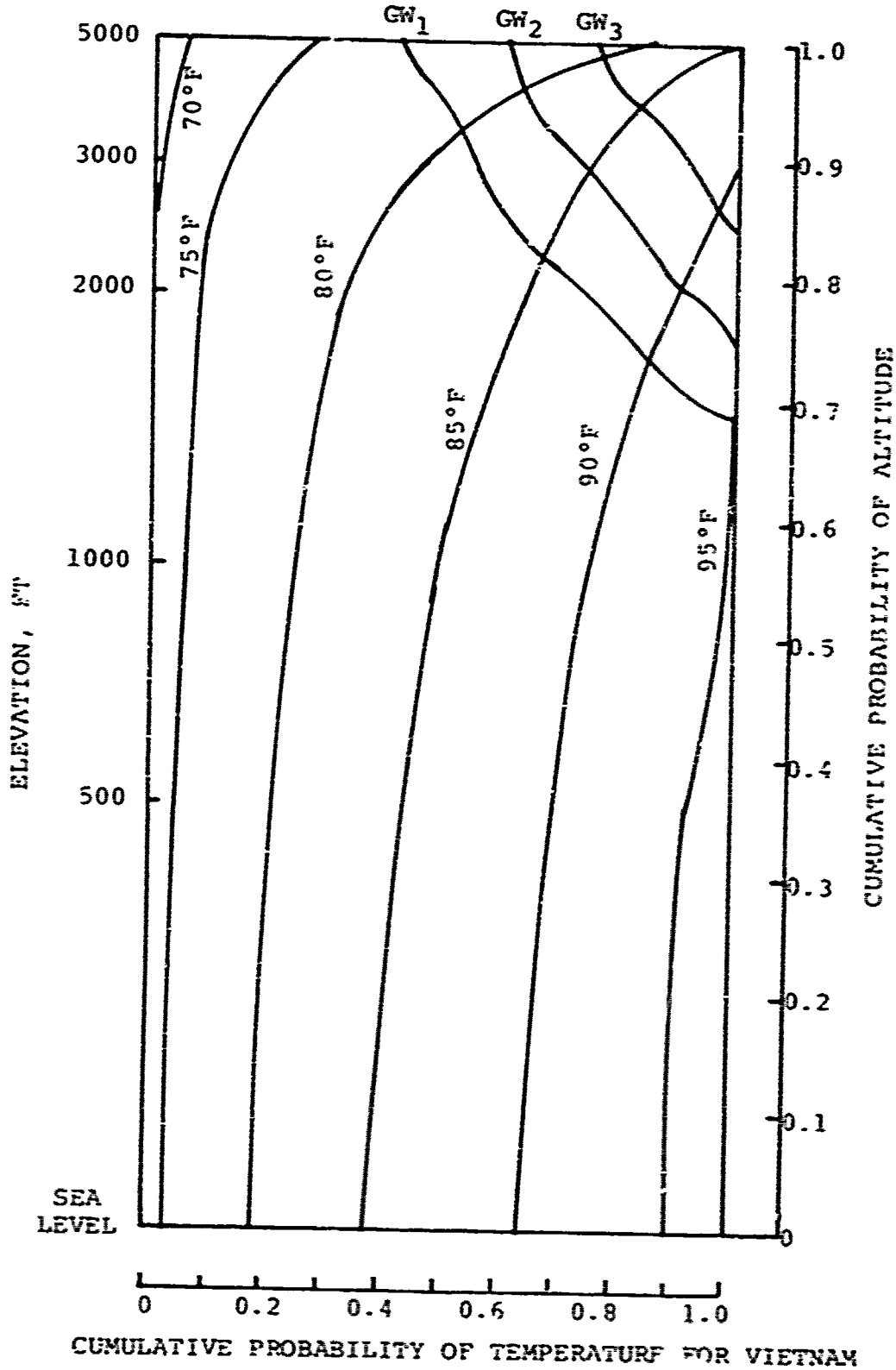


Figure 14. Typical Altitude - Temperature Data.

perform its mission in the given environment at the specified gross weight.

The process described above derives the probability of hover, or the probability that the helicopter can perform its mission under given conditions. To measure overall effectiveness, several different combinations of payload and environment must be considered. The method of handling several different payload levels is discussed in the next section. The method used in the model to account for operations in several different environments is discussed below.

To simplify the computation of overall hover probability in the areas of the world of prime interest to the Army, the characteristics of 19 countries contiguous to the Sino-Soviet block were combined. These are the same areas used in the analysis referenced above, and are listed in Table VII. The altitude and temperature profiles of the countries were combined to form an overall "world" altitude-temperature profile.

For combining altitude data, the probability of being at or below a given altitude is equivalent to the percentage of land at or below that given altitude if it is equally likely that one could or would be placed anywhere in the total land area. Thus,

$$f_{j\ cum} = P_{j\ cum} = \frac{\sum_k P_{jk} A_k}{\sum_k A_k}$$

where:

f_j is the frequency of occurrence or cumulative probability of being placed at the j-th altitude within the combined area of the k countries.

P_{jk} is the frequency of occurrence or cumulative probability of being placed at the j-th altitude within the k-th country.

A_k is the area of the k-th country.

TABLE VII. AREAS USED IN HOVER PROBABILITY CALCULATIONS
1. West Germany
2. France
3. Italy
4. Spain
5. Turkey
6. Syria - Lebanon
7. Saudi-Arabia
8. Iraq
9. Iran
10. Afghanistan
11. West Pakistan (Pakistan)
12. India
13. East Pakistan (Bangledesh)
14. Burma
15. Thailand
16. Laos
17. Cambodia
18. Viet Nam (North and South)
19. South Korea

For temperature, it is necessary to combine the temperature/probability data in such a way as to preserve the relationship between temperature and altitude. Hence, temperature should be derived on a weighted basis for each altitude investigated. Treating the probability of a temperature occurring as its frequency of occurrence:

$$f_{ij\ cum} = P_{ij\ cum} = \frac{\sum_k P_{ijk} A_{jk}}{\sum_k A_{jk}}$$

where:

f_{ij} is the cumulative frequency of occurrence of temperature T_i at altitude h_j within the combined areas of the k countries.

P_{ijk} is the probability of T_i occurring at h_j in each of the k countries.

A_{jk} is the area of each country at the altitude h_j.

For altitude, the simple calculation can be performed by multiplying the percentage of total area represented by a country and the cumulative percentage of area in that country at or below a given altitude to obtain the fraction of the total area of all of the countries at or below a given altitude represented by that country. For a given altitude, when this is done for each country and the results added, the total represents the cumulative percentage of the total area at or below the given altitude. Thus:

$$P_{cum\ j} = \sum_k P_{cum\ jk} \times (A_k/A)$$

For combining temperature data, the process is more complicated. Here, it is necessary to weight a given temperature value by both its frequency of occurrence in a given country at a given altitude and by the percentage of area of a given country at that altitude. Cumulative frequencies of temperature for a given temperature level, multiplied by the fraction (not cumulative) of area represented at a given altitude yields the cumulative frequency of occurrence of the temperature at the altitude of interest. Thus:

$$P_{i(cum)j} = \sum_k P_{i(cum)jk} \times (A_{jk}/A_j)$$

Since the altitude/fraction data is given on a cumulative basis for each country, a method to derive a frequency distribution histogram must be devised so that area fractions can be assigned to each altitude for each country. If a very large number of altitudes were being investigated, the histogram would closely approach the actual frequency distribution. However, only a limited number of altitudes were calculated in the current calculation. Applying the histogram approach to the entire altitude distribution curve might result in a rather gross histogram where much area would be represented by a single altitude. However, since the accuracy of the source cumulative frequency distributions for temperature (and altitude to some extent) is somewhat questionable (as indicated by the methods used to derive these data in Reference 4), extreme care with development of a histogram is probably not warranted considering the overall temperature accuracy.

To derive a consistent weighting function for the temperature data at each altitude, a sampling approach was used. Here, a narrow band of altitudes was selected around each altitude for which weighting data were required. This number was used as the weighting factor. The size of this number depends on

two things: (a) the size of the country; and (b) the slope of the altitude distribution curve in the band selected. Obviously, the size of the country should have an effect on the weighting of the temperature. The local slope indicates how rapidly area is being added to the cumulative distribution and thus reflects the percentage of area considered as representative at a given altitude.

After performing the necessary computations, the cumulative joint probability distribution for altitude and temperature was obtained. These data are shown in Figure 15 and were prepared in tabular form for use in the computer model.

Utilization Parameters

The previous sections discussed the choice of payload or endurance as mission effectiveness parameters, and how environmental conditions of hover altitude and temperature affect the helicopter's ability to perform its mission. The latter factor is dependent upon the amount of payload or endurance required, which in turn, depends on how the helicopter is utilized. Thus, a third factor to be considered in determining the effectiveness of a helicopter is the relative frequency with which it will operate with a given payload or endurance (fuel load) over its operational lifetime.

An example of utilization data is shown in Figure 16, which was prepared using data from Reference 5. These data illustrate typical utilization patterns for utility transport observation, crane, and gunship helicopters. Unfortunately, many assumptions would have to be made to derive a relationship between the gross weight distributions shown in Figure 16 and the corresponding payload/fuel load distribution. Since this analysis was beyond the scope of the current study, this effort was not conducted. However, it was recognized that this could be an important factor in determining overall effectiveness and in optimizing selection of a second design point. Hence, provision was made in the effectiveness model, and the computer program, for including this factor in the effectiveness calculation.

Effectiveness Equations

The basic equation used to calculate the effectiveness of a helicopter is:

$$MEI = (PL)(PHOV)(PUF)$$

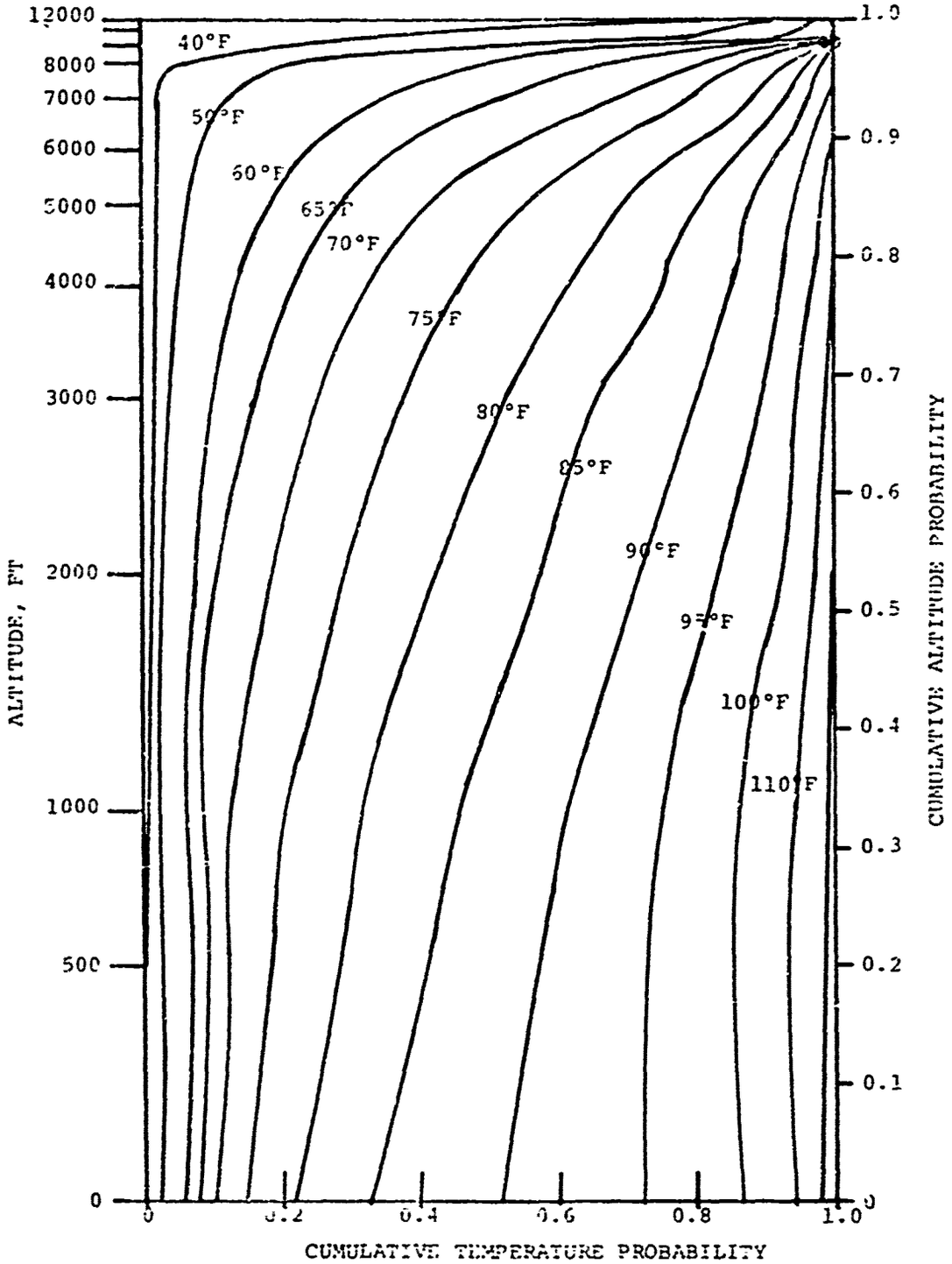


Figure 15. Combined Altitude - Temperature Data.

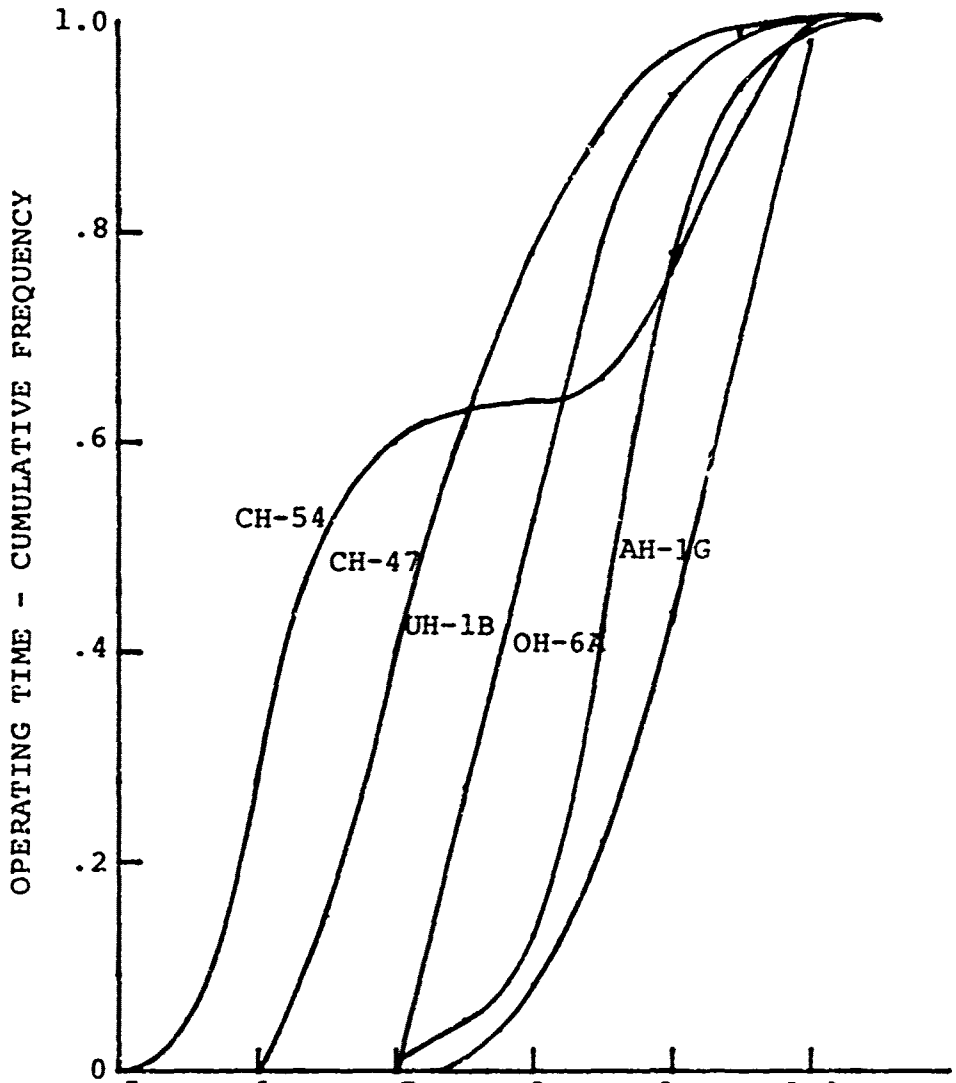


Figure 16. Sample Utilization Data.

The use of this effectiveness index is illustrated below.

It is desired to determine the optimum two-point design of a transport helicopter with a given design payload requirement at a hover condition of 4000 ft, 95°F, 500 ft/min rate of climb. The ship must have a range of 300 NM, a cruise speed of 160 knots, etc. Using the techniques described in previous sections, two helicopters are "designed": a helicopter with a single-point design (4000 ft/95°F) and a helicopter with two design points (4000 ft/95°F, 2000 ft/75°F). These helicopters have different gross weights and empty weights, but a common design payload.

To compare their respective effectiveness levels, representative payloads spanning their range of capabilities will be used, from 0 payload up to a payload that overloads the helicopter, say 10 percent over design payload. Using these payloads, a utilization frequency is obtained for each level. These are the same for either helicopter, since the payload increments match. However, the gross weight level associated with each payload level is different, depending on which helicopter is examined. This results in a different probability of hover, which in turn, affects the effectiveness index. After the effectiveness index is calculated for each payload increment, the sum yields the overall effectiveness index for each helicopter.

A sample calculation illustrating this technique is shown in Table VIII. Included in the table is a cost factor covering life cycle costs for each helicopter. A detailed explanation of how these costs are calculated is given in the next section. If costs are included, a cost effectiveness index can be calculated rather than just a simple effectiveness index. Thus

$$OCE = \sum_{j=0}^{k} \frac{(PL)(PHOV)(PUF)}{CPFH}$$

where:

CPFH is the total cost per flight hour for the helicopter.

After examining the case for the two-point design helicopter described above, another calculation can be performed for another one (say 4000 ft/95°F, 3000 ft/75°F). Several of these calculations then lead to a series of optimization curves, from which the optimum second design point for the helicopter with the given mission requirements can be determined.

TABLE VIII. SAMPLE COST EFFECTIVENESS CALCULATION

		Single Design Point Helicopter				Two Design Point Helicopter			
Payload (lb)	Utilization Frequency	Weight (lb)	Hover Prob	Index of Eff	Cost ($/hr)	Weight (lb)	Hover Prob	Index of Eff	Cost ($/hr)
0	.05	14,840	-1	0.0	285	17,090	-1	0.0	310
500	.05	15,340	-1	25.0	286	17,580	-1	25.0	311
1000	.05	15,840	-1	50.0	288	18,070	-1	50.0	312
1500	.05	16,300	.995	74.6	291	18,600	-1	75.0	314
2000	.05	16,760	.990	99.0	294	19,070	-1	100.0	316
2500	.05	17,220	.985	123.1	298	19,560	.990	123.8	319
3000	.10	17,700	.975	292.5	302	20,050	.989	296.7	322
3500	.05	18,220	.973	170.3	306	20,560	.983	172.0	326
4000	.20	18,780	.968	774.4	310	21,070	.976	780.8	330
4500	.10	19,350	.962	432.9	315	21,440	.970	436.5	335
5000 (Design)	.15	20,000	.950	712.5	320	22,200	.950	712.5	340
5500	.10	20,600	.891	490.0	340	22,755	.920	506.0	345
			MEI = 3244				MEI = 3278		
			OCE = 10.33				OCE = 9.86		

COST MODEL

The cost model is a statistical approach to aircraft cost estimating. The cost estimating relationships utilize empirically derived cost functions. The costs have been functionally related to basic aircraft parameters such as empty weight of airframe and installed power of engine. The model is sufficiently detailed to distinguish cost differentials between two-point design and single-point design helicopters. This approach allows cost estimates to be made based on primary aircraft parameters before a detailed design is actually completed. The cost model is broadly divided into initial and operating costs as illustrated in Figure 17.

Initial Costs

Initial Production costs are based on a model evolved by E. H. Yates (References 6 and 7). The production costs are subdivided into airframe, engines, and government-furnished aircraft equipment (GFAE).

Airframe direct labor costs were derived from direct labor man-hour data contained in Aeronautical Manufacturers Planning Reports (AMPR). The costs were estimated on a per-pound basis at production number 1000. The average labor learning curve as a function of production number, NP, is:

$$C = 43.12(NP)^{-.39}$$

The airframe labor cost, in dollars, is given in terms of AMPR weight*, WA, in pounds:

$$\text{Labor Cost} = 17.63\ WA^{.85}$$

Thus, when combined with the learning curve, the airframe labor cost is

$$CL = 760.2\ WA^{.85}(NP)^{-.39}$$

where CL is in dollars and WA is expressed in pounds.

---

* AMPR weight is Aeronautical Manufacturers' Planning Report weight which is empty weight of aircraft less (1) wheels and brakes, (2) engines, (3) starter, (4) cooling fluids, (5) instruments, etc.

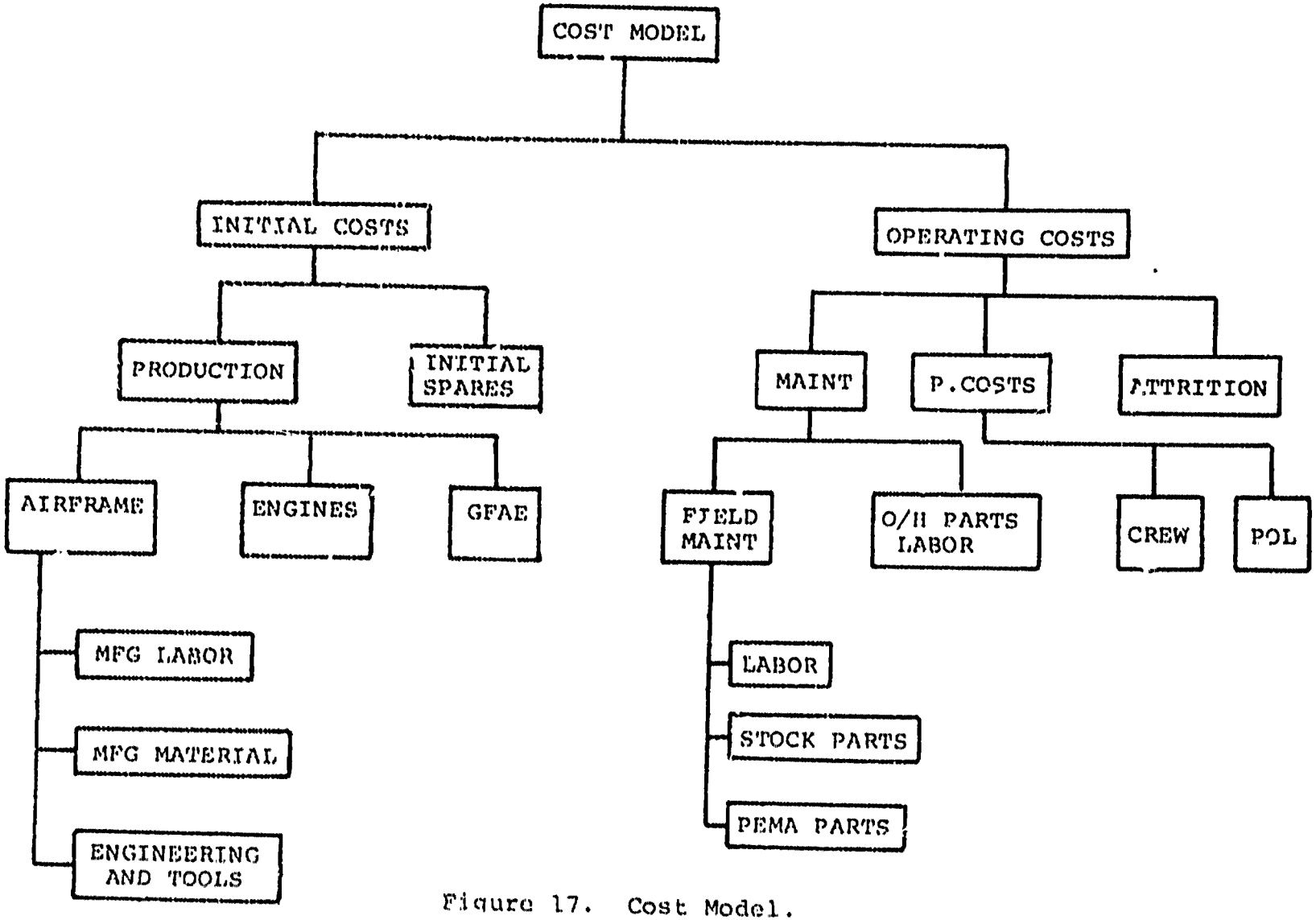


Figure 17. Cost Model.

The cost for airframe materials for subsonic aircraft was found to be a function of airspeed as well as weight and number produced. The following relationship is for fixed-wing aircraft, but since its effect is small and a better relationship was not found, airframe material cost is defined as:

$$CM = .007(WA)/(NP)^{.12}(TAS)^{1.24}$$

where TAS is the aircraft speed in knots.

Cumulative average engineering and tooling costs per airframe are related to the total number of units planned for production. Based on post World War II aircraft the relationship for engineering and tooling costs is:

$$CET = WA(220/NP + 7.5/NP^{.15})$$

where NP is the number of units planned for production.

There are additional costs which must be added to the above costs. These are general and administrative expense (GA), engineering change proposals (ECP), and profit (P). The following relations apply:

$$GA = 0.1(CL + CM)$$

$$ECP = 0.1(CL + CM + CET + GA)$$

$$P = 0.1(CL + CM + CET + GA + ECP)$$

In the equations above, the AMPR weight, WA, is related to the empty weight, WEM, by

$$WA = .749\ WEM + .126$$

The correlation coefficient of this equation is .997 with a standard error of .356 kilo-pounds (Reference 7).

Data for 17 turboshaft engines were analyzed. The best least-squares fit was made to obtain the following relationships:

$$CE = PRA(58 - .006\ PRA/NEN) \qquad PRA \leq 3500\ HP$$

$$CE = NEN(129{,}500 + 37(PRA/NEN - 3500) \qquad PRA > 3500\ HP$$

where PRA is in horsepower per engine and CE is in dollars.

The GFAE category usually includes the cost of ordnance and armament and electronics that are not an integral part of the airframe and engines. This cost item is highly dependent on mission objectives and type of helicopter, and therefore, is quite sensitive to the function of the aircraft. For this cost model, insufficient data exists to derive a cost estimating relationship for the various types of helicopters, i.e., utility, observation, gunship, etc. An estimation equation with limited data was found to be:

$$CG = -3760 + 4.79\ WA$$

The initial costs are plotted in Figure 18 for production number of 100 of a 120-knot helicopter.

The cost of production (CP) is then defined by:

$$CP = 1.331\ CL + 1.331\ CM + 1.21\ CET + CE + CG$$

To simplify the model, the constants are included in the individual terms to yield:

$$CL = 1011(WA)^{.85}/(NP)^{.39}$$

$$CM = .00931(WA)(TAS)^{1.24}/(NP)^{.12}$$

$$CET = 1.21(WA)(220/(NP) + .75/(NP)^{.15})$$

and

$$CP = CL + CM + CET + CE + CG$$

The costs of attrition (CA) and initial spares (CI) are defined as:

$$CA = (YAR)(CP)(SL)/(NP)$$

and

$$CI = .1(CP)$$

The production, spares, and attrition costs are summed to form:

$$CPIA = CP + CI + CA$$

Operating Costs

The operating cost portion of the model is broken down into three main categories, as is shown in Figure 17: maintenance, direct operating costs, and attrition.

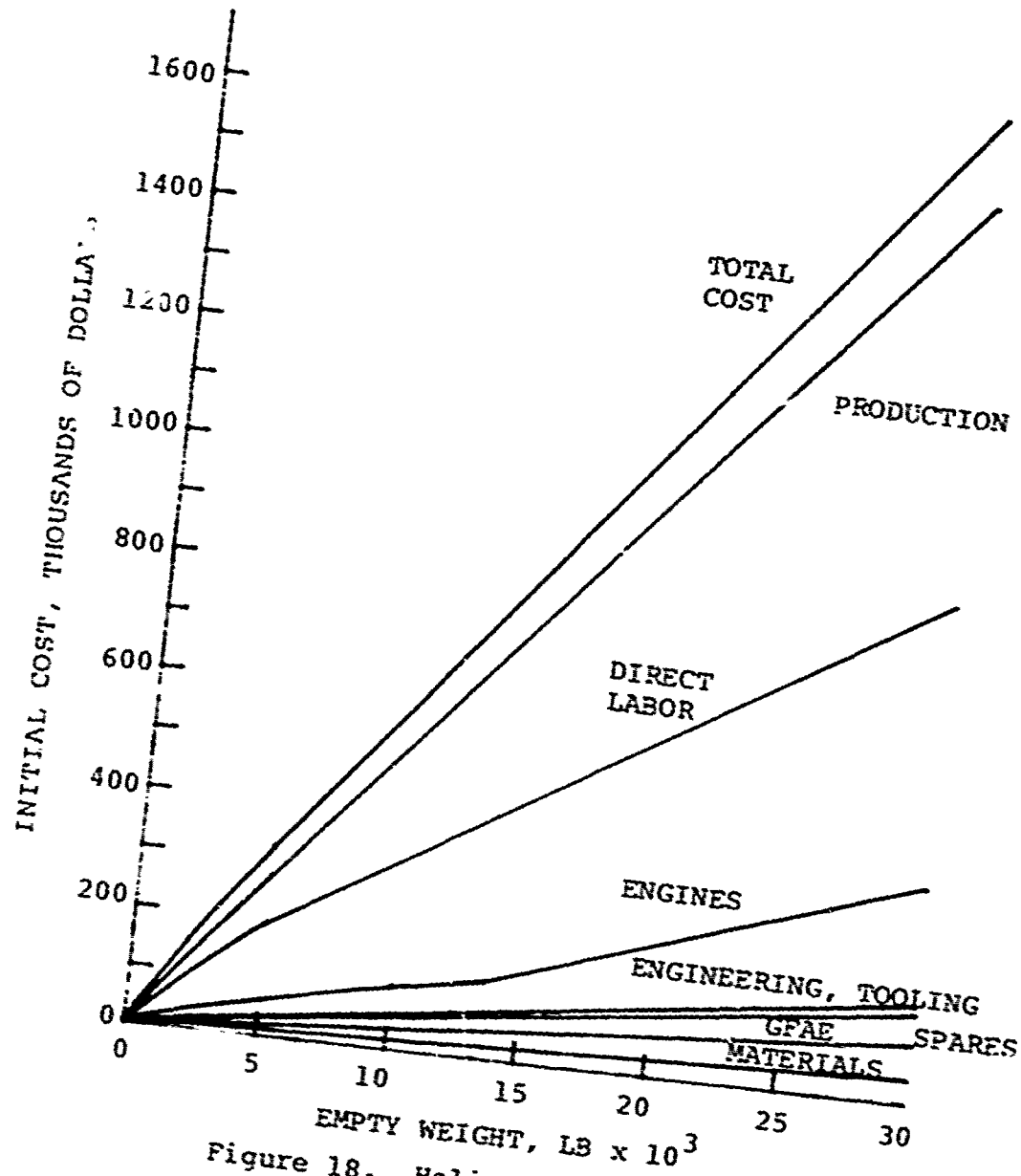


Figure 18. Helicopter Initial Costs.

Maintenance costs consist of overhaul parts and labor and field maintenance. The field maintenance category includes labor, stock fund parts, and PEMA (Procurement of Equipment and Missiles, Army) parts. The direct operating costs are composed of crew costs and petroleum, oil and lubrication (POL).

A regression analysis was performed on the operating costs (Reference 9) of 17 Army helicopters to determine cost estimating relationships. The correlations are based on the empty weight, WE, of the helicopter. These operating cost estimating relationships are given below:

$$\left.\begin{aligned} CFM &= .00191\ WEM^{1.323}\ TN/T \\ COPL &= 1.016\ WEM^{.4}\ TN/T \\ CPOL &= .0038\ WEM^{.89} \end{aligned}\right\}\ WE \leq 3200\ lb$$

$$\left.\begin{aligned} CFM &= -87.6 + .05362\ WEM\ (TN/T) \\ COPL &= 18.16 + .00233\ WEM\ (TN/T) \\ CPOL &= -2.5 + .00235\ WEM \end{aligned}\right\}\ WE > 3200\ lb$$

$$CC = 101 + .0033\ WEM$$

The fuel costs can be described by the above statistical relationship but since the actual fuel load is calculated, the fuel cost can be calculated using:

$$CFUL = .25\ WFL/6.5$$

The expression for the maintenance operating (CFM) costs is for average operating conditions. If the helicopter is operated at an overloaded or underloaded condition, the maintenance costs will differ from these. This comes about due to a change in the average failure rate of components of the helicopter system. Maintenance costs are inversely proportional to the mean time between failures. To determine the mean time between failure, the probability of failure as a function of loading was related to the probability of failure as a function of time.

The probability of failure, $P_F(L)$, as a function of load, L, is typically of the form (References 10 and 11):

$$P_F(L) = 1 - \exp\left\{-\left(\frac{L + a}{\beta}\right)^{\gamma}\right\}.$$

The probability of failure as a function of time shown is of the form:

$$P_F(t,T) = 1 - \exp\left(-.693 \frac{t}{T}\right)$$

where T is the mean time between failure.

When the elapsed time t = T, the probability of failure is .5; i.e., at the mean time between failure the probability of failure is 50%. This corresponds to an average loading of the helicopter (L = 1). If the loading is increased to L = 1.4 for example, the probability of failure increases to .9. A failure probability of .9 at time T1 determines a new failure curve and thus defines a new MTBF which is T2. In other words

$$P_F(L) = P_F(T1, T2)$$

resulting in

$$\frac{T2}{T1} = \frac{.693}{\left(\frac{L + \alpha}{\beta}\right)\gamma}$$

Thus, the new MTBF, T2, is determined from the load factor L and T1.

The mean-time-between-maintenance action, T1, was determined from data in Reference 12. Failure rates per flight hour were determined for 35 systems included in helicopters. From this data, the average flight hours per failure was determined and was correlated to the empty weight, WE, of the helicopter. The relationship is

$$T = .6778 + \frac{5544}{WE}$$

where T is the average mean time between failures in hours and WE is the empty weight in pounds.

The loaded maintenance costs are then obtained by multiplying the above equation for CFM by the inverse ratio of the mean time between failures.

The above relationships result in the relationship

$$T/TN = .693/((OLF + .01/7)/1.15)^3$$

where T, TN are the actual and the normal mean time between failures. OLF is the overload factor and is the ratio of the gross weight to the design gross weight. In practice, it is found that in effect that OLF is never less than .5, corresponding to T/TN of 7.6. This implies that maintenance is still performed even if the helicopter is not flown. When OLF $\geq$ 1.5, catastrophic failure occurs.

The costs are all reduced to cost per flight hour using the following equation:

$$CPFH = CPIA/(12(MFH)SL) + CD + CMT + 60\ CFUL/TTIM$$

where TTIM is mission time in minutes and CPFH is in $/hr.

COMPUTATIONAL METHOD - USER'S GUIDE TO ZODIAC II

The elements of the analytical model described above must, of necessity, originate in several different sections of an Engineering Department. Each of these model elements is subject to minor or major changes due to variations in mission and helicopter type under study, technology level, design philosophy, data available, or budgetary considerations. The overall logic is similarly subject to changes for these reasons and, in addition, because of the particular parameters to be varied and because of variations in the purpose of the study.

The conventional approach to this modeling study would range from a simple computer program which would have to be reprogrammed for each of the many modifications in the model to a complex program which includes prior provisions for all the possible modes of operation. The objections to these methods include the high cost of many changes (for the simple program) or high initial costs (for the complex program), the difficulty of varying modes of operation; changing criteria; and expanding capabilities. The most serious disadvantage, however, is the fact that the engineer is left out of the decision loop. Typically, the engineer would describe his requirements to a programmer (or make major program changes himself), or he would select options from a coded table. In this environment of continual change, the risk of getting meaningless data is significant.

The approach selected here for implementation is an outgrowth of a computer program originally devised at Kaman primarily for weights analyses, where typically the analytical models are continuously changing. This program, called ZODIAC, is described in Reference 13.

The primary motivation in the development of this program was that it should be a tool which is completely meaningful to, usable, and changeable by an engineer with no or little programming experience. It is believed that this objective has been achieved in ZODIAC II.

PROGRAM FEATURES

Prior to describing the program usage in detail, some of the features and the organization will be discussed. Examples of all these features can be found in the listings of the model formulations in Appendix II.

Modular Organization

The analytical model is divided into separate computational units called "modules". Each module generally is used to carry out one logically self-contained computation. Examples were discussed in the previous section on the Analytical Model. Typically, individual modules will represent computations like statistical weight, mission fuel requirements, initial costs, mission effectiveness, etc. Many of the quantities involved in the module will not be used any place else in the model. Only these variables which are shared in common with other modules must be specified. The module will consist almost entirely of the equations which define the computation. Each module must have a name so that it may be referred to by the "control module". Modules may be easily changed by adding, removing, or changing equations or the entire module.

Control Module

The overall logic of the analytical model is carried out by the "control module". This portion of the program consists mainly of instructions specifying which module to run next. The same module may be run more than one time by the control module.

Logical Operations

The number of types of logical operations has been limited to true operational decision making functions. It is preferred that the engineer change his model to reflect changed ground rules rather than include several possibilities and choose between them with pseudo-logical input codes. In this manner the engineer is always fully cognizant of his model and it is always under his full control. This approach is possible because of the ease and safety with which such changes can be made. The two major logical operations included are: (1) automatic iteration within a module or around several modules (as is commonly used in weight estimation or in determining the power for maximum range); and (2) a conditional evaluation of a variable (as might be used to prevent a power required computation from exceeding some torque limitation).

Equation Form

The actual equations make up almost all of the model. The equations are written in algebraic form (consistent with FORTRAN) and allow all arithmetic operations including exponentiation. See listings in Appendix II for many examples.

Table Look Up

Since tabular data will normally make up a substantial portion of the input data to any modeling program, ZODIAC II has an automatic table look up feature. Linear interpolation of tables with up to three independent variables is automatically performed.

Input Data

Input of data is extremely simple. The program does not presuppose any point of input or any particular data. Data is always self-identified and may be freely input with the modules. Additional input may be called for during the running by the simple statement, READ. At this point, any desired data may be input or changed.

Output of Data

Output is also extremely simple. All that is required is the simple statement PRINT, followed by a list of the variables. On output each quantity is automatically identified.

The input and output procedures are especially convenient when compared to more formal languages, like FORTRAN, where typically many programming hours are spent deciding on all the input options and output formats.

Checks

Because of the free form of computation allowed, it is necessary for the program to include a number of built-in automatic checks. If a variable used in an equation has not been previously input or calculated, a warning message is printed. If it is necessary to extrapolate during a table look up, a warning message is also printed. If the equations in a module cannot be evaluated in the order presented, they are automatically rearranged and an error is indicated if appropriate.

USERS RULES

The rules for using ZODIAC II are quite simple, and all one needs to know is specified in detail in the following pages. In some cases comments meant for FORTRAN programmers are included in parenthesis. These comments should be ignored by engineers not familiar with FORTRAN. There are a few definitions required prior to discussion of the allowable statements:

Statement - meaningful (to the computer) contents of a punched card

Variable - an algebraic quantity whose name has from 1-4 letters or numbers, the first of which must be a letter. Examples: A, QMR, B123, A7L6. If the variable is common (see below) its name may be preceded by a $.

Constant - a number. It may be positive or negative, it may or may not have a decimal, and it may contain an exponential. Examples: 12, 15.7, 1.73E-5(= 1.73×10^{-5}), 6.54E16, 2E10.

The following codes are used in the following sections.

v ≡ variable

c ≡ constant

vc ≡ variable or constant

n ≡ name

Capital letters indicate precise words required as part of the statement. Blanks are ignored and may be used anyplace on cards for clarity (with the single exception of input table data).

All statements may be followed by comments on the same card provided a semicolon (11-8-6 punch) is used to separate the two.

There are three groups of statements: Control Module Statements, Module Statements, and Data Statements. These will be discussed separately.

Control Module Statements

The control module is used to specify the order of computation and includes the major logic of the simulation. Unless otherwise instructed the statements are carried out in the order shown. The allowable statements are listed and then described immediately following.

```
RUN MOD n

v = simple expression

POINT n

ITERATE ON v, ATOL = vc, PTOL = vc, TIMES = c, FROM n

COMMON v, v, v...

        (LT)
IF vc IS EQ  vc, GO TO n
        (GT)

GO TO n

READ

PRINT v, v, v...
```

RUN MOD n (1-20 characters) - This statement causes the named module to be run. The name specified must also appear on a Module Name statement in the module which is to be run. (This is similar to a call statement in FORTRAN, but no argument list is used.)

v = simple expression - This statement is used to perform simple arithmetic in the control module. A simple expression is one which involves addition or subtraction of two terms, or a single term. The two forms which this type statement can have are shown below.

$$v = vc \pm vc$$

$$v = vc$$

Point = n - This statement specifies a point in the control module which may be referenced by an ITERATE, IF or GO TO statement. The point name specified must follow the same rules as a variable name and it may not appear in the control module as a variable name. Its use will be clear when the mentioned statements are discussed.

ITERATE ON v, ATOL = vc, PTOL = vc, TIMES = c, FROM n - This causes the program to iterate on the statements which are located between the POINT specified by FROM n and the iterate statement. Iteration will continue until two successive values of the variable specified by ON v are within the limits set by ATOL or PTOL or until the number of iterations exceeds the amount specified by TIMES. The details for each specification are as follows.

ON v - The variable which is specified is the variable on which iteration will occur. This specification is optional, if it is omitted the computations will be repeated the number of times specified by TIMES.

ATOL = vc, PTOL = vc - Both ATOL and PTOL are used as a test for convergence of the variable specified in the ON specification. ATOL is an absolute tolerance and PTOL is a percent tolerance. Both ATOL and PTOL are optional, but both are omitted, PTOL is set equal to 5 percent. For convergence, one or both of the following must be true:

$$|v_i - v_{i-1}| < \text{ATOL}$$

and/or

$$100 \frac{|v_i - v_{i-1}|}{|v_i|} < \text{PTOL}$$

Note that PTOL is put in in percent not decimal.

TIMES = c - This specification is used to specify the maximum number of iterations to be allowed. If convergence is not obtained before the number specified by TIMES is exceeded, iteration stops and an error message is printed out.

FROM = n - This specifies from which POINT in the program iteration is to occur. The point name specified must be on a POINT statement which must precede the ITERATE statement.

COMMON v, v, v - A COMMON statement is used to specify variables which are used in other modules. The order in which variables appear on the card is not important, but the variable name must be the same in both modules. A $ preceding a variable name has the same effect as placing the variable name in a COMMON statement. A common statement must appear in a module before any of the variables on it are used. This statement is normally placed at the beginning of the module. Each card may contain up to 40 variables. As many COMMON cards as necessary may be used. (As distinguished from FORTRAN the order of the names is immaterial, only the names themselves are important.)

IF vc IS $\begin{matrix}\text{(LT)}\\ \text{EQ}\\ \text{(GT)}\end{matrix}$ vc, GO TO n - In this statement a test is made to determine if the left variable or constant is less than (LT), equal to (EQ), or greater than (GT) the right variable or constant. If the statement is found to be true, then the module branches to POINT n. If the statement is not true, the next statement following the IF statement is executed.

GO TO n - This statement causes a branch to POINT n. The next computation to be performed immediately follows the POINT n. Note that a GO TO should be followed by a POINT statement or be at the end of the module since there is no way of getting to a statement which follows a GO TO unless it is a POINT statement.

READ - This statement will cause data to be read into the next module before it is run. The rules for the data statements are specified elsewhere. (No specification is made as to what or how much data is to be read, this is defined on the input cards.)

PRINT v, v, v - This statement causes the values of the variables to be printed and identified in a standard format, five to the line.

Module Statements

The module contains primarily computation and a minimum of logical instructions. The allowable statements and their descriptions are as follows:

```
MODULE NAME = n

v = general expression

v = TABLE table name (vc, vc, vc)

ITERATE ON v, ATOL = vc, PTOL = vc, TIMES = c

INITIALIZE v = c, v = c

COMMON v, v, v...
        (LT)
IF VC IS EQ  vc, vc = general expression
        (GT)

PRINT v, v, v

NO ORDER
```

MODULE NAME = n - This statement gives a name to the module. Every module must contain this statement. The name can consist of from 1 to 20 alphanumeric characters. This card is usually the first one in a module but it can be placed anywhere.

v = general expression - This is the basic statement and as such, there is a whole section devoted to it. But a few rules are: the minimum requirements are a variable, an equals sign and either a variable or a constant. Usually, the general expression is any algebraic expression which conforms to standard mathematical usage and consists of addition, subtraction, multiplication, division, and exponentiation. See the section titled Syntax Rules for Expressions.

v = TABLE table name (vc, vc, vc) - This statement is used for table look up. Up to three arguments are allowed but only one is necessary. The number of arguments must coincide with the size of the table. The value returned is a result of a linear interpolation. When an argument exceeds the limits of a table, a warning message is printed. The rules for a table name are the same as for a variable. All tables are essentially in COMMON and thus can be shared by all modules.

ITERATE ON v, ATOL = vc, PTOL = vc, TIMES = c - This statement is the same as used in the Control Module with the single and very important exception of the absence of the FROM specification. In a module the whole module is iterated upon except the INITIALIZE statement. The location of this statement in the module is not important.

INITIALIZE v = c, v = c...up to 17 variables per card - The initialize statement may be used in a module which contains an ITERATE statement. Before the first iteration the variables specified are initialized to the specified values.

COMMON v, v, v... - This statement is used to designate variables which are used in or come from other modules. The rules are the same as for the Control Module statement. Note that only variables which are used in the module need to be listed and that the order is not significant.

IF vc IS $\begin{matrix}\text{(LT)}\\ \text{EQ}\\ \text{(GT)}\end{matrix}$ vc, v = general expression - This statement is similar to the IF statement of the Control Module except that a general algebraic statement is evaluated if the logical statement is true. This statement is often used for limiting conditions: for example, a stall limit might be imposed by the following statements

```
VMAX = TABLE VEL (POW, POW)
IF VMAX IS GT VSTL, VMAX = VSTL
```

which says that VMAX will not be greater than VSTL.

PRINT v, v, v... - up to 40 variables per card - The statement is the same as above except that regardless of where it is located in the module the data is not printed until the module has been run.

NO ORDER - Normally a module is run with the statements in the same order as they were read, and if the sequence must be changed an error message is produced. This statement allows the sequence to be changed without producing an error message.

Data Statements

Data can be entered following any of the modules (including the Control Module) when the model is first loaded into the computer. This is done with no special instruction. For data to be read by a module during the running of the program, the RUN MOD n statement is preceded by a READ statement. At no time does the user have to specify in the model how much or what data is to be read.

For input of variables the following form is used on the cards:

v = c, v = c, v = c...

There may be up to twenty variables per card. The commas are required to separate the statements. The following example is an illustration of two cards of data.

ALT = 2000, OAT = 35., GW = 10,000

DL = 8.5, VRC = 500.

Only variables which are referred to in a module (or in its COMMON statement) may be entered.

Input of tables requires somewhat more care. The first card must be of the form:

TABLE NAME = n, SIZE = (c, c, c)

This statement is used to assign a name to a table and to define its size. This statement must be followed by the data for the table. The name must conform to the same rules as a variable name. The size need only contain the number of arguments needed to specify the array size; for instance, if a table has 10 "x" values and 3 "y" values, the size would be (10,3). The order in which the data is input is given below. The data cards are of the form:

c c c c

Each card contains up to 8 constants. This form is used only for table data input. This card is the only one with a specific format. Each constant must be contained in a field of 10 columns, 1-10, 11-20, etc. Each constant must also contain a decimal point. Otherwise the rules are the same as for a normal constant. The order in which the data is input will now be described. x, y and z correspond to the first, second and third arguments in a v = TABLE n(x,y,z)

statement. Consider a TABLE NAME statement with SIZE = (i, j, k). The first thing to be input would be i values of x followed by j values of y and then k values of z. Then the table is read in with the first index varying first, then the second, then the third. That is, the elements of a (3,4) table would be read in this order: (1,1), (2,1), (3,1), (1,2), (2,2),... A new card must be started whenever one of the indices returns to 1. The values of x, y, z must be in ascending order. They may be positive or negative and the increments need not be constant.

The following is an example of a table which has three x values and four y values.

	5	2.6	3.4	5.8
	2.5	2.1	3.2	5.5
Y	1	1.9	2.85	5.4
	0	2.0	3.0	5.5
		0	10	20
			X	

The data cards would be as follows:

Card #	Contents				Comments
1	TABLE	NAME = EXMP, SIZE = (3, 4)			
2	0	10.0	20.0		x values
3	0	1.0	2.5	5.0	y values
4	2.0	3.0	5.5		
5	1.9	2.85	5.4		
6	2.1	3.2	5.5		Table
7	2.6	3.4	5.8		

SYNTAX RULES FOR EXPRESSIONS

The rules for general expressions are essentially the same as for any algebraic expression (and are compatible with FORTRAN). Two things to keep in mind are that for multiplication, the multiplication operator must separate the two variables, and since equations are written on one line extra parentheses are sometimes required when dividing.

There are five operations which are allowed, the operation and their symbols are as follows:

	Operation	Symbols
1.	Addition	+
2.	Subtraction	-
3.	Multiplication	*
4.	Division	/
5.	Exponentiation	**

There are two other symbols which are allowed in a general expression. They are left and right parentheses which are used for associations and must be used in pairs.

The following are the rules for general expressions (these are a formalization of ordinary algebraic form):

1. All variables and constants, except the first must be preceded by an operator or a left parenthesis. The first may be preceded by a left parenthesis or a sign.

2. All variables and constants, except the last, must be followed by an operator or right parenthesis. The last may be followed by a right parenthesis.

3. A left parenthesis must be preceded by a left parenthesis or an operator unless it is the first symbol in the expression, in which case it may be preceded by a sign.

4. A left parenthesis must be followed by a left parenthesis, a variable, a constant, or a sign.

5. A right parenthesis must be preceded by a right parenthesis, a variable, or a constant.

6. A right parenthesis must, unless it is the last symbol on the card, be followed by a right parenthesis or an operator.

7. An operator must be preceded by a variable or a right parenthesis.

8. An operator must be followed by a left parenthesis or a variable.

The following are examples of general expressions which conform to the rules.

```
A + B

A*B + C**D

(A + B)**1.3 + (2*(3 + B - C))**(2*K)

(A + B + C)/(D + E + F)

(TIM + PHSE)* PIE*2

(((X + 3)*6 + 2)*B + C)**.5

6*A + 3*B + 5/C
```

The order in which operations are performed are:

1. Exponentiation

2. Multiplication and Division

3. Addition and Subtraction

The order of evaluation is from left to right, with parenthetical expressions being evaluated first and then the expression as a whole. Note that the division operator applies only to the variable or parenthetical expression immediately following it, i.e.,

$$A/B*C = \frac{AC}{B}$$

PROGRAM OPERATION

The first deck to be loaded must be the control module. This is followed by any appropriate data. Then the individual modules are loaded, each optionally followed by a block of data. These modules may be loaded in any order. The order in which they are used is solely determined by the statements of the control module. After the last module is input, appropriate blocks of data follow in the order in which they will be called by READ statements during execution of the program.

Each of the above units must be separated by an "end-of-file" card which contains an /* in columns 1 and 2. Thus the input deck will appear as follows:

```
(Control Module)
/*
(Data)
/*
(Module)
/*
(Data)
/*
(Module)
/*
:
(Last Module)
/*
(Data)
/*
/*
(Data)
/*
(Data)
/*
:
```

Data input is assumed after each module. Even if no data are entered then the two /* cards must still appear:

```
:
(Module)
/*
/*
(Module)
:
```

After the last module is entered, two /* cards instruct the computer to start running the model. If the last module is not followed by input data, then:

```
:
:
(Module)
/*
/*
/*
```

PROGRAM LIMITATIONS

The following limits exist in the present version of ZODIAC II.

Common variables	- 200 maximum
Noncommon variables	- 200 maximum in a module
Constants	- 200 maximum in a module
Equations	- 100 maximum in a module
Tables	- 20 maximum
Modules	- 30 maximum
Iterate statements	- 4 maximum in a module

Constants may not contain more than fifteen characters.

ERROR CODES

During operation of the program, certain violations of rules or apparent violations of logic may occur. When this happens a coded error message is printed. The codes are given in Table IX.

TABLE IX. ERROR CODES

Code	Meaning
1	Number of common variables exceeds 200
2	Number of constants exceeds 200 in module
3	Number of equations exceeds 100 in module
4	Missing parenthesis
5	"=" missing or incorrectly placed
6	Name length on data card too long
7	Constant exceeds 15 characters
8	Number of noncommon variables exceeds 200 in module
9	Undefined variable in module
10	Equations reordered - possible error
11	Name of argument in table lookup too long
12	More than 3 arguments in a table lookup
13	Invalid character
14	Incorrect symbol
15	More than one "=" on card
16	---
17	Undefined "FROM" on iterate card
18	No name on a MODULE NAME card
19	Incorrect argument on IF card
20	Incorrect operation on IF card

TABLE IX - Continued	
Code	Meaning
21	Incorrect format on IF card
22	More than 4 ITERATE statements in a module
23	More than 30 modules
24	Illegal command
25	Number cf iteration exceeds allowed number
26	Illegal statement in control module
27	More than 20 tables
28	Number of arguments in a table lookup is inconsistent
29	Table lookup requires extrapolation
30	Error on table input
31	More than 18 variables initialized in module
32	Invalid operator on a control module card
33	Name too long
34	Undefined variable in control module

In general, the program will continue to run even though errors are found. Invalid variables are ignored and other assumptions are made which will allow the completion of the computation. Care must be exercised since some computations with errors will be valid, some will be completely invalid, and some will have limited meaning. This approach, however, is expected to minimize the time required to verify a program.

The only error which will terminate a run is no. 34.

SUGGESTIONS FOR THE NEW USER

Gaining Experience

A good way for the new user of ZODIAC II to gain familiarity with the usage of the program is to first modify existing models, rather than attempting to develop a complete new analytical model.

Changes within individual modules corresponding to mission definition changes, effectiveness criteria, statistical weight analysis, etc., are especially straightforward. It is recommended that each equation be identified by making use of the comment capability on each card.

A next step in the familiarization process would be to add or delete modules and make the appropriate changes in the control module statements.

Making changes in the logic within the modules and especially in the control module will complete the process of familiarization. While these modifications are being performed, computer runs using various combinations of data variations should be conducted.

User Techniques

One of the major objectives of ZODIAC II was to make the model formulation virtually self-explanatory to the engineering user. This has been achieved providing the user does not artifically complicate his model by allowing numerous artificial choices between options, only one of which will be used at one time. For example, it would be tempting to a conventional programmer, when developing a module to estimate drag, to include the relationship for all the types of helicopters and selecting the appropriate one by a coded input such as "1" = cargo helicopter, "2" = utility helicopter, etc. This kind of programming is to be discouraged because it tends to obscure the analytical model.

It is strongly suggested that, when analyzing different helicopters, the proper equations are placed in the proper modules. These changes are easy to make and the engineer will always know what model he was using. This, of course, is the objective of this program.

One of the great advantages of this program is that computations can be easily performed that were not envisioned by the original programmer. It is suggested that the engineering users do not hesitate to increase the sophistication of the computations by adding new modules, modifying old ones, using new types of data.

Another advantage of this program is that it is possible to vary any parameters without having made prior provision in the basic program. For example, suppose it were desired to study several combinations of number produced (NP), monthly flight hours (MFH) and yearly attrition rate (YAR). In this case the control module could be preceded by a POINT and READ statement and followed by a PRINT and an ITERATE statement as follows:

```
POINT NEW
READ


Original control module

PRINT NP, YAR, MFH
ITERATE TIMES = 10, FROM NEW
```

This will cause data to be read in ten times and the complete computation repeated. In reading data, note that the last used values are in effect until a new value is computed or read in as data. Thus, the data package following this modified program could be as follows:

```
:
/*
/* (end of model input)
NP = 1000, YAR = 0, MFH = 100
/*
MFH = 300, NP = 100
/*
YAR = 10
/*
YAR = 50, NP = 100
/*
NP = 500
/*
YAR = 10, MFH = 200
/*
etc.
```

Thus, it is seen that by adding the few cards shown above, it is possible for the engineer to vary any parameters he wishes and in any manner he wishes.

There is an area where the user must use some caution. It was intentional in the design of ZODIAC II that no argument list be used (as in FORTRAN), and that all interchange of data between modules be carried out through the common variables. This technique helps to minimize the risk of making local changes in modules. However, if the same module is used at more than one point in the control module, some of the same variables will change their values more than once during the computation. This is no problem unless it is desired to use one of the results of an early use of a module after the module has been run again. The same effect occurs if the same variable is computed in different modules. For example, consider the computation of fuel load, WFL, at different points in a mission where each segment makes a computation of the form WFL = WFL + DELTA.

```
RUN MOD  TAKE OFF
RUN MOD  CLIMB
RUN MOD  CRUISE
etc
```

If it is desired to know at some later point the fuel used after takeoff but before climb, this can be accomplished by naming a new variable and inserting the simple equation as follows:

```
RUN MOD  TAKE OFF
 WFLO = WFL
RUN MOD  CLIMB
etc
```

WFLO will retain its value as desired. WFL must be in a COMMON statement in the control module.

Techniques such as suggested above will be found to be readily picked up by engineers without any conventional programming experience in a short time.

METHOD APPLICATIONS

UTILITY MISSION

As an application of the techniques and computer program discussed, a number of conditions have been analyzed. Most of the data presented here is for the utility mission as described in Tables I and II. In addition to the data discussed previously, the following parameters have been used in these computations:

NP (no. of ship produced) = 1000
YAR (yearly attrition rate) = 40
SL (system life) = 10 years
WCR (crew weight) = 400 lb
MFH (average monthly flight hours) = 100 hr
PM (power margin) = 5 percent
NEN (number of engines) = 2
DL (disk loading) = 8 lb/ft^2
TS (main rotor tip speed) = 700 ft/sec
BL (blade loading) = 80 lb/ft^2
NRM (no. of main rotor blades) = 4
SM (airspeed margin to stall) = 10 kt
RCP (fuel reserve) = 10 percent

The first design point was taken as 4000 ft, 95°F, 500 ft/sec rate of climb. "Current" SFC data was used (See Figure 3).

For the statistical weight analysis the following assumptions were made (see Table III):

AG = 1 (no. of auxiliary landing gears)
BF = 1 (blade folding included)
BRK = 1 (main rotor brake included)
ITR = 1 (intermediate tail rotor gearbox included)
KLG = .0329 (tricycle landing gear)
KNAC = .96 (nacelle for twin engines mounted to transmission)
NR = 1 (no. of main rotors)
NULT = 4.5 (ultimate load factors)
TAF = 13 (type of aft fuselage (see Table III))
TPU = 0 (no auxiliary power unit)
TPY = 62 (type of pylon configuration (see Table III))

The ground rules used are as follows:

1. The disc loading at the second design point is the same as for a single-point design helicopter. This results in a reduced disc loading at the first design point for the two-point design helicopter.

2. The rate of climb specified for the single-point design helicopter is also obtained for both the second and first design points of the two-point design helicopter.

3. The "probability of hover" calculation includes the requirement to achieve the above vertical rate of climb.

4. All designs include the fuel load required for the standard mission.

The second design points studied included all combinations of altitude from 0 to 4000 ft in increments of 1000 ft and temperatures from 20°C to 35°C in increments of 5°C.

Payload Utilization

A payload utilization function was used as follows:

Payload (% of 2nd Point Payload)	% Utilization
40	15
60	25
80	30
100	25
110	5

The model listing is given in Appendix II.

Weight Variation

The variation of the gross weights of the two-point design helicopters is shown in Figure 19 for combinations of design altitude and design temperature. This figure shows the gross weight at point two, which is the maximum gross weight at the second design point making use of all the power available at this condition. At this point, the helicopter can hover and carry out the specified vertical rate of climb (500 ft/sec).

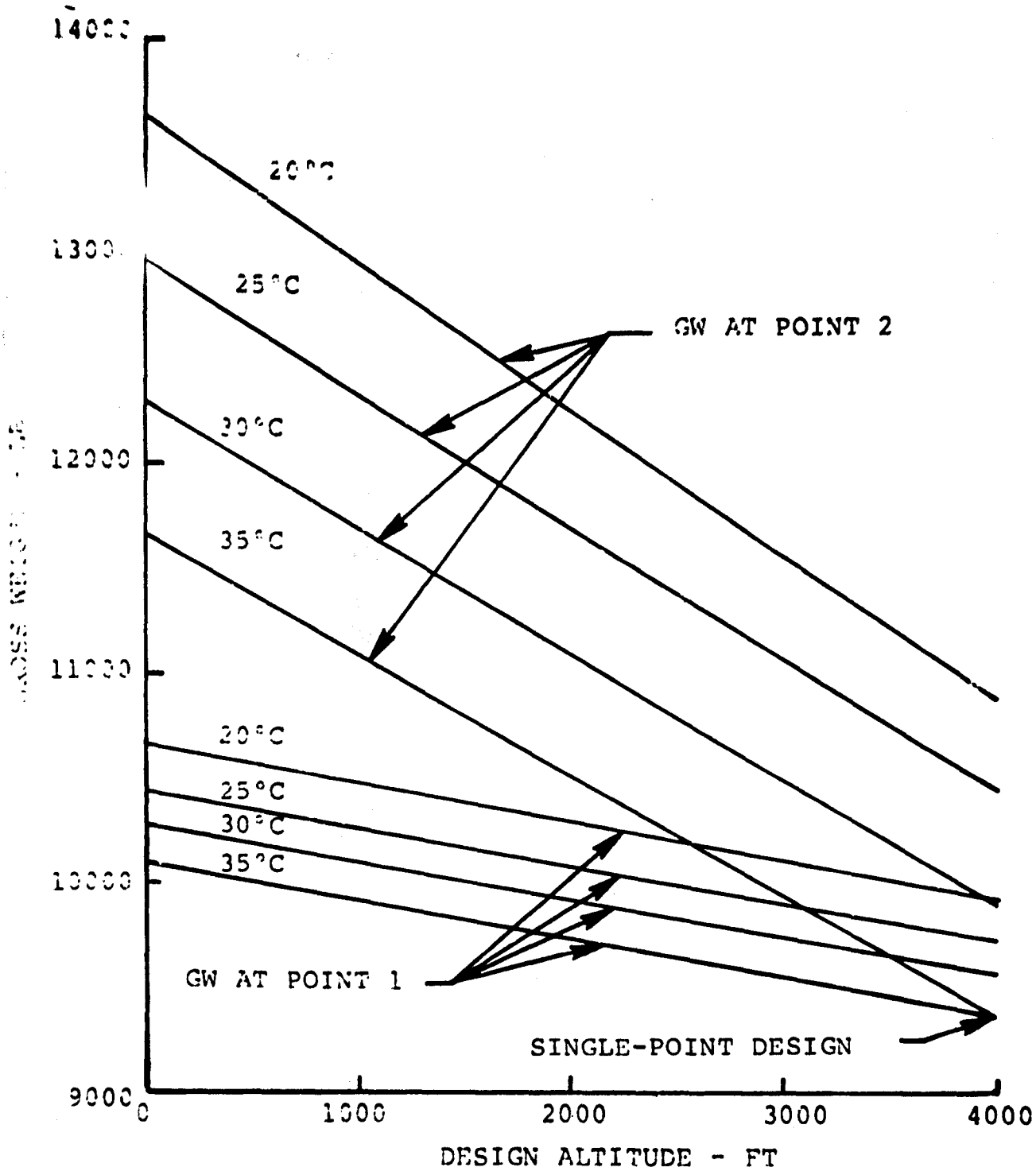


Figure 19. Two-Point Design Gross Weights for Utility Mission.

Also shown in Figure 19 is the resulting gross weight at point one, that is, with the mission payload. The changes in these weights are due to the changes in empty weight and mission fuel requirements. The gross weight at the second design point includes, in addition, the increased payload.

Figure 20 illustrates the payload capability as a function of the second design point. Note that at point one, the standard mission payload applies to all second design points.

Engine and Transmission Ratings

Figure 21 illustrates the effect of second-point design selection on transmission limits. As the second-point design temperature and altitude are reduced, the engine power available at the second design point is increased. The transmission rating is increased to match this power level, and the engine is derated less and less.

Figure 21 also indicates an effect of second-design point conditions on required power rating. This is a secondary effect and results from empty weight changes as the second-design point capability is added to the helicopter. As the second-design point temperature or altitude are decreased, the empty weight increases, and the first design point requirements can no longer be met with the single-point design engine.

Cost Variation

The computed cost per flight hour is shown in Figure 22 for several payloads. This cost includes initial costs which are primarily a function of empty weight and operating costs which reflect the effect of the ratio of actual gross weight to design (second point; gross weight. For small payloads the costs are only slightly sensitive to the second design point; however, as the payload increases this sensitivity also increases. This figure shows typical data obtained.

Hover Probability

The cumulative joint probability of hover is an extremely important factor in the overall cost effectiveness computation. It is a factor which is very sensitive to the assumed environmental operating conditions of the helicopter. It is also a quantity which tends to reduce the cost effectiveness of more stringent design conditions. As an illustration of this effect, at a 100-percent payload condition the single-design point helicopter can hover (and climb) 84 percent of the time. That is, 84 percent of the time the environmental

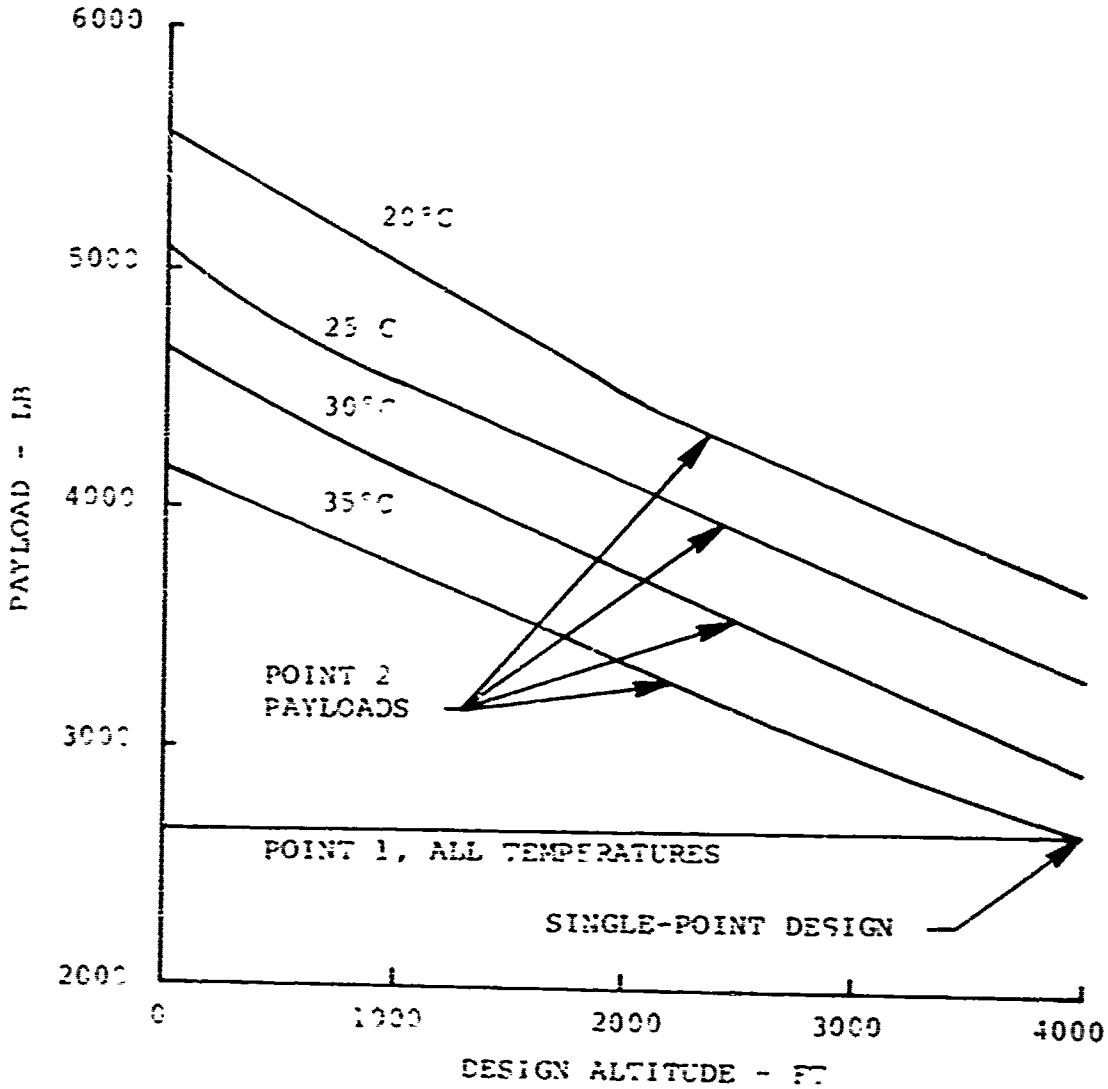


Figure 20. Two-Point Design Payloads for Utility Mission.

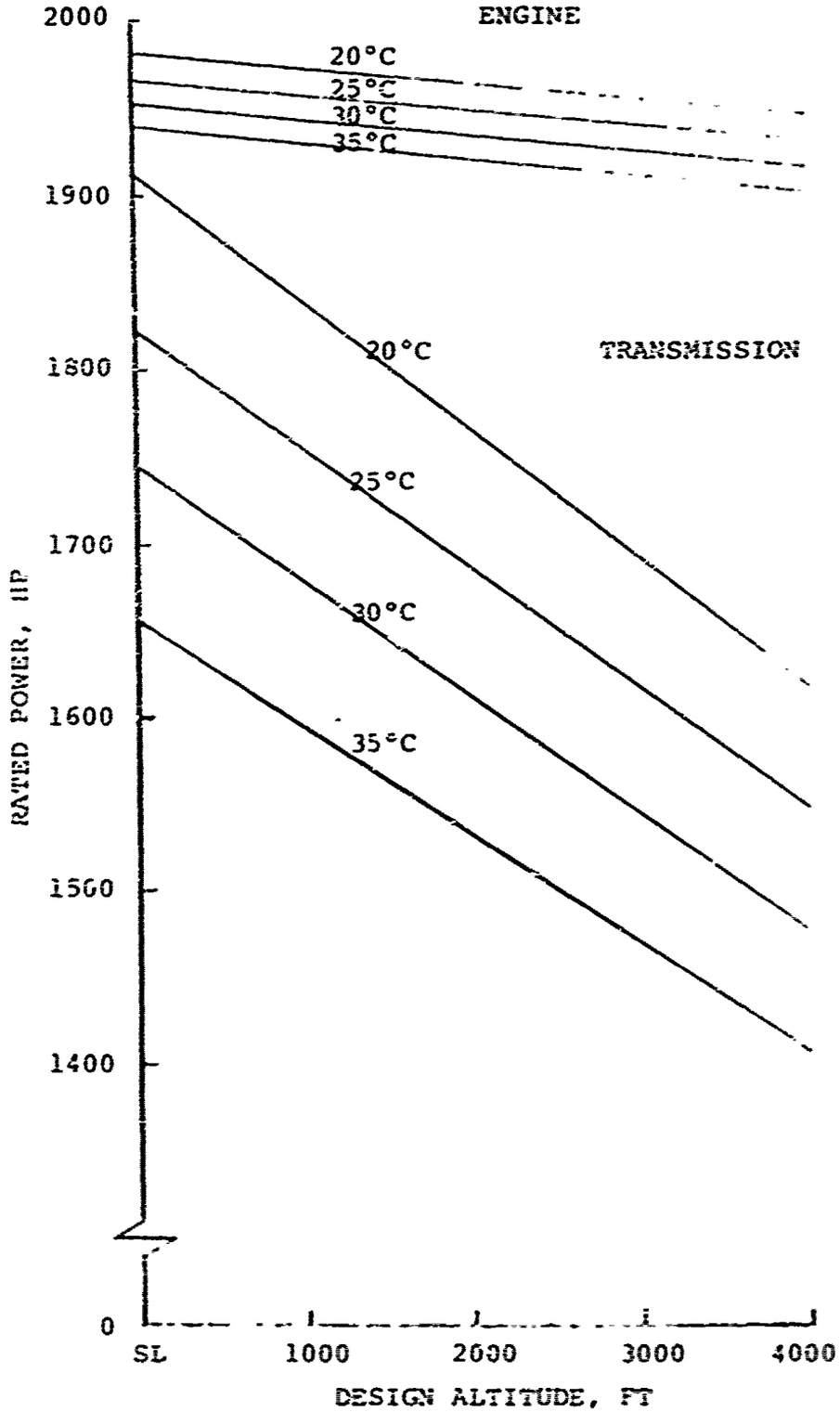


Figure 21. Engine and Transmission Ratings for Two-Point Design Helicopters.

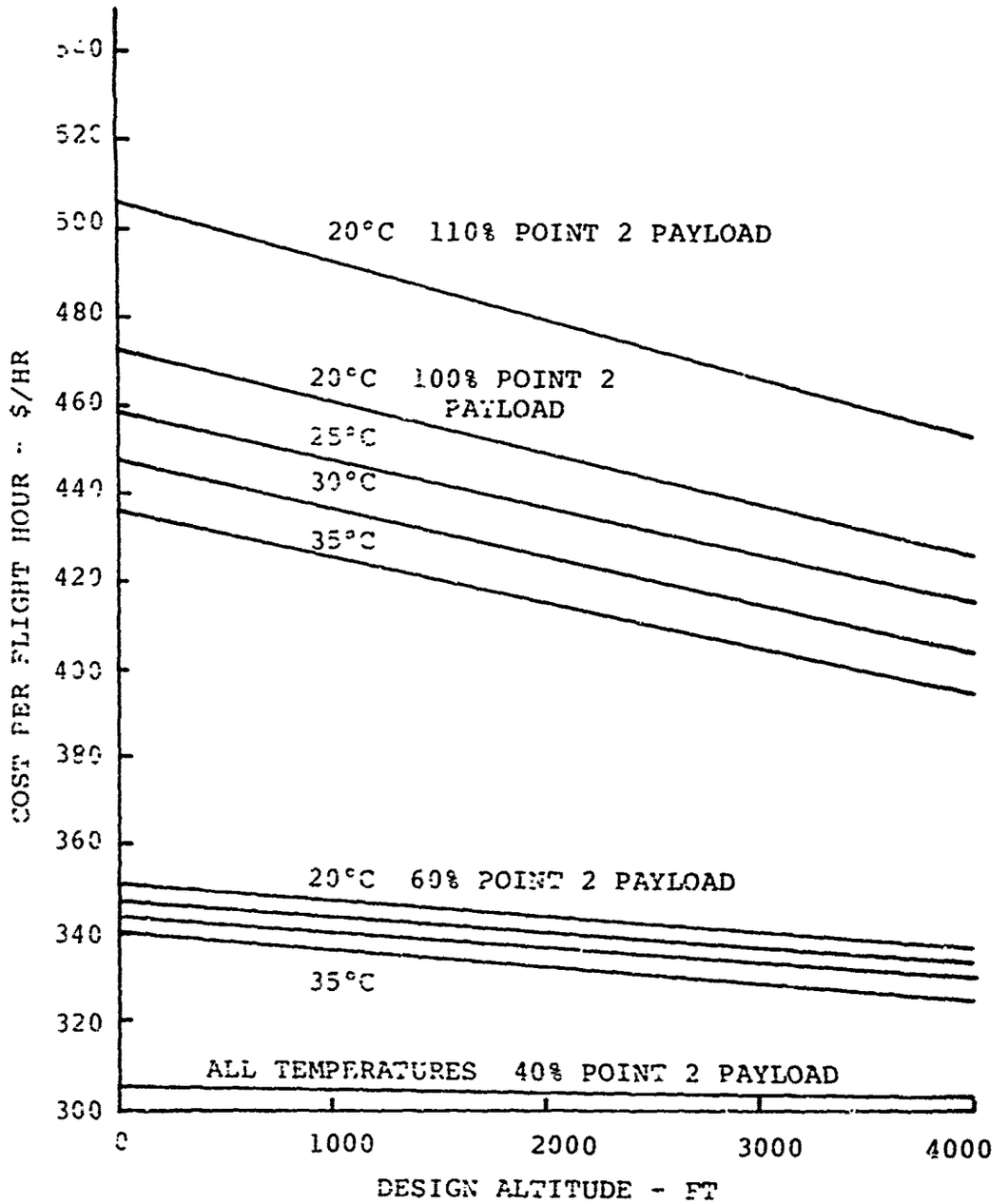


Figure 22. Typical Cost Per Flight Hour Results for Utility Mission.

conditions can be expected to be less stringent than 4000 ft, 35°C. However, a single-point helicopter designed for 2000 ft, 25°C, for example, will be able to hover with its design payload only 34 percent of the time since over much less of the area and time will the operating conditions be less stringent than this design point. Of course, the two-point design helicopter will be carrying larger payloads so there is a rather delicate balance between these two factors. Figure 23 (a-d) presents the probability of hover obtained.

Overall Cost Effectiveness

The final results, including the effects of assumed payload variation, costs, and hover probability, are given in Figure 24. It is seen that local peaks occur at certain temperatures. The optimum point calculated appears to be at 4000 ft, 20°C representing about a 10-percent increase in cost effectiveness compared to the single-point design.

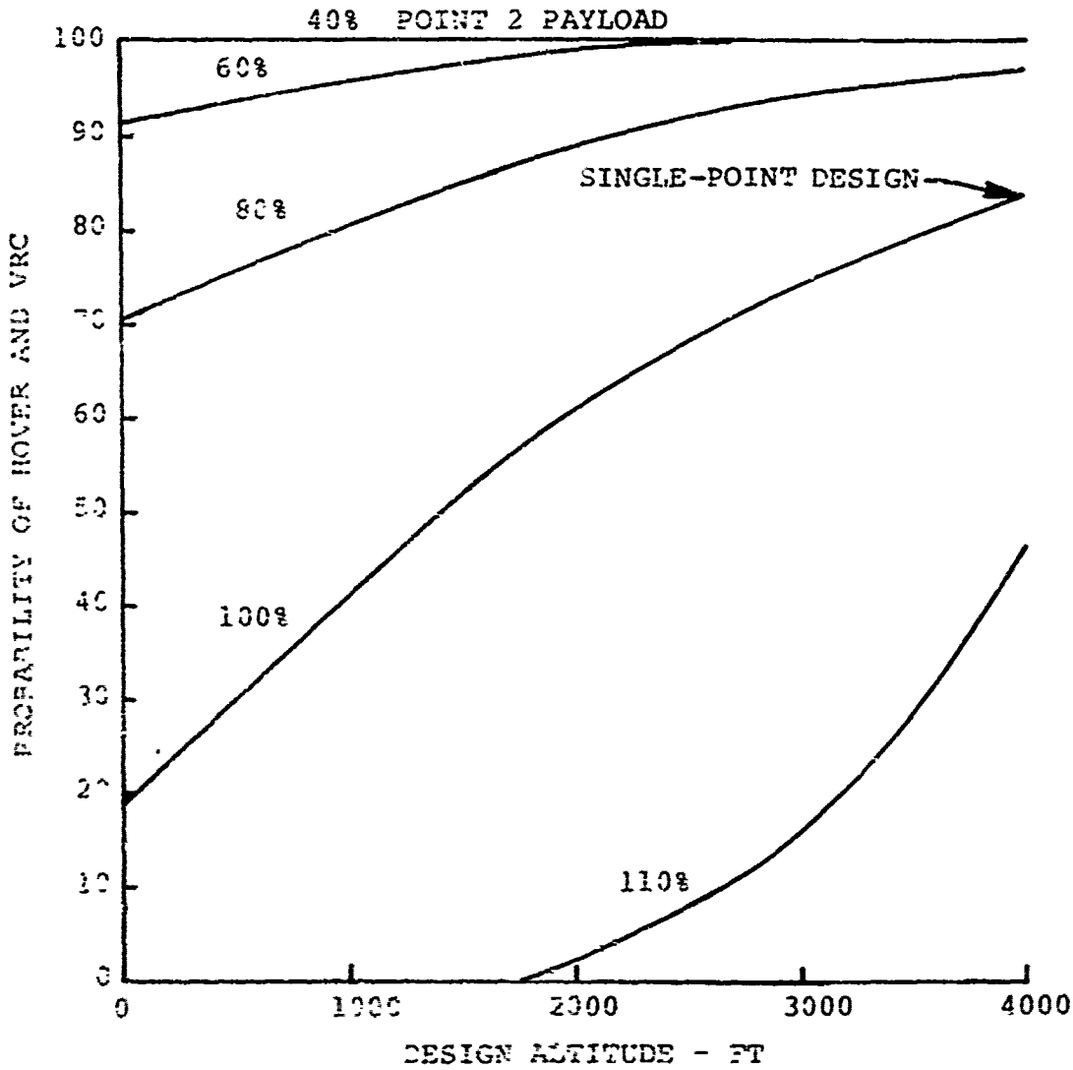
This "optimum" helicopter has an increased design gross weight of about 15 percent and an increased payload at the second design point of about 40 percent.

It must be emphasized, however, that these results are sensitive to the predefined mission, assumed number of production units, assumed payload utilization, environmental statistics, and numerous other factors which will vary with the particular requirements for the vehicle under study.

WEIGHT SENSITIVITY

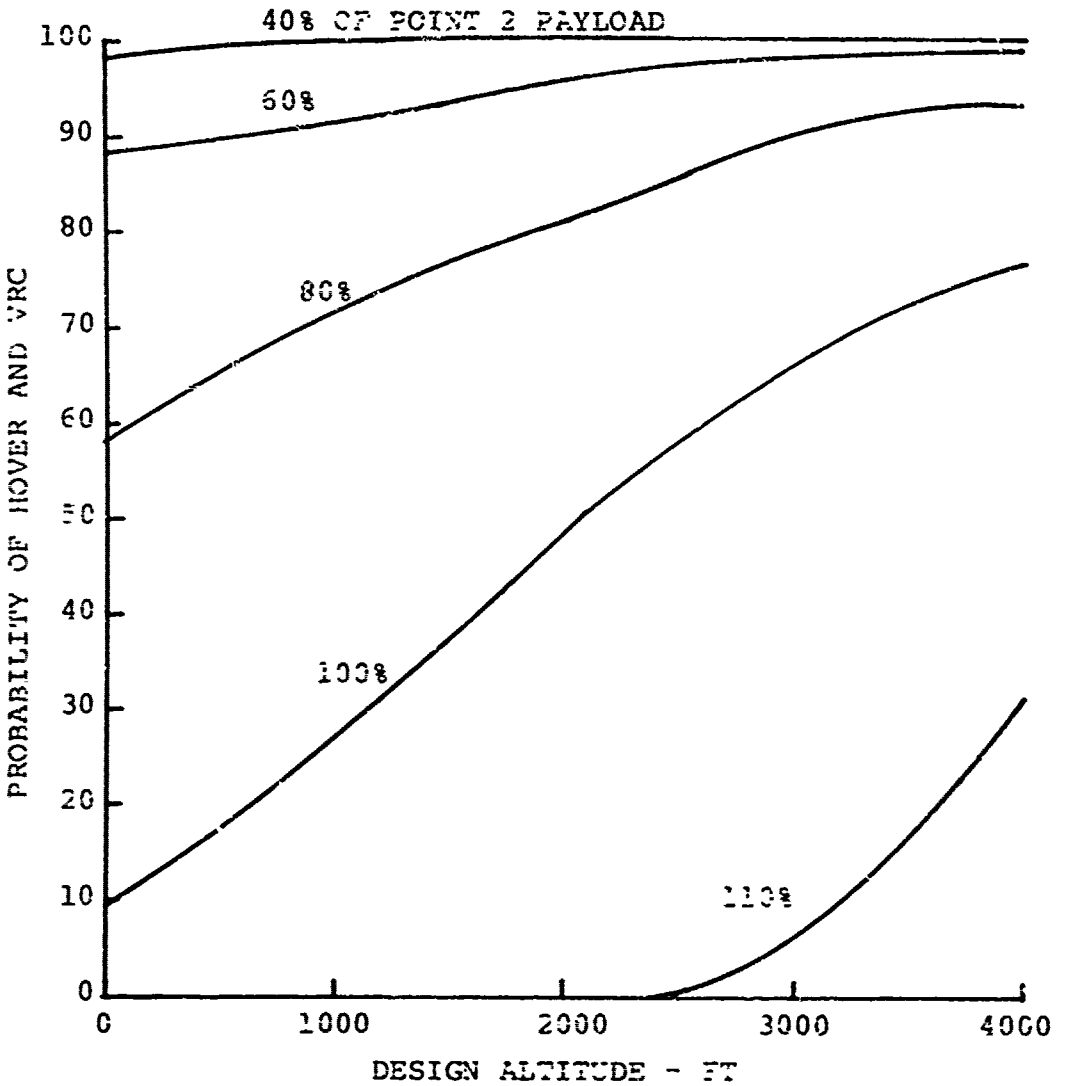
One of the most significant portions of the analytical model is the statistical weight model. In order to gain some insight into the sensitivity to the empty weight, some of the previous computations were repeated with an arbitrary 20-percent increase in predicted empty weight. For comparative purposes the payloads are shown in Figure 25. A comparison of the cost per hour is shown in Figure 26.

The change in probability of hover with increased weight is interesting. At payloads below 100 percent, the heavier helicopter has lower probability of hovering. At 100 percent payload (note that the payloads are not the same), the probabilities are equal. At over 100 percent the heavier helicopter has increased probability of hover. This effect is illustrated in Figure 27 for 25°C. The effect is the same at other temperatures.



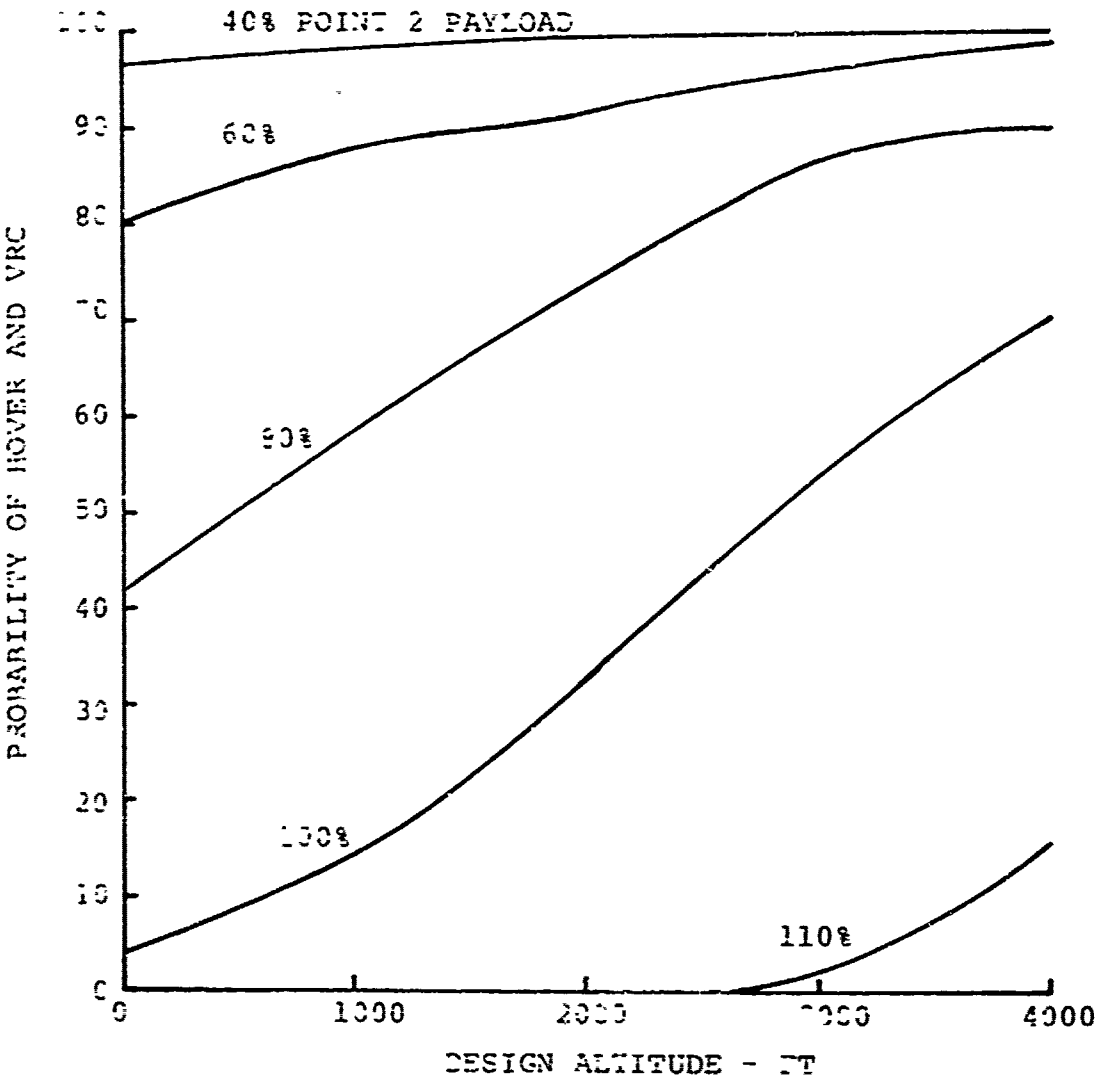
a. Design Temperature = 35°C

Figure 23. Probability of Hover and VRC for Utility Mission.



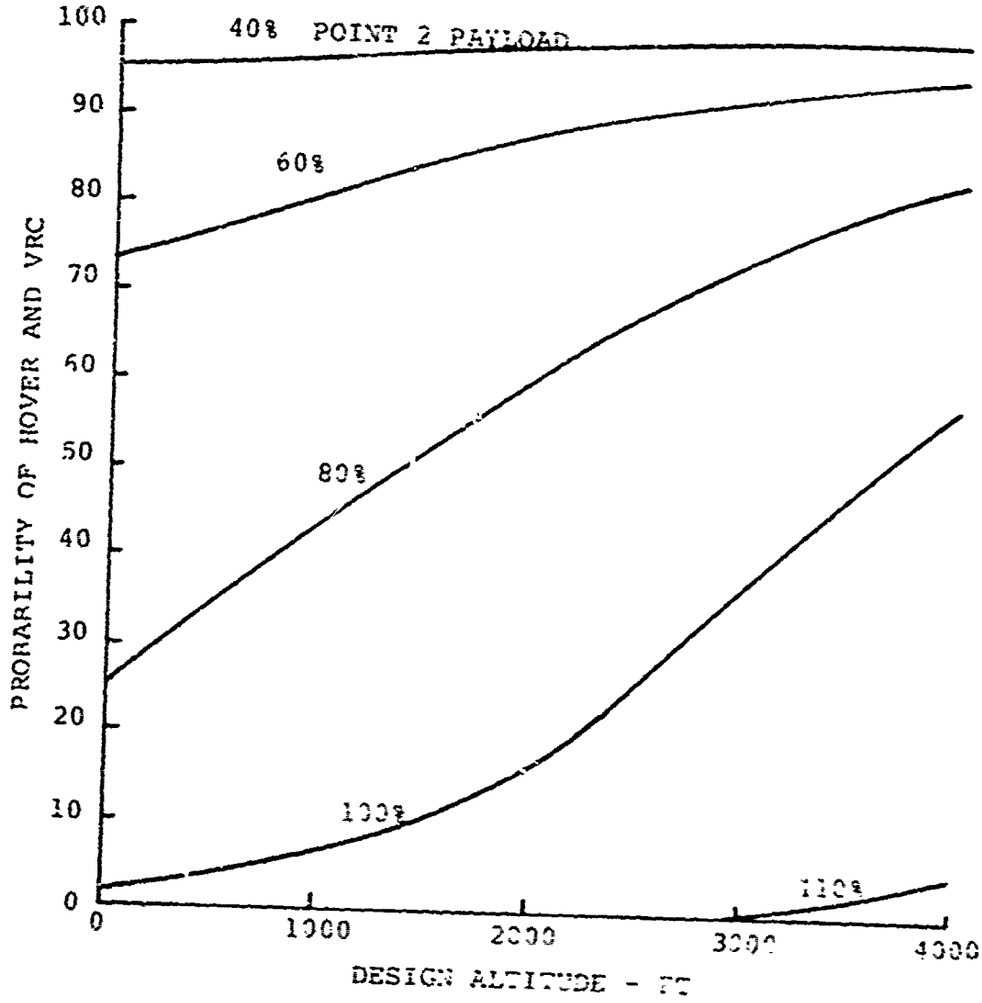
b. Design Temperature = 30°C

Figure 23 - Continued



c. Design Temperature = 25°C

Figure 23 - Continued



d. Design Temperature = 20°C

Figure 23 - Continued

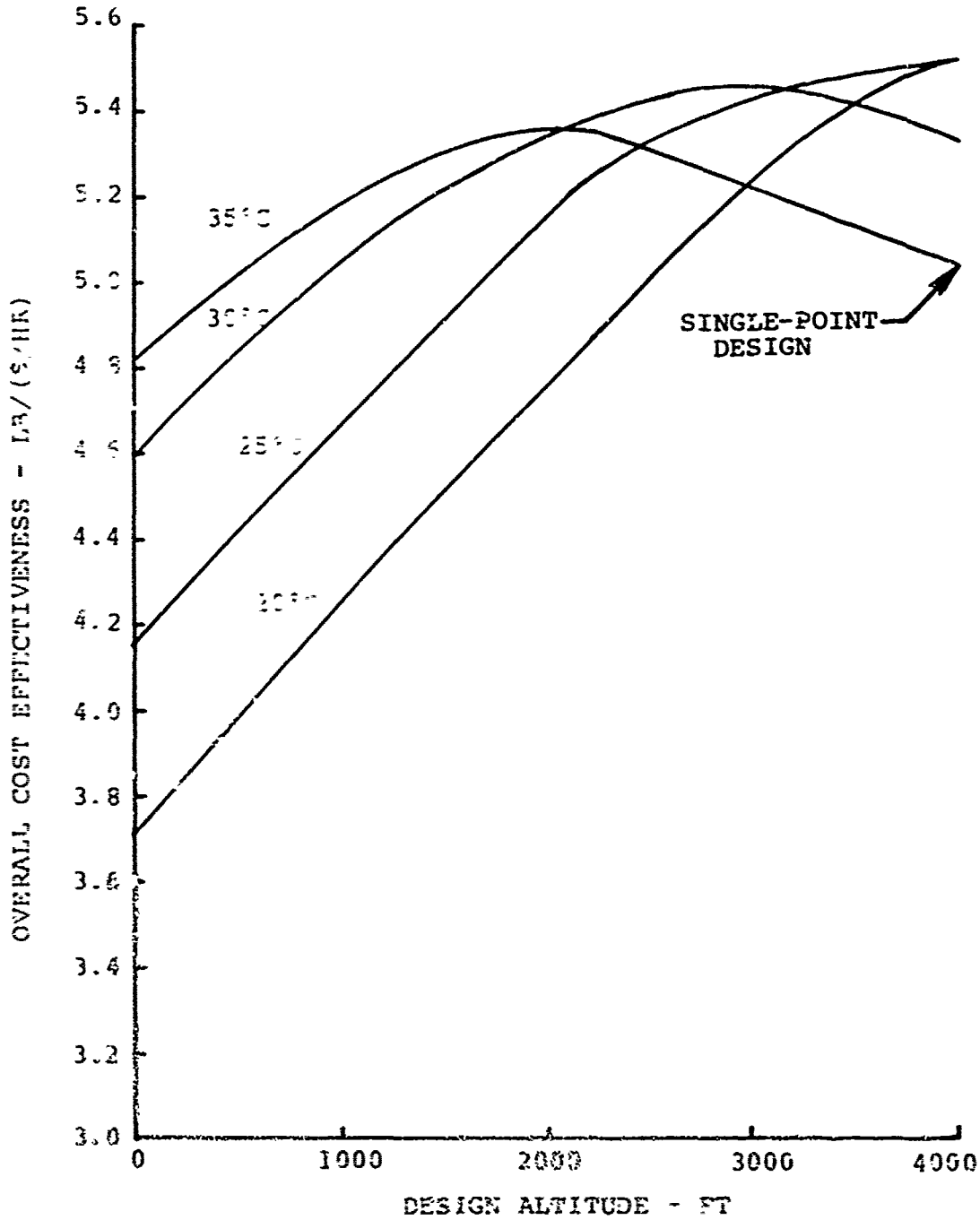


Figure 24 Overall Cost Effectiveness for Illustrative Utility Mission.

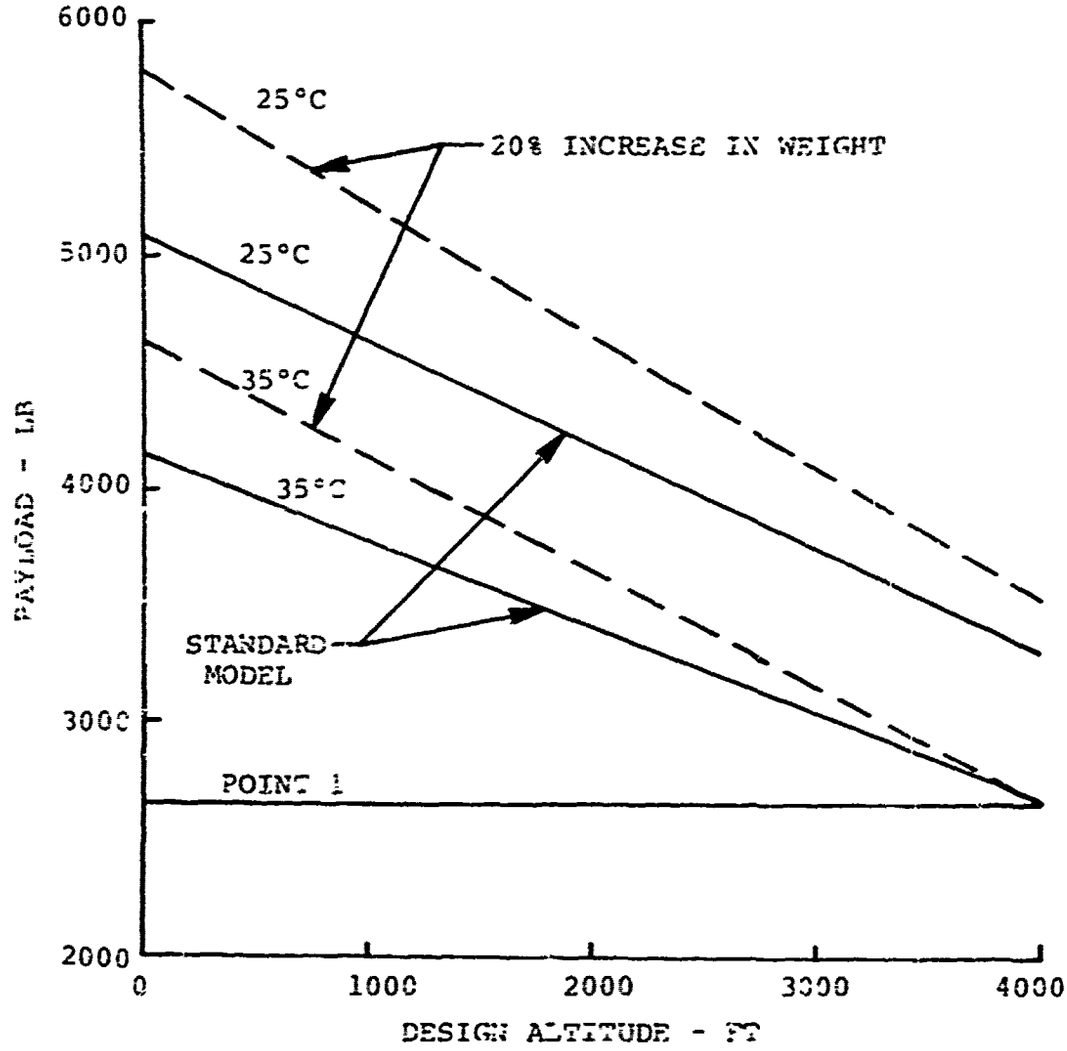


Figure 25. Effect of 20% Increase in Empty Weight Model on Payload - Utility Mission.

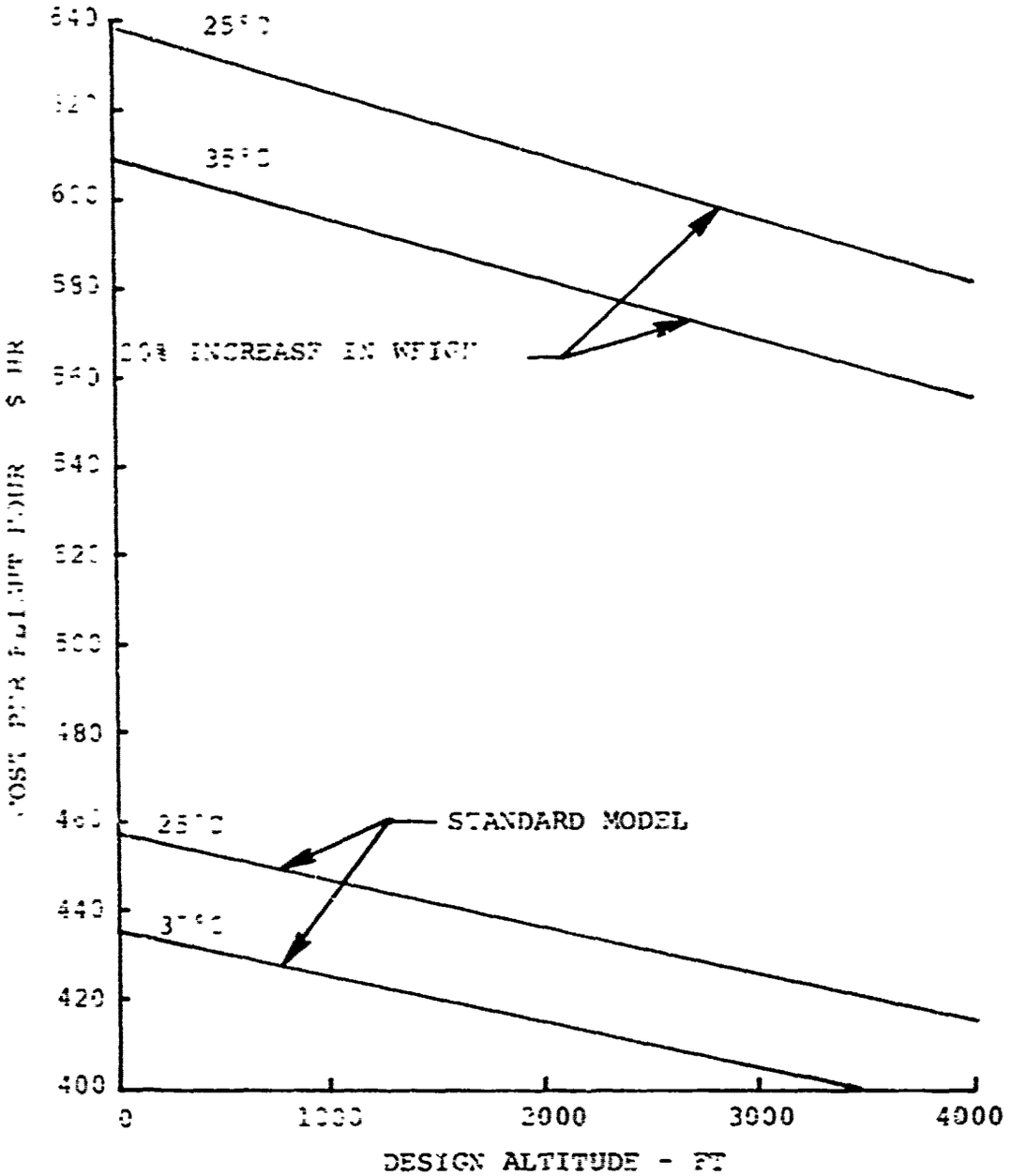


Figure 26. Effect of 20% Increase in Empty Weight Model on Cost Per Hour - Utility Mission.

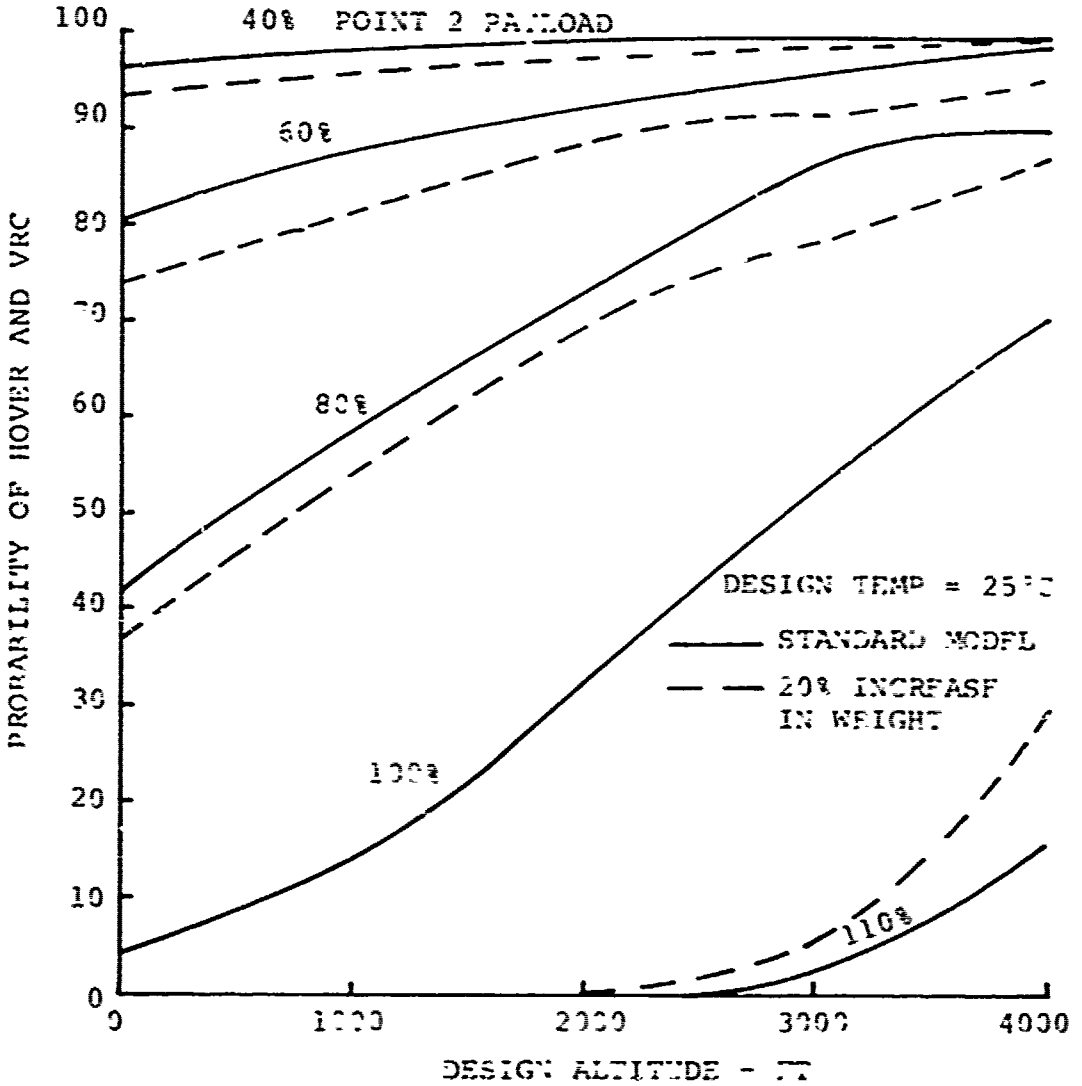


Figure 27. Effect of 20% Increase in Empty Weight Model on Probability of Hover - Utility Mission.

The overall cost effectiveness is shown in Figure 28. While the effectiveness is reduced because of increased costs and reduced probability of hover (for 70 percent of the time), the curves have very similar shapes.

PAYLOAD UTILIZATION EFFECTS

The previous analysis used a payload distribution based on percentages of the maximum. This implies that the size of the payload will always depend on the capability of the helicopter. At the opposite extreme is the concept of missions having no relationship to the maximum capability of the helicopter but having a requirement for the transportation of specific payloads. To consider the effect of such a payload distribution the same schedule of payload was used, except that the payload was a percentage of the fixed first point design payload rather than the variable second point payload.

While the first approach resulted in a highly loaded helicopter, this approach results in lightly loaded helicopters. In general, the probability of hover is significantly increased and the costs are reduced because of reduced maintenance. Since the same payloads are carried at reduced operating costs, the cost effectiveness tends to increase at the more stringent design points. This effect is illustrated in Figure 29. Note that the data of Figures 19 and 20 also apply to this condition.

This result, in effect says that for a given fixed payload, the larger the helicopter the less the cost will be because the reduced unscheduled maintenance is the dominating factor. This appears unrealistic and suggests a flaw in the cost model.

In actuality, a distribution which is partially fixed and partly dependent on the helicopter's capability is probably more realistic. If we take as an example a 50-percent mix of the two types discussed, the curves on Figure 29 are the result. A comparison of the curves of Figure 29 and Figure 24 illustrates how sensitive the optimization can be to the predicted payload schedule.

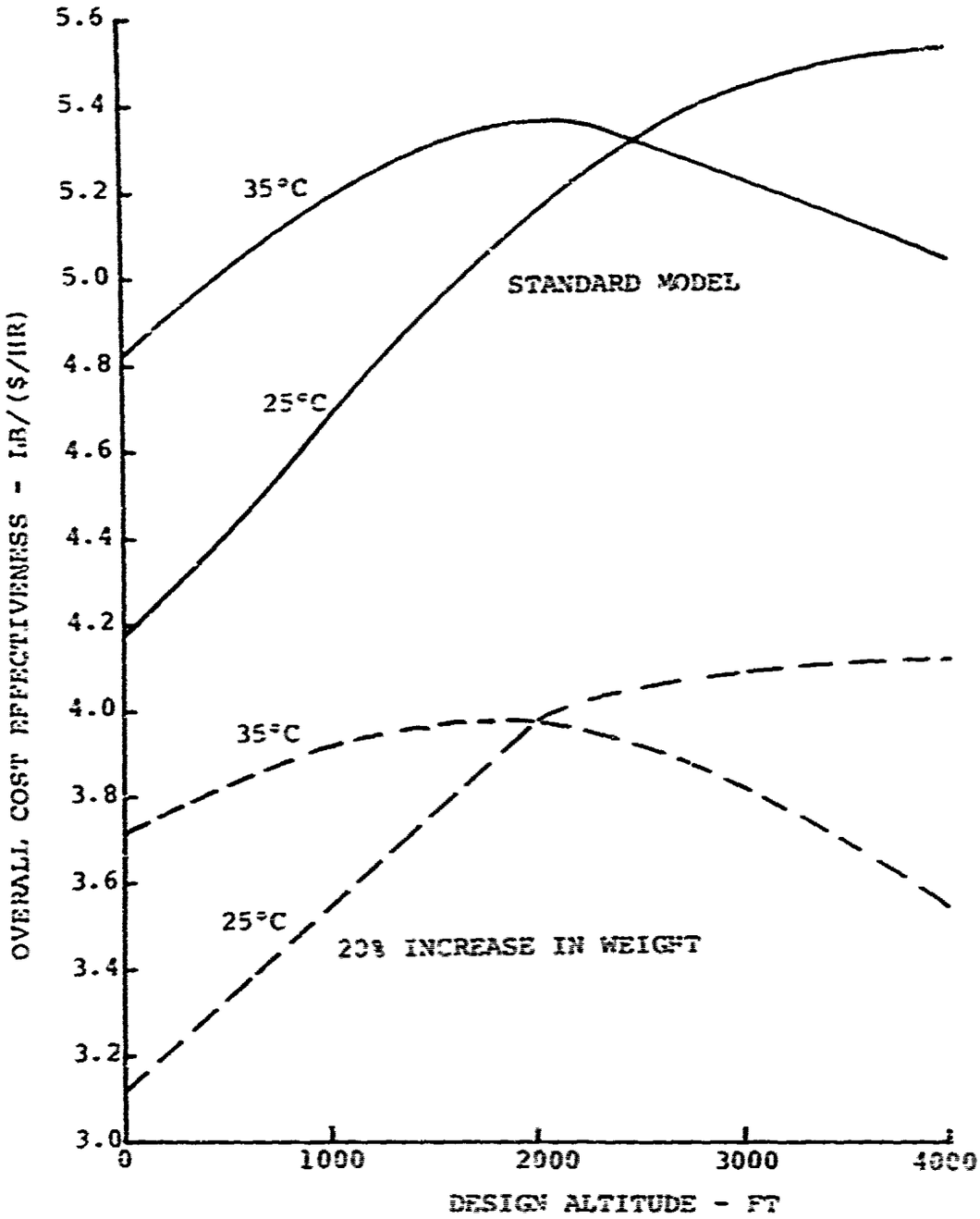


Figure 28. Effect of 20% Increase in Empty Weight Model on Overall Cost Effectiveness of Utility Mission.

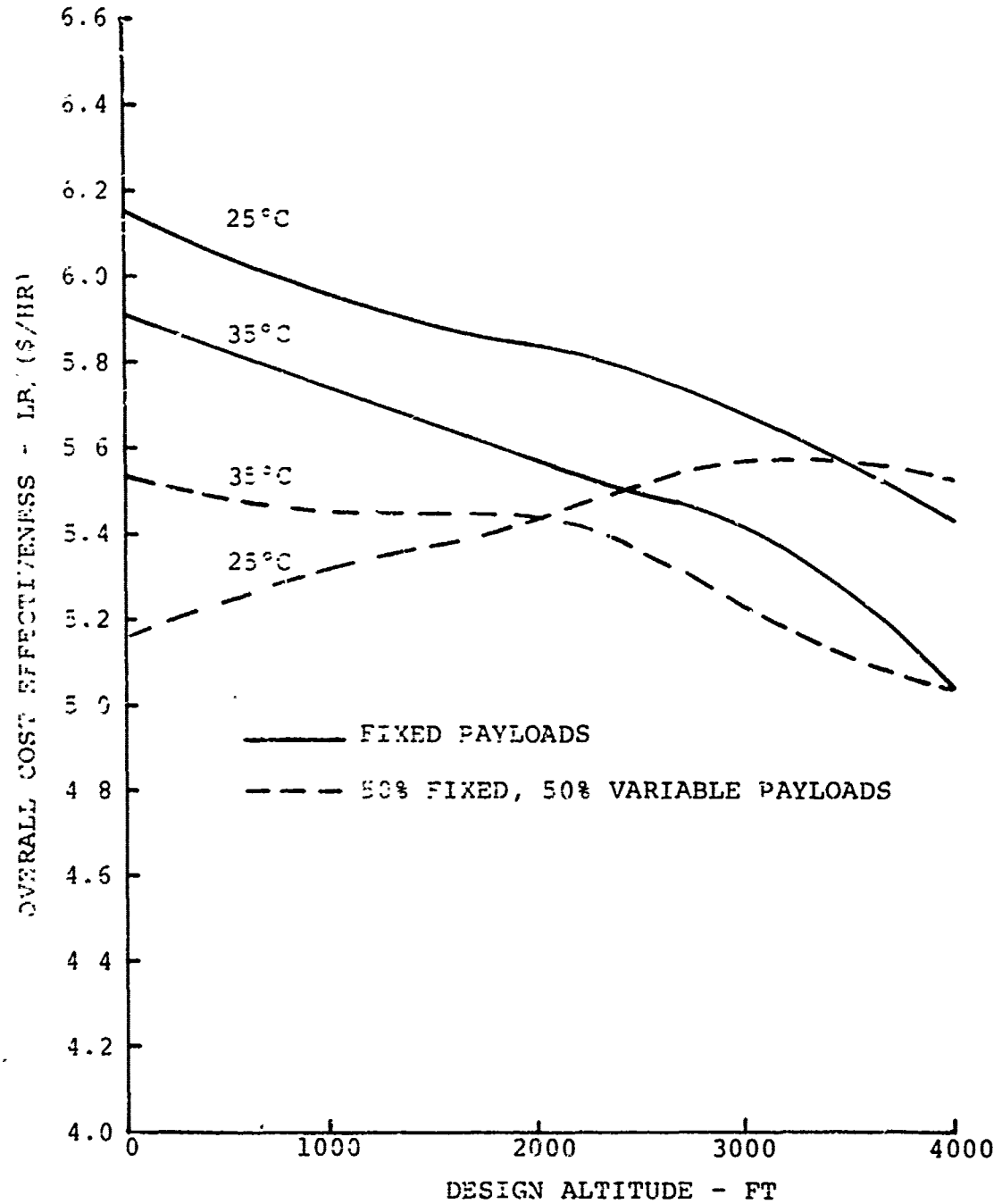


Figure 29. Overall Cost Effectiveness for Fixed and Variable Payload for Utility Mission.

GUNSHIP

Sample computations were carried out for the Gunship Mission. The input is the same as for the Utility Mission except as follows:

DL (Disc Loading) = 9 lb/ft^2
BL (Blade Loading) = 90 lb/ft^2
TAF = 15 (See Table III)
TPY = 14 (See Table III)

The gross weights and the overall cost effectiveness results for 35°C are shown in Figures 30 and 31.

CRANE

Computations for the Crane Mission were performed for the mission as described in Tables I and II. The input is the same as for the utility ship except for the following:

NEN (no. of engines) = 4
DL (disc loading) = 9 lb/ft^2
TS (tip speed) = 750 ft/sec
NMR (no. of main rotor blades) = 6
BL (blade loading) = 90 lb/ft^2
ITR = 0 (See Table III)
KLG = .0405 (See Table III)
KNAC = 2.26 (See Table III)
TAF = 10 (See Table III)
TPU = 1 (See Table III)
TPY = 25 (See Table III)

The gross weights and payloads and the overall cost effectiveness are shown on Figures 32 and 33.

TRANSPORT

The results of a sample run of the transport model are presented in Figure 34. The mission for the transport is described in Tables I and II. The transport model differs from the Utility in the following manner:

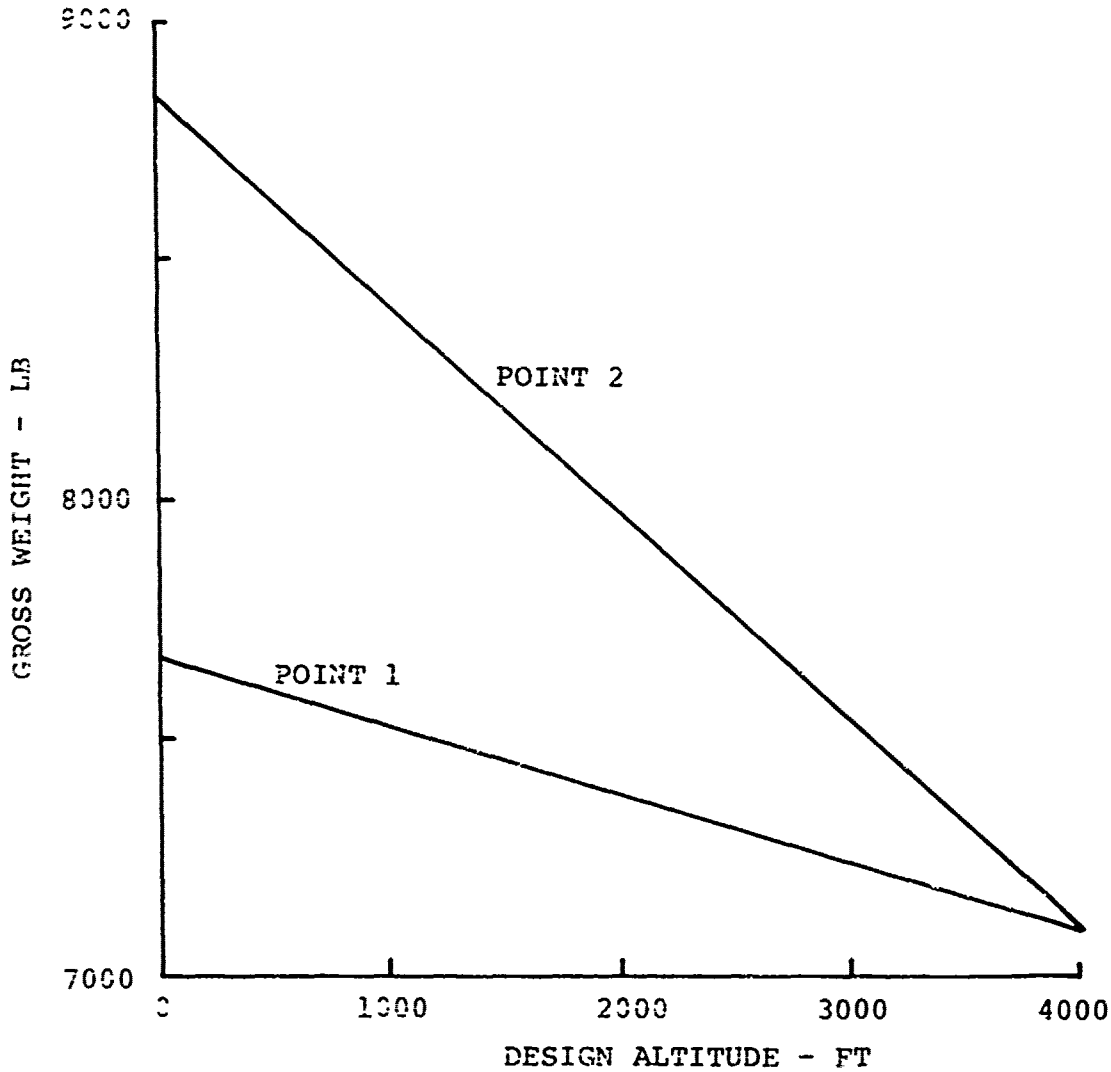


Figure 30. Gross Weights for Gunship Mission for 35°C.

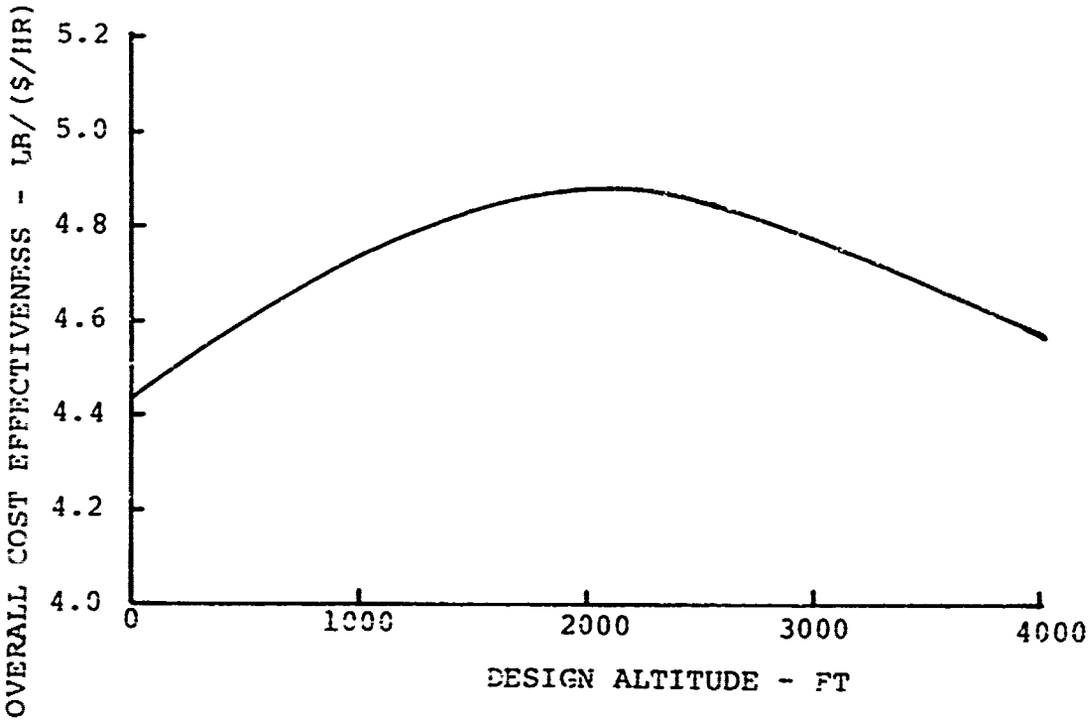


Figure 31 Overall Cost Effectiveness for Gunship Mission for 35°C.

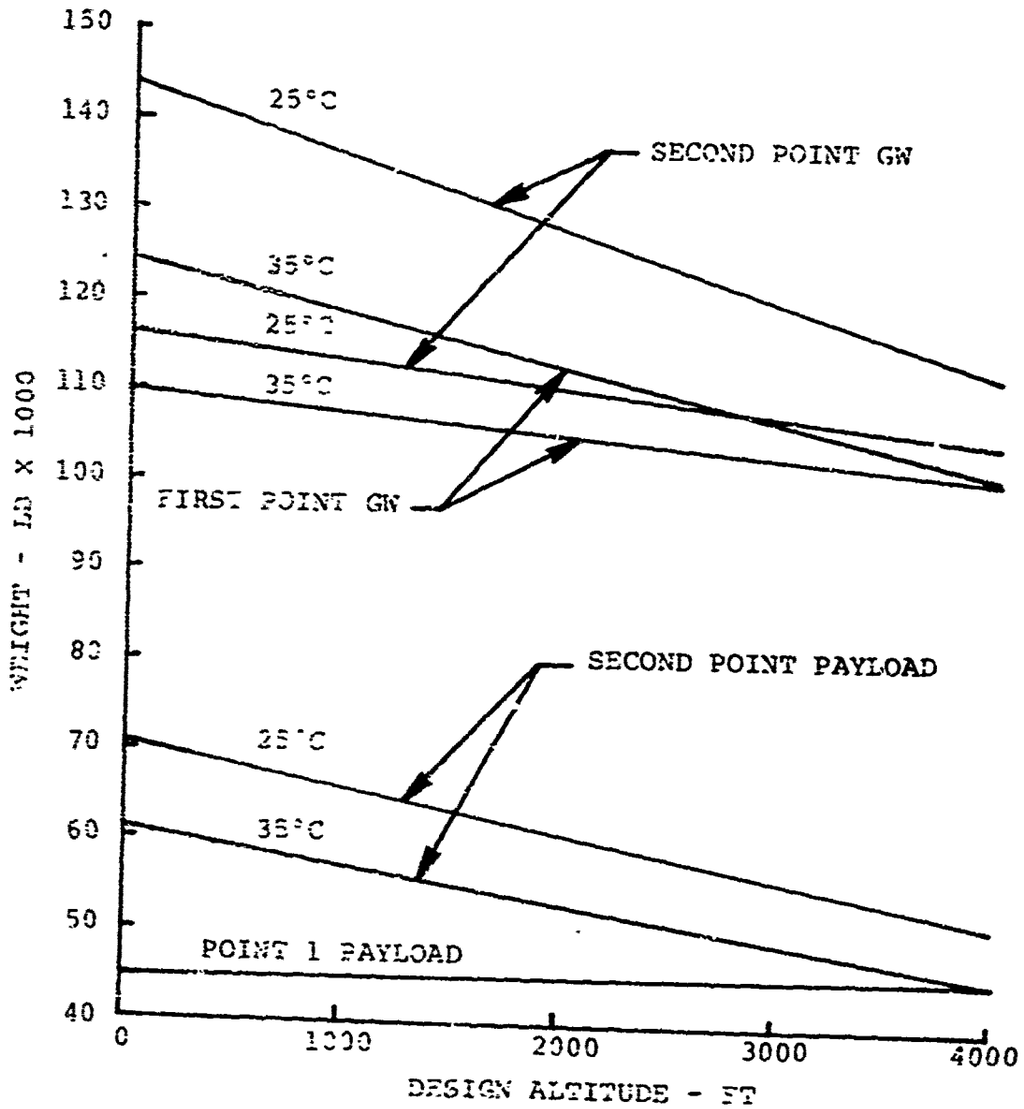


Figure 32 Payloads and Gross Weights for the Crane.

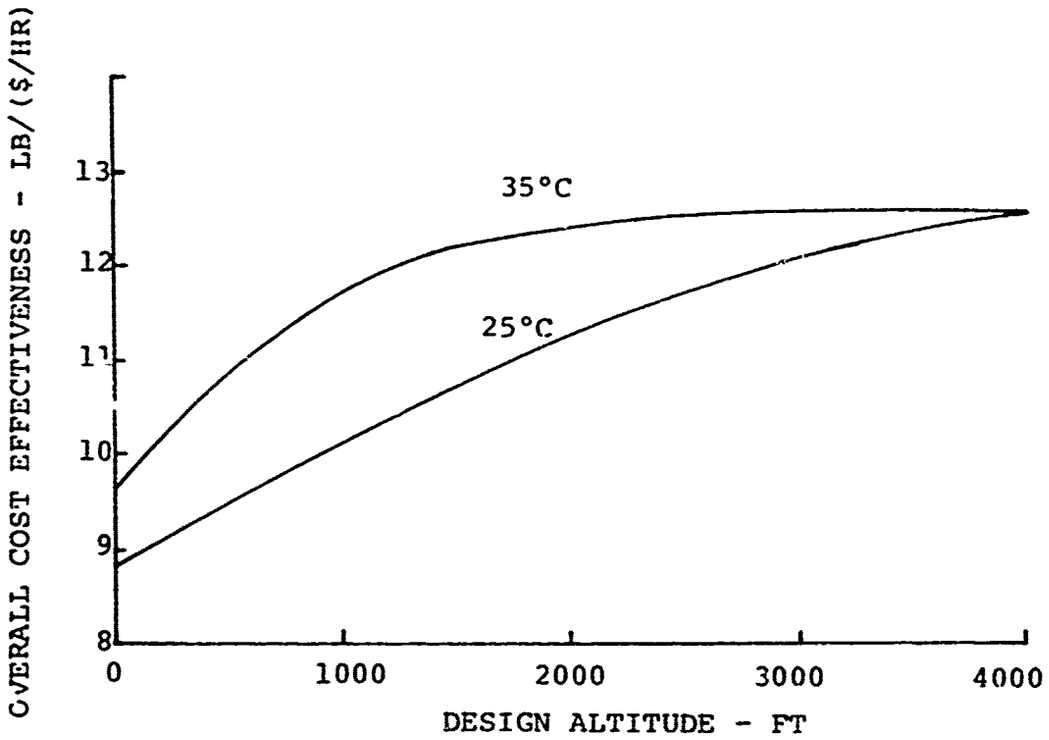


Figure 33. Overall Cost Effectiveness for Crane Mission.

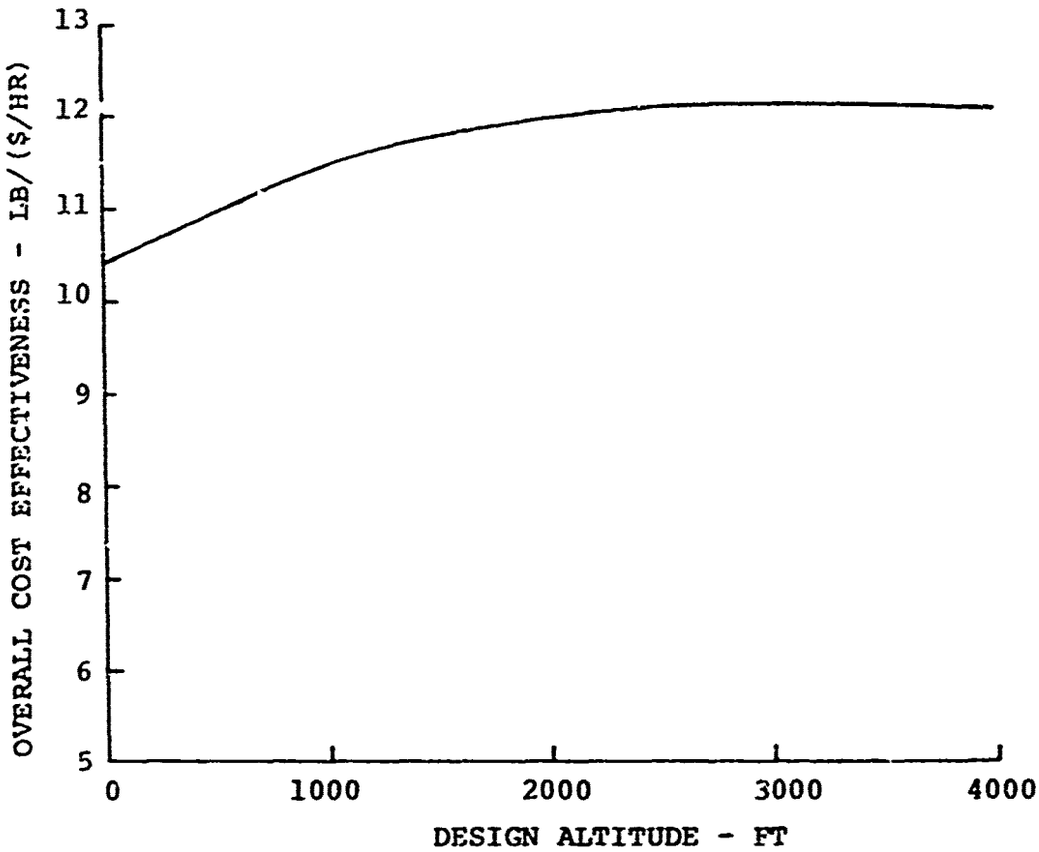


Figure 34. Overall Cost Effectiveness for Transport Model.

DL (disc loading) = 9 lb/ft^2
NMR (no. of blades) = 5
NEN (no. of engines) = 4
TAF = 9 (See Table III)
TPY = 45 (See Table III)
KLG = .0247 (See Table III)
= 2 (See Table III)
AC = 2.26 (See Table III)
PU = 1 (See Table III)

OBSERVATION

The Observation Model differs from Utility in a very basic manner. Instead of increasing the payload at the second design point, the loiter time was increased. The cost effectiveness portion of the model is simplified considerably since there are no probability of hover or payload distribution considerations. For cost effectiveness loiter time is used instead of payload. The mission for the Observation ship is contained in Tables I and II. Other data which differs from the Utility Model is as follows:

DL (disc loading) = 4 lb/ft^2
TS (tip speed) = 650 ft/sec
BF = 0 (See Table III)
ITR = 0 (See Table III)
BRK = 0 (See Table III)
TAF = 10 (See Table III)
TPY = 25 (See Table III)
KLG = .0157 (See Table III)
AG = 0 (See Table III)
KNAL = 1.19 (See Table III)

The cost effectiveness results of a run of the Observation model are shown in Figure 35.

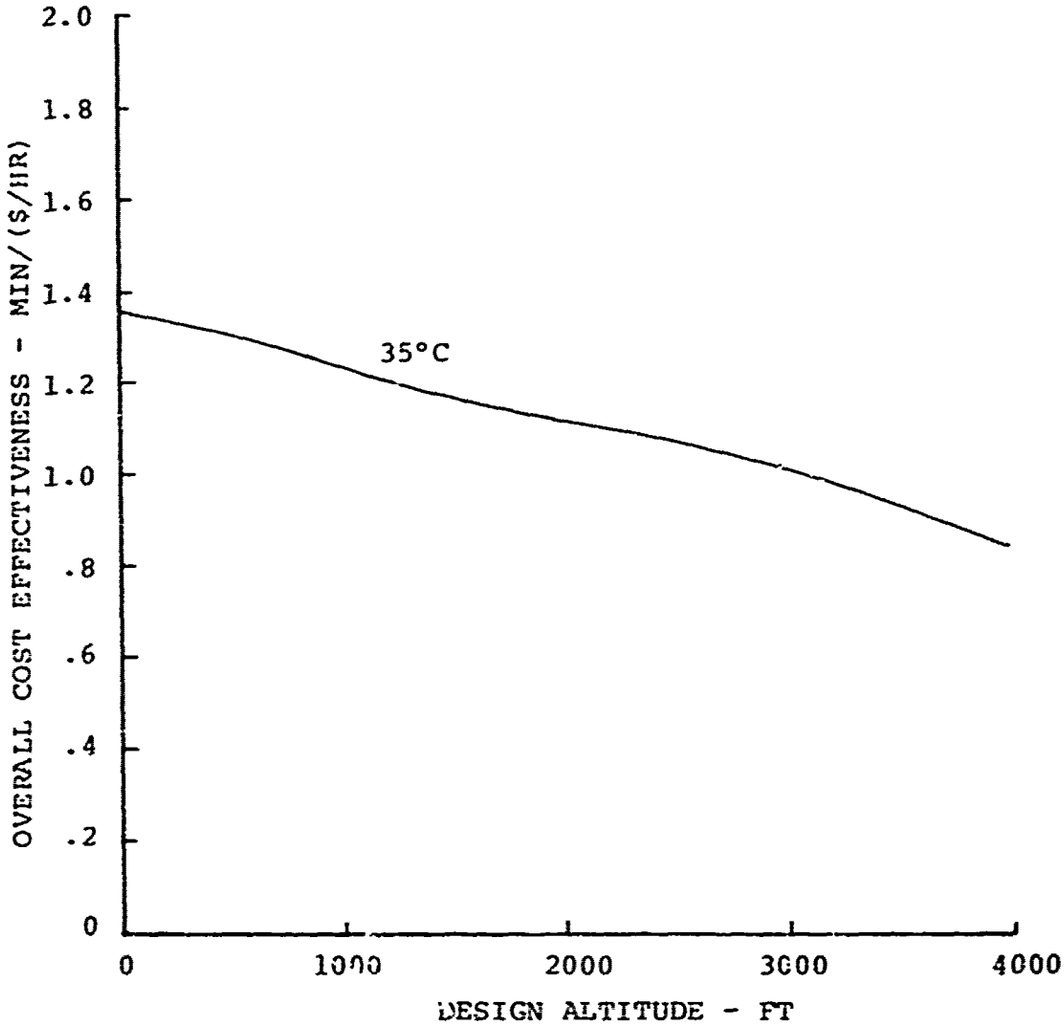


Figure 35. Overall Cost Effectiveness for Observation Model.

CONCLUSIONS

1. An analytical model has been assembled capable of evaluating the effects of a second design point for helicopters.

2. A computer program has been developed which is capable of implementing the solution of these analytical models.

3. It has been shown that optimization points may be obtained that are quite sensitive to several factors. Two of the most important are the assumed payload utilization statistics and the assumed operating environment.

4. The analytical models developed appear to generally give rational results. There are areas, however, where improvements may be made. One of the most worthwhile areas for improvement is the cost model.

5. The objective of this project has been achieved: development of a method which can be used to provide an insight into two-point design criteria and which is capable of selecting criteria which will improve helicopter productivity and cost effectiveness.

LITERATURE CITED

1. Tanner, W. H., CHARTS FOR ESTIMATING ROTARY WING PERFORMANCE IN HOVER AND AT HIGH FORWARD SPEEDS, NASA CR-114, National Aeronautics and Space Administration, Washington, D. C., November 1964.

2. Bossler, R. B., Jr., POWER TRANSFER SYSTEMS FOR FUTURE NAVY HELICOPTERS, Kaman Aerospace Corporation; Report No. R-1032, Naval Air Systems Command, Washington, D. C., June 1972.

3. Bellaire, Robert, and Bousman, William, Lt., A STUDY OF THE ARMY HOT DAY DESIGN HOVER CRITERION, ADS TN 68-1, AVSCOM, St. Louis, Mo., August 1970, AD 717025.

4. LaValiee, R. S., and Sing, C. Y., UH-1D HORSEPOWER REQUIREMENT STUDY, Technical Operations, Incorporated; CDRG-M-185, United States Army Combat Developments Command, Fort Belvoir, Virginia, June 1965.

5. Porterfield, John D. and Maloney, P.F., EVALUATION OF HELICOPTER FLIGHT SPECTRUM DATA, Kaman Aerospace Corporation; USAAVLABS Technical Report 68-68, U. S. Army Aviation Materiel Laboratories, Fort Eustis, Virginia, October 1968, AD 680280.

6. Yates, E. H., COST ANALYSIS AS AN AID TO AIRCRAFT DESIGN, Journal of Aircraft, Vol. 2, No. 2, March-April 1965, pp. 100-107.

7. Yates, E. H., A METHOD FOR ESTIMATING THE PROCUREMENT COST OF AIRCRAFT, General Electric Co., Technical Military Planning Operation, RM60TMP-30, Vol. I, June 1960.

8. Biagioli, M. A., COST ESTIMATING RELATIONSHIPS FOR MILITARY HELICOPTER AIRFRAME PROCUREMENT, U. S. Army Aviation Material Command, St. Louis, Mo., November 1967, AD 673184.

9. U. S. ARMY AVIATION PLANNING MANUAL, Department of the Army Field Manual, FM 101-20, Headquarters, Department of the Army, August 1968.

LITERATURE CITED (Continued)

10. Freudenthal, A. M., and Wang, P. Y., ULTIMATE STRENGTH ANALYSIS OF AIRCRAFT STRUCTURES, Journal of Aircraft, Vol. 7, No. 3., May-June 1970, pp. 205-210.

11. Chenowith, H. B., AN INDICATOR OF RELIABILITY OF ANALYTICAL STRUCTURE DESIGN, Journal of Aircraft, Vol. 7, No. 1, January-February 1970, pp. 13-17.

12. Cook, T. N., RELIABILITY AND MAINTAINABILITY DATA ON FIVE HELICOPTERS, Kaman Aerospace Corporation Internal Memo, 21 March 1972.

13. Berman, A., ZODIAC - COMPUTER PROGRAM FOR PARAMETRIC ANALYSIS, Kaman Aircraft Corporation; RN 70-8, December 1970.

APPENDIX I
ZODIAC II PROGRAM LISTING

```
      IMPLICIT INTEGER(A-U),REAL(V-Z)                                   ZD2   1
      DEFINE FILE 4(1000,102,U,FI4)                                     ZD2   2
      DEFINE FILE 5(101,20,U,FI5)                                       ZD2   3
      DEFINE FILE 6(2),905,U,FI6)                                       ZD2   4
      INTEGER*2 EQL(100,10,4),PRTLST(50)                                ZD2   5
      COMMON /BUG/ DEBUG                                                ZD2   6
      COMMON /MOD/ VARARY(620),NAMARY(400),EQU,PRTLST                   ZD2   7
      COMMON /COMMN/ STPCOM,STTREG,STPREG,COM(200)                      ZD2   8
      COMMON /GEN/ ALBET(50)                                            ZD2   9
      DIMENSION MODNAM(5)                                               ZD2  10
      STPCOM = 200                                                      ZD2  11
      STTREG = 201                                                      ZD2  12
      STPREG = 400                                                      ZD2  13
      CALL INIT                                                         ZD2  14
      CALL MAININ                                                       ZD2  15
      CALL DATAIN                                                       ZD2  16
      CALL MANSET                                                       ZD2  17
      IF (DEBUG .EQ. 1)  CALL PRNTOT(1)                                 ZD2  18
  100 CALL MODIN(N,MODNAM,MODNUM)                                       ZD2  19
      IF(N.EQ.0) GO TO 110                                              ZD2  20
      CALL DATAIN                                                       ZD2  21
      IF (DEBUG .EQ. 1)  CALL PRNTOT(1)                                 ZD2  22
      CALL STOMOD(MODNAM,MODNUM)                                        ZD2  23
      GO TO 100                                                         ZD2  24
  110 CALL MAINRN                                                       ZD2  25
      CALL ERROR(0)                                                     ZD2  26
      CALL EXIT                                                         ZD2  27
      END                                                               ZD2  28
```

```
      SUBROUTINE INIT
      IMPLICIT INTEGER(A-U),REAL(V-Z)
      INTEGER*2 EQU(100,10,4),PRTLST(50)
      COMMON /GEN/ ALBET(50)
      COMMON /MOD/ VARARY(620),NAMARY(400),EQU,PRTLST
      COMMON /COMMN/ STPCOM,STTREG,STPREG,COM(200)
      COMMON /TAB/ TABL(20,2)
      DIMENSION LBET(50)
      DATA  LBET /'=        ,        -   )   *   $   £   +   (   .        9
     A   8   7   6   5   4   3   2   1   0   Z   Y   X   W   V   U   T
     B S   R   Q   P   O   N   M   L   K   J   I   H   G   F   E   D   C
     C   B   A        '/
      DO 100 I = 1,620
  100 VARARY(I) = 0
      LBET(2) = 2101362752
      LBET(4) = 1631600704
      DO 110 I = 1,50
  110 ALBET(I) = LBET(I)
      CALL ITCHK(0,-7)
      DO 120 I = 1,100
      DO 120 J = 1,10
      DO 120 K = 1,4
  120 EQU(I,J,K) = 0
      EQU(1,1,1) = -1
      DO 130 I = 1,400
  130 NAMARY(I) = 0
      DO 140 I = 1,200
  140 COM(I) = 0
      NONE = 0
      ONE = 1
      WRITE (5*1) NONE,ONE
      DO 150 I = 1,20
      TABL(I,1) = 0
  150 TABL(I,2) = 0
      RETURN
      END
```

```
      SUBROUTINE ERROR(N)                                               ERR   1
      INTEGER*2 NER(100)                                                ERR   2
      CATA I,NER/101*0/                                                 ERR   3
      IF(N .EQ. C)  GO TO 110                                           ERR   4
      WRITE (3,100) N                                                   ERR   5
  100 FORMAT (2(18(' ERROR ')/),' ERROR NUMBER ',I4,/,2(18(' ERROR ')/)) ERR   6
      I = I + 1                                                         ERR   7
      IF (I .LE. 100 ) NER(I) = N                                       ERR   8
      RETURN                                                            ERR   9
  110 WRITE (3,120) (NER(J),J = 1,I)                                    ERR  10
  120 FORMAT ('1 SUMMARY OF ERRORS ',/,4313/4313/14I3)                  ERR  11
      RETURN                                                            ERR  12
      END                                                               ERR  13
```

```
      SUBROUTINE MODIN(NCARD,MODNA,MODNUM)                              EIN   1
      IMPLICIT INTEGER (A-U),REAL(V-Z)                                  EIN   2
      COMMON /GEN/ ALBET(50)                                            EIN   3
      COMMON /ALL/ VLST(50),MODNAM(30,5)                                EIN   4
      INTEGER*2 EQU(100,10,4),PRTLST(50)                                EIN   5
      COMMON /MOD/ VARARY(620),NAMARY(400),EQU,PRTLST                   EIN   6
      COMMON /COMMN/ STPCOM,STTREG,STPREG,COM(200)                      EIN   7
      DIMENSION STAT(80), NAME(4), TEM(80),NVAR(16)                     EIN   8
      DIMENSION ARG(40),MODNA(5)                                        EIN   9
      NCARD = 0                                                         EIN  10
      NOEQ = 100                                                        EIN  11
      DO 100 I = 1,100                                                 1EIN  12
      DO 100 J = 1,10                                                  2EIN  13
      DO 100 K = 1,4                                                   3EIN  14
  100 EQU(I,J,K) = 0                                                   3EIN  15
      EQU(1,1,1) = -1                                                   EIN  16
      EQU(2,1,1) = -1                                                   EIN  17
      EQU(1,1,4) = 2                                                    EIN  18
      DO 110 I = 1,400                                                 1EIN  19
  110 NAMARY(I) = 0                                                    1EIN  20
      DO 120 I = STTREG,600                                            1EIN  21
  120 VARARY(I) = 0                                                    1EIN  22
      DO 130 I = 1,50                                                  1EIN  23
  130 PRTLST(I) = 0                                                    1EIN  24
C                                                                       EIN  25
C     READ STATEMENTS IN                                                EIN  26
C                                                                       EIN  27
      END=0                                                             EIN  28
  140 READ(1,150,END=440)  STAT                                         EIN  29
  150 FORMAT(80A1)                                                      EIN  30
      NCARD = NCARD + 1                                                 EIN  31
      IF (NCARD .EQ. 1) WRITE (3,160)                                   EIN  32
  160 FORMAT ('1    EQUATIONS')                                         EIN  33
      IF (NCARD .EQ. 60) WRITE (3,170)                                  EIN  34
  170 FORMAT ('1    EQUATIONS CONT.')                                   EIN  35
      WRITE(3,180) NCARD,STAT                                           EIN  36
  180 FORMAT(I4,3X,80A1)                                                EIN  37
      DO 190 I = 1,80                                                  1EIN  38
  190 TEM(I) = C                                                       1EIN  39
C                                                                       EIN  40
C     REMOVE BLANKS AND COMMENTS                                        EIN  41
C                                                                       EIN  42
      NCM = 0                                                           EIN  43
      DO 200 K = 1,80                                                  1EIN  44
      IF (STAT(K) .EQ. ALBET(50))  NCM = 1                             1EIN  45
  200 IF (STAT(K) .EQ. ALBET(13) .OR. NCM .EQ. 1)  STAT(K) = 0         1EIN  46
      CALL RMVZER(STAT,L)                                               EIN  47
C     CHECK TYPE OF STATEMENT                                           EIN  48
  210 CALL COMND(STAT,TYP,ARG,ERR)                                      EIN  49
      IF (ERR .NE. 0) GO TO 140                                         EIN  50
      IF (TYP .EQ. 0) GO TO 370                                         EIN  51
      IF (TYP .EQ. 13)  GO TO 320                                       EIN  52
      IF (TYP .EQ. 14)  GO TO 140                                       EIN  53
      GO TO (220,240,250,260,260,140,260,140,260,270,260,290),TYP       EIN  54
C                                                                       EIN  55
```

```
C     ITERATE STATEMENT                                                 EIN  56
C                                                                       EIN  57
  220 NIT = EQU(1,1,4)                                                  EIN  58
      IF (NIT .LE. 5) GO TO 230                                         EIN  59
      CALL ERROR(22)                                                    EIN  60
      GO TO 140                                                         EIN  61
  230 EQU(1,NIT,1) = ARG(1)                                             EIN  62
      EQU(1,NIT,2) = ARG(3)                                             EIN  63
      EQU(1,NIT,3) = ARG(4)                                             EIN  64
      EQU(1,NIT,4) = ARG(5)                                             EIN  65
      EQU(1,1,4) = NIT + 1                                              EIN  66
      GO TO 140                                                         EIN  67
C                                                                       EIN  68
C     INORDER STATEMENT                                                 EIN  69
C                                                                       EIN  70
  240 EQU(1,1,3) = 1                                                    EIN  71
      GO TO 140                                                         EIN  72
C                                                                       EIN  73
C     MODULE NAME STATEMENT                                             EIN  74
C                                                                       EIN  75
  250 CALL MONAME(K,ARG)                                                EIN  76
      EQU(1,1,1) = K                                                    EIN  77
      MODNUM = K                                                        EIN  78
      GO TO 140                                                         EIN  79
C                                                                       EIN  80
C     ILLEGAL COMMAND FOR MODULE                                        EIN  81
C                                                                       EIN  82
  260 CALL ERROR(24)                                                    EIN  83
      GO TO 140                                                         EIN  84
C                                                                       EIN  85
C     PRINT STATEMENT                                                   EIN  86
C                                                                       EIN  87
  270 DO 280 I = 1,40                                                   1EIN  88
  280 PRTLST(I) = ARG(I)                                                1EIN  89
      EQU(1,10,1) = 1                                                   EIN  90
      GO TO 140                                                         EIN  91
C                                                                       EIN  92
C     IF - EXPRESSION STATEMENT                                         EIN  93
C                                                                       EIN  94
  290 DO 300 I  = 2,NOEQ                                                1EIN  95
      NIF = I                                                           1EIN  96
      IF (EQU(I,1,1) .EQ. 0) GO TO 310                                  1EIN  97
  300 CONTINUE                                                          1EIN  98
      CALL ERROR(2)                                                     EIN  99
      GO TO 140                                                         EIN 100
  310 EQU(NIF,2,1) = -6                                                 EIN 101
      EQU(NIF,2,2) = ARG(1)                                             EIN 102
      EQU(NIF,2,3) = ARG(3)                                             EIN 103
      EQU(NIF,2,4) = ARG(2)                                             EIN 104
      GO TO 370                                                         EIN 105
C                                                                       EIN 106
C     TABLE LOOK UP                                                     EIN 107
C                                                                       EIN 108
  320 DO 330  I = 1,100                                                 1EIN 109
      EQ = I                                                            1EIN 110
```

```
      IF (EQU(I,1,1) .EQ. 0)  GO TO 340                                 1EIN 111
  330 CONTINUE                                                          1EIN 112
      CALL ERROR (3)                                                     EIN 113
      GO TO 140                                                          EIN 114
  340 DO 350  I = 1,6                                                   1EIN 115
      NVAR(I) = C                                                       1EIN 116
      IF (STAT(I) .EQ. ALBET(1))  GO TO 360                             1EIN 117
  350 NVAR(I) = STAT(I)                                                 1EIN 118
  360 CALL NAM(NVAR,K)                                                   EIN 119
      EQU(EQ,1,1) = -1                                                   EIN 120
      EQU(EQ,1,3) = 3                                                    EIN 121
      EQU(EQ,1,4) = NCARD                                                EIN 122
      EQU(EQ,2,1) = -8                                                   EIN 123
      EQU(EQ,2,2) = ARG(3)                                               EIN 124
      EQU(EQ,2,3) = ARG(4)                                               EIN 125
      IF (EQU(EQ,2,3) .EQ. 0)  EQU(EQ,2,3) = 401                         EIN 126
      EQU(EQ,2,4) = K                                                    EIN 127
      NVAR(1) = ALBET(23-ARG(2))                                         EIN 128
      NVAR(2) = C                                                        EIN 129
      CALL NAM(NVAR,K)                                                   EIN 130
      EQU(EQ,3,1) = -8                                                   EIN 131
      EQU(EQ,3,2) = ARG(5)                                               EIN 132
      IF (ARG(5) .EQ. 0)  EQU(EQ,3,2) = 401                              EIN 133
      EQU(EQ,3,3) = K                                                    EIN 134
      EQU(EQ,3,4) = ARG(1)                                               EIN 135
      GO TO 140                                                          EIN 136
C                                                                        EIN 137
C     SEARCH OUT SYMBOLS AND NAMES                                       EIN 138
C     SYMBOLS ARE REPLACED BY THE FOLLOWING NUMBERS                      EIN 139
C     +=-1,-=-2,*=-3,/=-4,**=-5,(=-6,)=-7,==-8                           EIN 140
C                                                                        EIN 141
  370 NP = 1                                                             EIN 142
      I = C                                                              EIN 143
  380 I = I + 1                                                          EIN 144
  390 IF( I .GT. L ) GO TO 430                                           EIN 145
      IF(STAT(I) .EQ. ALBET(8)) GO TO 400                                EIN 146
      IF ( STAT(I) .LT. 0 .OR. STAT(I) .EQ. ALBET (12)) GO TO 400        EIN 147
      NARG = STAT(I)                                                     EIN 148
      CALL SYMBOL(NARG)                                                  EIN 149
      IF (NARG .EQ. -3 .AND. STAT(I+1) .EQ. ALBET(7)) NARG =-5           EIN 150
      IF (NARG .EQ. -5 .AND. STAT(I+2) .EQ. ALBET(5))  NARG = -9         EIN 151
      TEM(NP) = NARG                                                     EIN 152
      NP = NP+1                                                          EIN 153
      IF (NARG.EQ. -5) I=I+1                                             EIN 154
      IF (NARG .EQ. -9)  I = I + 2                                       EIN 155
      GO TO 380                                                          EIN 156
  400 DO 410 K = 1,16                                                   1EIN 157
  410 NVAR(K) = 0                                                       1EIN 158
      K = C                                                              EIN 159
      NMB = 0                                                            EIN 160
      IF (STAT(I) .GT. ALBET(24) .AND. STAT(I) .LT. ALBET(11)) NMB = 1   EIN 161
  420 K = K + 1                                                          EIN 162
      NVAR(K)=STAT(I)                                                    EIN 163
      I=I+1                                                              EIN 164
      IF(STAT(I) .EQ. ALBET(8)) GO TO 420                                EIN 165
```

```
      IF (STAT(I). LT. 0 .OR. STAT(I) .EQ.  ALBET(12)) GO TO 420        EIN 166
      IF (NVAR(K) .EQ. ALBET(45) .AND. NNB .EQ. 1) GO TO 420            EIN 167
      CALL NAM(NVAR,K)                                                  EIN 168
      TEM (NP) =  K                                                     EIN 169
      NP= NP + 1                                                        EIN 170
      GO TO 390                                                         EIN 171
  430 CONTINUE                                                          EIN 172
      CALL SETUP(TEM,NCARD)                                             EIN 173
C                                                                       EIN 174
C     PREPARE FOR NEXT STATEMENT OR RETURN                              EIN 175
      GO TO 140                                                         EIN 176
  440 RETURN                                                            EIN 177
      END                                                               EIN 178
```

```
      SUBROUTINE SYMBOL (N)                                             SMB   1
      INTEGER SYM(8)                                                    SMB   2
      DATA  SYM /'+   -   *   /       (   )   =   '/                   SMB   3
C                                                                       SMB   4
C                                                                       SMB   5
      DO 100 I =1,8                                                    1SMB   6
      IF (N .NE. SYM(I)) GO TO 100                                     1SMB   7
      N= -I                                                            1SMB   8
      RETURN                                                           1SMB   9
  100 CONTINUE                                                         1SMB  10
      END                                                               SMB  11
```

```
      SUBROUTINE NAM(N,K)                                               NAM   1
      IMPLICIT INTEGER(A-U),REAL(V-Z)                                   NAM   2
      INTEGER*2 EQU(100,10,4),PRTLST(50)                                NAM   3
      COMMON /MOD/ VARARY(626),NAMARY(400),EQU,PRTLST                   NAM   4
      COMMON /COMMN/ STPCOM,STTREG,STPREG,COM(200)                      NAM   5
      COMMON /GEN/ ALBET(26)                                            NAM   6
      DIMENSION N(16)                                                   NAM   7
C                                                                       NAM   8
C     CHECK TO SEE IF N IS A NAME OR A CONSTANT.                        NAM   9
C                                                                       NAM  10
      IF(N(1).GE. ALBET(23) .AND. N(1) .NE. ALBET(8)) GO TO 180         NAM  11
C                                                                       NAM  12
C     N IS A VARIABLE NAME                                              NAM  13
C                                                                       NAM  14
      CALL NAMCMP(N,NAME,NC)                                            NAM  15
      IF (NC .EQ. 0)  GO TO 120                                         NAM  16
      DO 100 I = 1,STPCOM                                              INAM  17
      L = I                                                            INAM  18
      IF (COM(I) .EQ. NAME .OR. COM(I) .EQ. 0)  GO TO 110              INAM  19
  100 CONTINUE                                                         INAM  20
      CALL ERROR(1)                                                     NAM  21
  110 IF (COM(L) .EQ. 0)   VARARY(L) = 123.459E-15                      NAM  22
      COM(L) = NAME                                                     NAM  23
      NAMARY(L) = NAME                                                  NAM  24
      GO TO 170                                                         NAM  25
  120 DO 130 I = 1,STPCOM                                              INAM  26
      L = I                                                            INAM  27
      IF (NAMARY(I) .EQ. NAME) GOTO 170                                INAM  28
      IF (COM(I)    .EQ.  0 )  GO TO 140                               INAM  29
  130 CONTINUE                                                         INAM  30
  140 DO 150 I = STTREG,400                                            INAM  31
      L = I                                                            INAM  32
      IF (NAMARY(I) .EQ. NAME)  GO TO 170                              INAM  33
      IF (NAMARY(I) .EQ. 0)  GO TO 160                                 INAM  34
  150 CONTINUE                                                         INAM  35
      CALL ERROR (2)                                                    NAM  36
  160 NAMARY(L) = NAME                                                  NAM  37
      VARARY(L) = 123.459E-15                                           NAM  38
  170 K = L                                                             NAM  39
      RETURN                                                            NAM  40
C                                                                       NAM  41
C     N IS A CONSTANT.                                                  NAM  42
C                                                                       NAM  43
  180 CALL NUMCMP(N,V)                                                  NAM  44
      L = 401                                                           NAM  45
      IF ( V .EQ. 0) GO TO 200                                          NAM  46
      DO 190 I = 402,600                                               INAM  47
      L = I                                                            INAM  48
      IF (VARARY(I) .EQ. V) GO TO 210                                  INAM  49
      IF (VARARY(I) .EQ. 0) GO TO 200                                  INAM  50
  190 CONTINUE                                                         INAM  51
      CALL ERROR(2)                                                     NAM  52
  200 VARARY(L) = V                                                     NAM  53
  210 K = L                                                             NAM  54
      RETURN                                                            NAM  55
      END                                                               NAM  56
```

```
      SUBROUTINE SETUP(S,NCARD)                                         SET   1
      IMPLICIT INTEGER (A-U),REAL(V-Z)                                  SET   2
      INTEGER*2 EQU(100,10,4),PRTLST(50)                                SET   3
      INTEGER*2 TEM(40,4)                                               SET   4
      COMMON /MOD/ VARARY(620),NAMARY(400),EQU,PRTLST                   SET   5
      COMMON /GEN/ ALBET(50)                                            SET   6
      DIMENSION S(80),TEM1(80),TEM2(80)                                 SET   7
  100 NS = 1                                                            SET   8
      TVAR = 603                                                        SET   9
      DO 110 I= 1,80                                                   1SET  10
  110 TEM2(I) = 0                                                      1SET  11
      DO 120 I = 1,40                                                  1SET  12
      DO 120 J = 1,4                                                   2SET  13
  120 TEM(I,J) = 0                                                     2SET  14
C                                                                       SET  15
C     FIND LOCATION FOR NEXT EQUATION                                   SET  16
C                                                                       SET  17
      DO 130 I = 1,100                                                 1SET  18
      NE = I                                                           1SET  19
      IF(EQU(I,1,1) .EQ.0) GO TO 140                                   1SET  20
  130 CONTINUE                                                         1SET  21
      CALL ERROR(3)                                                     SET  22
C                                                                       SET  23
C     FIND INNERMOST SET OF PARENTHESES                                 SET  24
C                                                                       SET  25
  140 NPAR = 0                                                          SET  26
      HNPAR = 0                                                         SET  27
      DO 150 J= 1,80                                                   1SET  28
      IF (S(J) .EQ. -6) NPAR = NPAR + 1                                1SET  29
      IF (S(J) .EQ. -7) NPAR = NPAR - 1                                1SET  30
      IF (NPAR .LT. 0) CALL ERROR(4)                                   1SET  31
  150 IF (HNPAR .LT. NPAR) HNPAR = NPAR                                1SET  32
      IF (HNPAR .EQ. 0) GO TO 310                                       SET  33
      DO 160 I = 1,80                                                  1SET  34
  160 TEM1(I) = 0                                                      1SET  35
      NPAR = 0                                                          SET  36
      DO 170 J = 1,80                                                  1SET  37
      K = J                                                            1SET  38
      IF (S(J) .EQ. -6) NPAR = NPAR + 1                                1SET  39
      IF (S(J) .EQ. -7) NPAR = NPAR - 1                                1SET  40
      IF (NPAR .EQ. HNPAR) GO TO 180                                   1SET  41
  170 CONTINUE                                                         1SET  42
  180 S(K) = TVAR                                                       SET  43
      K = K + 1                                                         SET  44
      THISVR = TVAR                                                     SET  45
      TVAR = TVAR + 1                                                   SET  46
      DO 190 J = 1,80                                                  1SET  47
      IF (S(K) .EQ. -7) GO TO 200                                      1SET  48
      TEM1(J) = S(K)                                                   1SET  49
      S(K) = 0                                                         1SET  50
  190 K = K + 1                                                        1SET  51
  200 S(K) = 0                                                          SET  52
      CALL RMVZER(S,N)                                                  SET  53
      K = K + 1                                                         SET  54
  210 DO 220 I= 1,K                                                    1SET  55
```

```
      IF (TEM1(I) .NE. -5 .AND. TEM1(I) .NE. -9)  GO TO 220          1SET  56
      TEM(NS,1) = TEM1(I)                                            1SET  57
      TEM(NS,2) = TEM1(I-1)                                          1SET  58
      TEM(NS,3) = TEM1(I+1)                                          1SET  59
      TEM(NS,4) = TVAR                                               1SET  60
      TEM1(I-1) = TVAR                                               1SET  61
      TEM1(I) = 0                                                    1SET  62
      TEM1(I+1) = 0                                                  1SET  63
      TVAR = TVAR + 1                                                1SET  64
      NS = NS + 1                                                    1SET  65
  220 CONTINUE                                                       1SET  66
      CALL RMVZER(TEM1,K)                                             SET  67
      K = K + 1                                                       SET  68
      LEN = 0                                                         SET  69
      KS = 1                                                          SET  70
      NTERM = 0                                                       SET  71
      IF (TEM1(1) .LT. 0) KS = 2                                      SET  72
      DO 270 I = KS,K                                                1SET  73
      IF (TEM1(I) .LE. 0 .AND. TEM1(I) .GE. -2)  GO TO 230           1SET  74
      LEN = LEN + 1                                                  1SET  75
      TEM2(LEN) = TEM1(I)                                            1SET  76
      IF (LEN .EQ. 1)  ST = I                                        1SET  77
      GO TO 270                                                      1SET  78
  230 NTERM = NTERM + 1                                              1SET  79
      IF (LEN .LE. 1)  GO TO 260                                     1SET  80
      KJ = I-1                                                       1SET  81
      DO 240 JK = ST,KJ                                              2SET  82
  240 TEM1(JK) = 0                                                   2SET  83
      TEM1(I-1) = TVAR                                               1SET  84
      CALL TERM(TEM,TEMZ,NS,LEN,TVAR)                                1SET  85
      DO 250 IZ = 1,80                                               2SET  86
  250 TEM2(IZ) = 0                                                   2SET  87
  260 LEN = 0                                                        1SET  88
  270 CONTINUE                                                       1SET  89
      CALL RMVZER(TEM1,K)                                             SET  90
      CALL GATHER(TEM,TEM1,NTERM,THISVR,NS)                           SET  91
      IF (NPAR .GT. 0) GO TO 140                                      SET  92
  280 EQU(NE,1,1) = -1                                                SET  93
      EQU(NE,1,4) = NCARD                                             SET  94
      K = 1                                                           SET  95
      IF (EQU(NE,2,1) .EQ. -6)  K = K + 1                             SET  96
      NS = NS -1                                                      SET  97
      DO 300 I = 1,NS                                                1SET  98
      K = K + 1                                                      1SET  99
      DO 290 J = 1,4                                                 2SET 100
  290 EQU(NE,K,J) = TEM(I,J)                                         2SET 101
      IF (I .EQ. NS) GO TO 300                                       1SET 102
      IF (K .LT. 10) GO TO 300                                       1SET 103
      K = 1                                                          1SET 104
      EQU(NE,1,3) = 10                                               1SET 105
      NE = NE + 1                                                    1SET 106
      EQU(NE,1,1) = -2                                               1SET 107
      EQU(NE,1,4) = NCARD                                            1SET 108
  300 CONTINUE                                                       1SET 109
      EQU(NE,1,3) = K                                                 SET 110
```

```
      RETURN                                                            SET 111
  310 IF (S(2) .NE. -8) CALL ERROR(3)                                   SET 212
      THISVR = S(1)                                                     SET 113
      DO 320 I = 3,80                                                  1SET 114
  320 TEM1(I-2) = S(I)                                                 1SET 115
      TEM1(79) = 0                                                      SET 116
      TEM1(80) = 0                                                      SET 117
      CALL RMVZER(TEM1,K)                                               SET 118
      GO TO 210                                                         SET 119
      END                                                               SET 120
```

```
      SUBROUTINE RMVZER (N1,N)                                          RMV   1
      DIMENSION N1(80)                                                  RMV   2
      K = 0                                                             RMV   3
      DO 100 I = 1,80                                                  1RMV   4
      IF (N1(I) .EQ. 0)  GO TO 100                                     1RMV   5
      K = K + 1                                                        1RMV   6
      N1(K) = N1(I)                                                    1RMV   7
      IF (K .LT. I )  N1(I) = 0                                        1RMV   8
  100 CONTINUE                                                         1RMV   9
      N = K                                                             RMV  10
      RETURN                                                            RMV  11
      END                                                               RMV  12
```

```
      SUBROUTINE TERM(T,T1,N,L,NV)                                      TRM   1
      INTEGER*2 T(40,4)                                                 TRM   2
      INTEGER T1(80)                                                    TRM   3
      NS = NV                                                           TRM   4
      NV = NV + 1                                                       TRM   5
      T(N,1) = T1(2)                                                    TRM   6
      T(N,2) = T1(1)                                                    TRM   7
      T(N,3) = T1(3)                                                    TRM   8
      T(N,4) = NS                                                       TRM   9
      I = 4                                                             TRM  10
  100 N = N + 1                                                         TRM  11
      IF (I .GT. L) RETURN                                              TRM  12
      T(N,1) = T1(I)                                                    TRM  13
      T(N,2) = NS                                                       TRM  14
      T(N,3) = T1(I+1)                                                  TRM  15
      T(N,4) = NS                                                       TRM  16
      I = I + 2                                                         TRM  17
      GO TO 100                                                         TRM  18
      END                                                               TRM  19
```

```
      SUBROUTINE GATHER (T,T1,NT,NV,N)                                  GTR   1
      INTEGER*2 T(40,4)                                                 GTR   2
      INTEGER T1(80)                                                    GTR   3
      IF (NT .EQ. 1 .AND. N .GT. 1)  GO TO 130                          GTR   4
      I = 1                                                             GTR   5
      T(N,1) = -10                                                      GTR   6
      T(N,2) = 1                                                        GTR   7
      IF (T1(1) .GT. 0) GO TO 100                                       GTR   8
      IF(T1(1) .EQ. -2)  T(N,2) = -1                                    GTR   9
      I = I + 1                                                         GTR  10
  100 T(N,3) = T1(I)                                                    GTR  11
      I = I + 1                                                         GTR  12
      T(N,4) = 601                                                      GTR  13
  110 N = N + 1                                                         GTR  14
      IF (T1(I) .NE. 0) GO TO 120                                       GTR  15
      T(N-1,4) = NV                                                     GTR  16
      RETURN                                                            GTR  17
  120 T(N,1) = T1(I)                                                    GTR  18
      T(N,2) = 601                                                      GTR  19
      T(N,3) = T1(I + 1)                                                GTR  20
      T(N,4) = 601                                                      GTR  21
      I = I + 2                                                         GTR  22
      GO TO 110                                                         GTR  23
  130 T(N-1,4) = NV                                                     GTR  24
      RETURN                                                            GTR  25
      END                                                               GTR  26
```

```
      SUBROUTINE CATAIN                                                 DIN   1
      IMPLICIT INTEGER (A-U), REAL(V-Z)                                 DIN   2
      INTEGER*2 EQU(100,10,4),PRTLST(50)                                DIN   3
      COMMON /MOD/ VARARY(620),NAMARY(400),EQU,PRTLST                   DIN   4
      COMMON /GEN/ ALBET(50)                                            DIN   5
      DIMENSION CARD(80),NAM(16),NUM(16)                                DIN   6
      NCDS = 0                                                          DIN   7
C                                                                       DIN   8
C     READ DATA CARD                                                    DIN   9
C                                                                       DIN  10
  100 READ(1,110,END=240)CARD                                           DIN  11
  110 FORMAT (80A1)                                                     DIN  12
      NCDS = NCDS + 1                                                   DIN  13
      IF (NCDS .EQ. 1) WRITE (3,120)                                    DIN  14
  120 FORMAT('1   DATA')                                                DIN  15
C                                                                       DIN  16
C     REMOVE COMMENTS AND BLANKS                                        DIN  17
C                                                                       DIN  18
      DO 130 I =1,80                                                    1DIN 19
      IF (CARD(I) .EQ. ALBET (50)) GO TO 140                            1DIN 20
      IF (CARD(I) .EQ. ALBET (13)) CARD(I) = 0                          1DIN 21
  130 NS = I+1                                                          1DIN 22
      GO TO 160                                                         DIN  23
  140 DO 150 I = NS,80                                                  1DIN 24
  150 CARD(I) = 0                                                       1DIN 25
  160 CALL RMVZER(CARD,NS)                                              DIN  26
C                                                                       DIN  27
C     CHECK FOR TABLE                                                   DIN  28
C                                                                       DIN  29
      IF (CARD(1).NE.ALBET(30).OR.CARD(2).NE.ALBET(49)) GO TO 170       DIN  30
      IF (CARD(3).NE.ALBET(48).OR.CARD(4).NE.ALBET(38)) GO TO 170       DIN  31
      IF (CARD(5).NE.ALBET(45)) GO TO 170                               DIN  32
      CALL TABLIN(CARD)                                                 DIN  33
      GO TO 100                                                         DIN  34
  170 CONTINUE                                                          DIN  35
C                                                                       DIN  36
C     CHECK FOR COMMA IN LAST POSITION                                  DIN  37
C                                                                       DIN  38
      IF (CARD(NS) .EQ. ALBET(3)) NS = NS-1                             DIN  39
      IF (NS .EQ. 0) GO TO 240                                          DIN  40
      IF (NS.LT.80) CARD(NS+1) = 0                                      DIN  41
C                                                                       DIN  42
C     FIND THE NUMBER OF DATA ITEMS                                     DIN  43
C                                                                       DIN  44
      ND = 1                                                            DIN  45
      DO 180 I =1,NS                                                    1DIN 46
  180 IF (CARD(I) .EQ. ALBET(3)) ND = ND + 1                            1DIN 47
C                                                                       DIN  48
C     FIND VARIABLE NAME AND DATA THEN STORE                            DIN  49
C                                                                       DIN  50
      DO 230 JK =1,ND                                                   1DIN 51
      DO 190  I= 1,5                                                    2DIN 52
      NAM(I) = 0                                                        2DIN 53
      K = I + 1                                                         2DIN 54
      IF (CARD(I) .EQ. ALBET(1)) GO TO 200                              2DIN 55
```

```
      NAM(I) = CARD(I)                                                  2DIN  56
  190 CARD(I) = 0                                                       2DIN  57
      NAM(6) = 0                                                        1DIN  58
      IF (CARD(6) .NE. ALBET(1))CALL ERROR (6)                          1CIN  59
  200 KS = K + 15                                                       1DIN  60
      CARD(K-1) = 0                                                     1DIN  61
      J = 0                                                             1DIN  62
      DO 210 I= K,KS                                                    2DIN  63
      J = J + 1                                                         2DIN  64
      NUM(J) = 0                                                        2DIN  65
      KE = I                                                            2CIN  66
      IF (CARD(I) .EQ. 0 .OR. CARD(I) .EQ. ALBET(3)) GO TO 220          2DIN  67
      NUM(J) = CARD(I)                                                  2DIN  68
  210 CARD(I) = 0                                                       2DIN  69
      CALL ERROR(7)                                                     1CIN  70
  220 CARD(KE) = 0                                                      1DIN  71
      CALL NAMCMP(NAM,NAME,NC)                                          1DIN  72
      CALL NUMCMP(NUM,X)                                                1DIN  73
      CALL STORE(NAME,X,NC)                                             1CIN  74
  230 CALL RMVZER(CARD,NS)                                              1DIN  75
      GO TO 100                                                         DIN  76
  240 RETURN                                                            DIN  77
      END                                                               DIN  78
```

```
      SUBROUTINE NUMCMP(NUM,X)                                          NUC   1
      IMPLICIT INTEGER(A-U),REAL(V-Z)                                   NUC   2
      INTEGER*2 EQU(100,10,4),PRTLST(50)                                NUC   3
      COMMON /MOD/ VARARY(620),NAMARY(<00),EQU,PRTLST                   NUC   4
      COMMON /GEN/ ALBET(50)                                            NUC   5
      DIMENSION NUM(16)                                                 NUC   6
C                                                                       NUC   7
C     FIND OUT IF FIRST CHARACTER IS A  -.                              NUC   8
C                                                                       NUC   9
      SGN = 1                                                           NUC  10
      IF (NUM(1) .NE. ALBET(5))  GO TO 100                              NUC  11
      SGN = -1                                                          NUC  12
      NUM(1) = -264224704                                               NUC  13
C                                                                       NUC  14
C     FIND DECIMAL POINT,EXPONENT, AND CONVERT CHARACTERS TO INTEGERS   NUC  15
C                                                                       NUC  16
  100 ND=-1                                                             NUC  17
      NE=0                                                              NUC  18
      DO 120 I = 1,16                                                   INUC  19
      IF (NUM(I) .EQ. 0)  GO TO 130                                     INUC  20
      IF (NUM(I) .GT. 0 .OR. NUM(I) .EQ.ALBET (45))  GO TO 110          INUC  21
      NUM(I)=(NUM(I) + 264224704)/16777216                              INUC  22
      GO TO 120                                                         INUC  23
  110 IF (NUM(I) .EQ. ALBET(12)) ND=I                                   INUC  24
      IF (NUM(I) .EQ. ALBET(45)) NE = I                                 INUC  25
  120 N=I                                                               INUC  26
  130 IF (ND .EQ. -1 .AND. NE .EQ. 0)  ND = N + 1                       NUC  27
      IF (ND .EQ. -1 .AND. NE .GT. 0)  ND = NE                          NUC  28
      NP = ND - 1                                                       NUC  29
      IF (NE .EQ. 0)  GO TO 140                                         NUC  30
      IF (NE .EQ. N)  NP = NE-1                                         NUC  31
      NUM(NE) = 0                                                       NUC  32
      IF (NUM(NE+1) .GT.10 )NE = NE+1                                   NUC  33
      NEX = NUM(NE+1)                                                   NUC  34
      IF (N .EQ. NE+2)  NEX = NEX*10 + NUM(NE+2)                        NUC  35
      IF (NUM(NE) .EQ. ALBET(5)) NEX = -NEX                             NUC  36
      NP = NP + NEX                                                     NUC  37
      N = NE -1                                                         NUC  38
  140 X=0                                                               NUC  39
      DO 150 I = 1,N                                                    INUC  40
      IF (I .EQ. ND) GO TO  150                                         INUC  41
      NP = NP-1                                                         INUC  42
      Y = NUM(I)                                                        INUC  43
      XNP = NP                                                          INUC  44
      X = X + Y*(10.0**XNP)                                             INUC  45
      IF (NP .EQ. 0)  X = AINT(X+.1)                                    INUC  46
  150 CONTINUE                                                          INUC  47
      X= X*SGN                                                          NUC  48
      DO 160 I = 1,16                                                   INUC  49
  160 NUM(I) = 0                                                        INUC  50
      RETURN                                                            NUC  51
      END                                                               NUC  52
```

```
      SUBROUTINE NAMCMP(NAM,NAME,NC)                                    NAC   1
      IMPLICIT INTEGER(A-U),REAL(V-Z)                                   NAC   2
      INTEGER*2 EQU(100,10,4),PRTLST(50)                                NAC   3
      COMMON /MOD/ VARARY(620),NAMARY(400),EQU,PRTLST                   NAC   4
      COMMON /GEN/ ALBET(50)                                            NAC   5
      DIMENSION NAM(16)                                                 NAC   6
      P16 = 16**6                                                       NAC   7
C                                                                       NAC   8
C     FIND NAME LENGTH                                                  NAC   9
C                                                                       NAC  10
      DO 100 I=1,5                                                     INAC  11
      IF (NAM(I) .EQ. 0) GO TO 110                                     INAC  12
  100 NN=I                                                             INAC  13
C                                                                       NAC  14
C     CHECK FOR $                                                       NAC  15
C                                                                       NAC  16
  110 NC=0                                                              NAC  17
      IF(NAM(II .EQ. ALBET(8)) NC=I                                     NAC  18
      IF (NN-NC .LE. 4)  GO TO 120                                      NAC  19
      CALL ERROR(33)                                                    NAC  20
      NN = NC + 4                                                       NAC  21
  120 NN = NN - NC                                                      NAC  22
C                                                                       NAC  23
C     PUT BLANKS IN NAME                                                NAC  24
C                                                                       NAC  25
      IF (NN .EQ. 4) NAME =   NAM(1+NC) - 4210752                       NAC  26
      IF (NN .LE. 3) NAME = 1073741824                                  NAC  27
      IF (NN .LE. 2) NAME = NAME + 4194304                              NAC  28
      IF (NN .GT. 1) GO TO 130                                          NAC  29
      NAME = NAME + 16384 + NAM(1+NC)/P16 + 255                         NAC  30
      GO TO 150                                                         NAC  31
C                                                                       NAC  32
C     FILL IN CHARACTERS                                                NAC  33
C                                                                       NAC  34
  130 NS=1+NC                                                           NAC  35
      NT=NN+NC                                                          NAC  36
      IF (NN .EQ. 4) NS=NS+1                                            NAC  37
      DO 140 I= NS, NT                                                 INAC  38
      NAM(I) = NAM(I)/P16 + 255                                        INAC  39
  140 NAME = NAME + NAM(I)*16**(2*(NT-I))                              INAC  40
  150 DO 160 I = 1,16                                                  INAC  41
  160 NAM(I) = 0                                                       INAC  42
      RETURN                                                            NAC  43
      END                                                               NAC  44
```

```
      SUBROUTINE STORE(N,X,NC)                                          STR   1
      IMPLICIT INTEGER(A-U),REAL(V-Z)                                   STR   2
      INTEGER*2 EQU(100,10,4),PRTLST(50)                                STR   3
      COMMON /MOD/ VARARY(620),NAMARY(400),EQU,PRTLST                   STR   4
      COMMON /COMMN/ STPCOM,STTREG,STPREG,COM(200)                      STR   5
C                                                                       STR   6
C     PRINT VARIABLE AND VALUE                                          STR   7
C                                                                       STR   8
      WRITE (3,100) N,X                                                 STR   9
  100 FORMAT (5X,A4,' = ',1PE13.6)                                      STR  10
      IF (NC .EQ. 1) WRITE (3,110)                                      STR  11
  110 FORMAT ('+  $')                                                   STR  12
C                                                                       STR  13
C     CHECK TO SEE IF NAME IS IN COMMON AREA                            STR  14
C                                                                       STR  15
      DO 120 I = 1,STPCOM                                              1STR  16
      K = I                                                            1STR  17
      IF (N .EQ. NAMARY(I))  GO TO 190                                 1STR  18
  120 CONTINUE                                                         1STR  19
  130 IF (NC .NE. 1) GO TO 150                                          STR  20
      WRITE (3,140)                                                     STR  21
  140 FORMAT ('+',28X,'THIS VARIABLE IS NOT IN COMMON. DATA IGNORED.')  STR  22
      RETURN                                                            STR  23
  150 DO 160 I = STTREG,400                                            1STR  24
      K = I                                                            1STR  25
      IF (N .EQ. NAMARY(I)) GO TO 190                                  1STR  26
      IF (NAMARY(I) .EQ. 0) GO TO 170                                  1STR  27
  160 CONTINUE                                                         1STR  28
  170 WRITE (3,180)                                                     STR  29
  180 FORMAT ('+',28X,'THIS VARIABLE IS NOT USED. DATA IGNORED.')       STR  30
      RETURN                                                            STR  31
  190 IF (VARARY(K) .EQ. 123.459E-15)  GO TO 210                        STR  32
      WRITE (3,200) VARARY(K)                                           STR  33
  200 FORMAT ('+',28X,'THIS VARIABLE WAS DEFINED AS ',1PE13.6)          STR  34
  210 VARARY(K) = X                                                     STR  35
      RETURN                                                            STR  36
      END                                                               STR  37
```

```
      SUBROUTINE ORDER(IN)                                              ORD   1
      IMPLICIT INTEGER*2(A-E),LOGICAL*1(F),INTEGER(G-U),REAL(V-Z)       ORD   2
      INTEGER*2 EQU(100,10,4),PRTLST(50)                                ORD   3
      COMMON /MOD/ VARARY(620),NAMARY(400),EQU,PRTLST                   ORD   4
      INTEGER*2 UNDEF(200)                                              ORD   5
      COMMON /COMMN/ STPCOM,STTREG,STPREG,CCM(200)                      ORD   6
      COMMON /GEN/ GBET(50)                                             ORD   7
      DIMENSION F(620)                                                  ORD   8
      DO 100 I = 1,620                                                 10ORD   9
  100 F(I) = .TRUE.                                                    10ORD  10
      IN = EQU(1,1,3)                                                   ORD  11
C                                                                       ORD  12
C     SET UP LOGICAL ARRAY.                                             ORD  13
C                                                                       ORD  14
      DO 110 I = 1,400                                                 10ORD  15
  110 IF (VARARY(I) .EQ. 123.459E-15 .OR. NAMARY(I) .EQ. 0) F(I)=.FALSE.10ORD  16
      DO 120 I = 402,600                                               10ORD  17
  120 IF (VARARY(I) .EQ. 0)  F(I) = .FALSE.                            10ORD  18
      DO 130 I = 601,620                                               10ORD  19
  130 F(I) = .FALSE.                                                   10ORD  20
C                                                                       ORD  21
C     DETERMINE ORDER OF EVALUATION.                                    ORD  22
C                                                                       ORD  23
      DO 140 I = 2,10                                                  10ORD  24
      DO 140 J = 1,3,2                                                 20ORD  25
      IF (EQU(2,I,J) .LE. 0)   GO TO 150                               20ORD  26
  140 F(EQU(2,I,J)) = .TRUE.                                           20ORD  27
  150 ALPOS = 1                                                         ORD  28
      GLPOS = 1                                                         ORD  29
  160 NCHG = 0                                                          ORD  30
      NEED = 0                                                          ORD  31
      DO 260 I = 3,100                                                 10ORD  32
      IF (EQU(I,1,1) .EQ. -2)  GO TO 230                               10ORD  33
      IF (EQU(I,1,1)) 170,270,260                                      10ORD  34
  170 I1 = I                                                           10ORD  35
      I2 = I                                                           10ORD  36
      IF(EQU(I1,2,1) .NE. -8)  GO TO 180                               10ORD  37
      IF (.NOT.(F(EQU(I1,2,2)).AND.F(EQU(I1,2,3)).AND.F(EQU(I1,3,2))))GO10ORD  38
     A TO 145                                                          10ORD  39
      F(EQU(I1,2,4)) = .TRUE.                                          10ORD  40
      GO TO 220                                                        10ORD  41
  180 DO 200 K = 2,10                                                  20ORD  42
      IF(EQU(I1,K,1) .EQ.0) GO TO 210                                  20ORD  43
      IF (.NOT. F(EQU(I1,K,3))) GO TO 230                              20ORD  44
      IF (EQU(I1,K,1) . EQ. -10)  GO TO 190                            20ORD  45
      IF (.NOT. F(EQU(I1,K,2)))  GO TO 230                             20ORD  46
      IF (EQU(I1,K,1) .EQ. -6) GO TO 200                               20ORD  47
  190 F(EQU(I1,K,4))= .TRUE.                                           20ORD  48
  200 CONTINUE                                                         20ORD  49
      K = K + 1                                                        10ORD  50
  210 IF (EQU(I1+1,1,1) .NE. -2)  GO TO 220                            10ORD  51
      I1 = I1 + 1                                                      10ORD  52
      GO TO 180                                                        10ORD  53
  220 EQU(GLPOS,1,2) = I2                                              10ORD  54
      EQU(I2,1,1) = ALPOS                                              10ORD  55
```

```
      ALPOS = I2                                                        10RD  56
      CLPOS = I2                                                        10RD  57
      I2 = I2 + 1                                                       10RD  58
      NCHG = NCHG + 1                                                   10RD  59
      IF (I2 .LE. I1)  GO TO 220                                        10RD  60
      GO TO 240                                                         10RD  61
  230 NEED = NEED + 1                                                   10RD  62
  240 DO 250 L = 601,620                                                20RD  63
  250 F(L) = .FALSE.                                                    20RD  64
  260 CONTINUE                                                          10RD  65
C                                                                        ORD  66
C     CHECK TO SEE IF ALL EQUATIONS HAVE BEEN USED.                      ORD  67
C                                                                        ORD  68
  270 IF (NEED .EQ. 0)  GO TO 290                                        ORD  69
C                                                                        ORD  70
C     CHECK TO SEE IF ANY ADDITIONS WERE MADE ON THE LAST PASS.          ORD  71
C                                                                        ORD  72
      IF (NCHG .GT. 0)  GO TO 160                                        ORD  73
      CALL ERROR(9)                                                      ORD  74
      GO TO 320                                                          ORD  75
  280 RETURN                                                             ORD  76
C                                                                        ORD  77
C     IF INORDER WAS REQUESTED CHECK ORDER.                              ORD  78
C                                                                        ORD  79
  290 IF (IN .EQ.1) RETURN                                               ORD  80
      DO 300 I = 3,100                                                  10RD  81
      IF (EQU(I,1,2) .EQ. 0) RETURN                                     10RD  82
      IF (EQU(I,1,2) .NE. I+1)  GO TO 310                               10RD  83
  300 CONTINUE                                                          10RD  84
      RETURN                                                             ORD  85
  310 CALL ERROR(10)                                                     ORD  86
      RETURN                                                             ORD  87
  320 K = 1                                                              ORD  88
      DO 330 I = 1,STPCOM                                               10RD  89
      IF (NAMARY(I) .EQ. 0 .OR. F(I))  GO TO 330                        10RD  90
      UNDEF(K) = I                                                      10RD  91
      K = K + 1                                                         10RD  92
  330 CONTINUE                                                          10RD  93
      DO 340 I = STTREG,400                                             10RD  94
      IF (NAMARY(I) .EQ. 0)  GO TO 350                                  10RD  95
      IF (F(I))  GO TO 340                                              10RD  96
      UNDEF(K) = I                                                      10RD  97
      K = K + 1                                                         10RD  98
  340 CONTINUE                                                          10RD  99
  350 K = K - 1                                                          ORD 100
      WRITE (3,360) (NAMARY(UNDEF(I)),I = 1,K)                           ORD 101
  360 FORMAT ('1  UNDEFINED VARIABLES ',/,10(5X,20(2X,A4),/ ))           ORD 102
      GO TO 280                                                          ORD 103
      END                                                                ORD 104
```

```
      SUBROUTINE RUNMOD                                                 RNE   1
      IMPLICIT INTEGER(A-D),INTEGER*2(E),INTEGER(F-U),REAL(V-Z)         RNE   2
      INTEGER*2 EQU(100,10,4),PRTLST(50)                                RNE   3
      COMMON /MOD/       V(620),NAMARY(400),EQU,PRTLST                  RNE   4
      COMMON /GEN/ ALBET(50)                                            RNE   5
      DIMENSION ITERY(4,5),VIT(4)                                       RNE   6
      DIMENSION VTAB(3)                                                 RNE   7
      COMMON /ALL/ VLST(50),MODNAM(30,5)                                RNE   8
      DO 100 I = 2,10                                                  1RNE   9
      DO 100 J = 1,3,2                                                 2RNE  10
      IF (EQU(2,I,J) .EQ. 0)  GO TO 110                                2RNE  11
  100 V(EQU(2,I,J)) = V(EQU(2,I,J+1))                                  2RNE  12
  110 ITR = 0                                                           RNE  13
      IF (EQU(1,1,4) .LE. 0) GO TO 130                                  RNE  14
      NT = EQU(1,1,4) - 2                                               RNE  15
      ITR = NT                                                          RNE  16
      DO 120 I = 1,NT                                                  1RNE  17
      ITERY(I,1) = 0                                                   1RNE  18
      VIT(I) = 0                                                       1RNE  19
      DO 120 K = 2,5                                                   2RNE  20
  120 ITERY(I,K) = EQU(1,I+1,K-1)                                      2RNE  21
  130 NEXT = EQU(1,1,2)                                                 RNE  22
      DO 320 I = 1,100                                                 1RNE  23
      N = NEXT                                                         1RNE  24
      NEXT = EQU(N,1,2)                                                1RNE  25
      DO 300 J = 2,10                                                  2RNE  26
      OP = -EQU(N,J,1)                                                 2RNE  27
      IF (OP .EQ. 0)  GO TO 310                                        2RNE  28
      N1 = EQU(N,J,2)                                                  2RNE  29
      N2 = EQU(N,J,3)                                                  2RNE  30
      N3 = EQU(N,J,4)                                                  2RNE  31
      GO TO (140,150,160,170,210,250,300,240,230,290),OP               2RNE  32
  140 V(N3) = V(N1) + V(N2)                                            2RNE  33
      GO TO 300                                                        2RNE  34
  150 V(N3) = V(N1) - V(N2)                                            2RNE  35
      GO TO 300                                                        2RNE  36
  160 V(N3) = V(N1) * V(N2)                                            2RNE  37
      GO TO 300                                                        2RNE  38
  170 IF (V(N2) .NE. 0) GO TO 200                                      2RNE  39
      WRITE(3,180) N                                                   2RNE  40
  180 FORMAT(' **WARNING  DIVISION BY ZERO IN EQUATION ',I3)           2RNE  41
      IF(N2 .LE. 200) WRITE(3,190) NAMARY(N2)                          2RNE  42
  190 FORMAT('+',50X,'DIVISOR WAS ',A4)                                2RNE  43
      V(N3) = 1.0E60                                                   2RNE  44
      GO TO 300                                                        2RNE  45
  200 V(N3) = V(N1)/V(N2)                                              2RNE  46
      GO TO 300                                                        2RNE  47
  210 IF (V(N1) .LE. 0) WRITE(3,220) N                                 2RNE  48
  220 FORMAT(' ** WARNING A NEGATIVE NUMBER WAS RAISED TO A POWER IN EQU2RNE  49
     ATION ',I3)                                                       2RNE  50
      V(N3) = ABS(V(N1))**V(N2)                                        2RNE  51
      GO TO 300                                                        2RNE  52
  230 IF (V(N1) .LE. 0) WRITE(3,220) N                                 2RNE  53
      V(N3) = ABS(V(N1))**(-V(N2))                                     2RNE  54
      GO TO 300                                                        2RNE  55
```

```
240 VTAB(1) = V(N1)                                                  2RNE  56
    VTAB(2) = V(N2)                                                  2RNE  57
    VTAB(3) = V(EQU(N,3,2))                                          2RNE  58
    INT1 = V(EQU(N,3,3))                                             2RNE  59
    INT2 = EQU(N,3,4)                                                2RNE  60
    CALL TABLOK(VTAB,INT1,INT2,VRET)                                 2RNE  61
    V(N3) = VRET                                                     2RNE  62
    GO TO 310                                                        2RNE  63
250 GO TO (260,270,280),N3                                           2RNE  64
260 IF(V(N1) .LT. V(N2)) GO TO 300                                   2RNE  65
    GO TO 310                                                        2RNE  66
270 IF(V(N1) .EQ. V(N2)) GO TO 300                                   2RNE  67
    GO TO 310                                                        2RNE  68
280 IF(V(N1) .GT. V(N2)) GO TO 300                                   2RNE  69
    GO TO 310                                                        2RNE  70
290 V(N3) = N1 * V(N2)                                               2RNE  71
300 CONTINUE                                                         2RNE  72
310 IF (NEXT .LE. 0) GO TO 330                                       1RNE  73
320 CONTINUE                                                         1RNE  74
330 IF (ITR .GT. 0) GO TO 350                                         RNE  75
340 IF (EQU(1,10,1) .NE. 0) CALL PRNTOT(0)                            RNE  76
    RETURN                                                            RNE  77
350 DO 390 I = 1,NT                                                  1RNE  78
    K = I                                                            1RNE  79
    ITERY(I,1) = ITERY(I,1) + 1                                      1RNE  80
    IF (ITERY(I,2) .NE. 0) GO TO 360                                 1RNE  81
    IF (ITERY(I,1) .LE. ITERY(I,3)) GO TO 130                        1RNE  82
    GO TO 390                                                        1RNE  83
360 VR = V(ITERY(I,2))                                               1RNE  84
    IF (ITERY(I,1) .GT. 1) GO TO 370                                 1RNE  85
    VIT(I) = VR                                                      1RNE  86
    GO TO 130                                                        1RNE  87
370 VRT = ABS(VIT(I)-VR)                                             1RNE  88
    VIT(I) = VR                                                      1RNE  89
    IF (VRT .LT. V(ITERY(I,5))) GO TO 380                            1RNE  90
    IF (ABS(VRT/VR) .GT. V(ITERY(I,4))/100) GO TO 400                1RNE  91
380 ITERY(I,1) = 0                                                   1RNE  92
390 CONTINUE                                                         1RNE  93
    GO TO 340                                                         RNE  94
400 IF (ITERY(K,1) .LT. ITERY(K,3)) GO TO 130                         RNE  95
    CALL ERROR(25)                                                    RNE  96
    GO TO 340                                                         RNE  97
    END                                                               RNE  98
```

```
      SUBROUTINE PRNTOT(NAR)                                            PRT   1
      IMPLICIT INTEGER(A-U),REAL(V-Z)                                   PRT   2
      INTEGER*2 EQU(100,10,4),PT(50)                                    PRT   3
      COMMON /MOD/ VARARY(620),NAMARY(400),EQU,PT                       PRT   4
      COMMON /COMMN/ STPCOM,STTRES,STPREG,COM(200)                      PRT   5
      IF (NAR .EQ. 0)  GO TO 250                                        PRT   6
      WRITE (3,100)                                                     PRT   7
  100 FORMAT (///,'    EQUATIONS'// )                                   PRT   8
      I = 1                                                             PRT   9
      WRITE(3,120) I,((EQU(I,J,K),K=1,4),J=1,10)                        PRT  10
      DO 110 I = 2,100                                                  1PRT  11
      IF (EQU(I,1,1) .EQ. 0)  GO TO 130                                 1PRT  12
      NST = EQU(I,1,3)                                                  1PRT  13
  110 WRITE(3,120) I,((EQU(I,J,K),K=1,4),J=1,NST)                       1PRT  14
  120 FORMAT (/,I3,1X,4I4,5(4X,4I4)/,4(4X,4I4))                         PRT  15
  130 WRITE (3,140)                                                     PRT  16
  140 FORMAT (/////'    CONSTANTS'//)                                   PRT  17
      DO 150 I = 402,600                                                1PRT  18
      K = I - 1                                                         1PRT  19
      IF (VARARY(I) .EQ. 0) GO TO 160                                   1PRT  20
  150 CONTINUE                                                          1PRT  21
  160 WRITE (3,170) (I,VARARY(I),I=401,K)                               PRT  22
  170 FORMAT (5(5X,I4,' = ',1PE13.6))                                   PRT  23
      WRITE(3,180)                                                      PRT  24
  180 FORMAT(//////'  VARIABLES',//)                                    PRT  25
      STT = 1                                                           PRT  26
      STP = STPCOM                                                      PRT  27
      K = 0                                                             PRT  28
      DO 190 I = 1,STPCOM                                               1PRT  29
      IF (COM(I) .EQ. 0)  GO TO 220                                     1PRT  30
  190 K = I                                                             1PRT  31
      GO TO 220                                                         PRT  32
  200 K = 0                                                             PRT  33
      DO 210 I = STT,STP                                                1PRT  34
      IF (NAMARY(I) .EQ. 0)  GO TO 220                                  1PRT  35
  210 K = I                                                             1PRT  36
  220 IF (K .EQ. 0)  GO TO 240                                          PRT  37
      WRITE (3,230) (I,NAMARY(I),I = STT,K)                             PRT  38
  230 FORMAT (20(10(3X,I3,' = ',A4)/))                                  PRT  39
  240 IF (STT .EQ. STTREG)  RETURN                                      PRT  40
      STT = STTREG                                                      PRT  41
      STP = 400                                                         PRT  42
      GO TO 200                                                         PRT  43
  250 WRITE (3,260)                                                     PRT  44
  260 FORMAT (//)                                                       PRT  45
      DO 270 I = 1,50                                                   1PRT  46
      K = I-1                                                           1PRT  47
      IF (PT(I) .EQ. 0)  GO TO 280                                      1PRT  48
  270 CONTINUE                                                          1PRT  49
  280 WRITE (3,290) (NAMARY(PT(I)),VARARY(PT(I)),I=1,K)                 PRT  50
  290 FORMAT (10(5(5X,A4,' = ',1PE13.6)/))                              PRT  51
      RETURN                                                            PRT  52
      END                                                               PRT  53
```

```
      SUBROUTINE TABLOK(VI,N1,N2,VRET)                                  TAB   1
      IMPLICIT INTEGER(A-U),REAL(V-Z)                                   TAB   2
      DIMENSION VI(3),VT(600),VX(100),VY(100),VZ(100)                   TAB   3
      READ (6*N2) NT,NX,NY,NZ,(VT(I),I=1,NT),(VX(I),I=1,NX),(VY(I),I=1, TAB   4
     A NY),(VZ(I),I=1,NZ)                                               TAB   5
      CALL INTERP(VI,NX,NY,NZ,VT,VX,VY,VZ,VRET,N2)                      TAB   6
      RETURN                                                            TAB   7
      END                                                               TAB   8
```

```
      SUBROUTINE COMMD(S,T,R,ER)
      IMPLICIT INTEGER (A-U),REAL(V-Z)
      INTEGER*2 EQU(100,10,4),PRTLST(50)
      COMMON /MOD/ VARARY(620),NAMARY(400),EQU,PRTLST
      COMMON /GEN/ ALBET(50)
      COMMON /TAB/ TABL(20,2)
      EQUIVALENCE (A(1),ALBET(1))
      DIMENSION S(80),R(40),C(10),NA(16), SAV(80),A(50),CDAT(10)
C     THIS ROUTINE CHECKS FOR COMMAS AND RETURNS THEIR ARGUMENTS
C     ER = 1  IF STATEMENT IS IN ERROR
C     ER = 0  IF STATEMENT IS CORRECT
C     T INDICATES TYPE
C     T = 0  EQUATION
C         1  ITERATE
C         2  INORDER
C         3  MODULE NAME
C         4  RUN MOD
C         5  GET
C         6  GIVE
C         7  POINT
C         8  COMMON
C         9  READ
C        10  PRINT
C        11  IF - GO TO
C        12  IF - EXPRESSION
C        13  TABLE LOOK UP
C
C     R  IS THE ARGUMENT LIST RETURNED
C
      DATA CDAT/'ITERNCOMMODURUNMGE: GIVEPOINCOMMREADPRIN'/
      DATA GTO /'GOTO'/
      DO 100 I = 1,10
  100 C(I) = CDAT(I)
      T = 0
      ER = 1
      DO 110 I = 1,40
      IF (I .LT. 17)  NA(I) = 0
      SAV(I) = 0
      SAV(I+40) = 0
  110 R(I) = 0
C
C     CHECK FOR INITIALIZE
C
      IF (S(1) .NE. A(41) .OR. S(2) .NE. A(36))  GO TO 120
      IF (S(3) .NE. A(41) .OR. S(4) .NE. A(30))  GO TO 120
      IF (S(5) .NE. A(41) .OR. S(6) .NE. A(49))  GO TO 120
      IF (S(7) .NE. A(38) .OR. S(8) .NE. A(41))  GO TO 120
      IF (S(9) .NE. A(24) .OR. S(10) .NE. A(45))  GO TO 120
      GO TO 610
C
C     CHECK FOR TABLE LOOK UP
C
  120 DO 130 I = 1,20
      IF (S(I) .EQ. A(1))  GO TO 140
  130 K = I
```

```
      GO TO 150                                                         CMD  56
  140 K = K + 2                                                         CMD  57
      IF (S(K).NE.A(30).OR.S(K+1).NE.A(49).OR.S(K+2).NE.A(48))GO TO 150 CMD  58
      K = K + 3                                                         CMD  59
      IF (S(K) .EQ. A(32).AND.S(K+1).EQ.A(45)) GO TO 530                CMD  60
C                                                                       CMD  61
C     COMPRESS FIRST FOUR CHARACTERS                                    CMD  62
C                                                                       CMD  63
  150 DO 160 I = 1,4                                                    1CMD 64
  160 NA(I) = S(I)                                                      1CMD 65
      CALL NAMCMP(NA,NAME,NC)                                           CMD  66
      IF (NAME .EQ. GTO)  GO TO 690                                     CMD  67
      DO 170 I = 1,10                                                   1CMD 68
      IF(C(I).EQ.NAME)GO TO (190,210,220,280,290,340,350,360,370,380 ), 1CMD 69
     A1                                                                 1CMD 70
  170 CONTINUE                                                          1CMD 71
      IF(S(1).EQ.A(43).AND.S(2).EQ.A(45).AND.S(3).EQ.A(30)) GOTO 390    CMD  72
      IF(S(1).EQ.A(41).AND.S(2).EQ.A(44))  GO TO 390                    CMD  73
      ER = 0                                                            CMD  74
      DO 180 I = 1,80                                                   1CMD 75
      IF (S(I) .EQ.A(1)) RETURN                                         1CMD 76
  180 CONTINUE                                                          1CMD 77
      ER = 1                                                            CMD  78
      RETURN                                                            CMD  79
C                                                                       CMD  80
C     ITERATE STATEMENT                                                 CMD  81
C     R(1) = ON       DEFAULT = 0                                       CMD  82
C     R(2) = FROM     DEFAULT = 1                                       CMD  83
C     R(3) = TIMES    DEFAULT = 50                                      CMD  84
C     R(4) = PTOL     DEFAULT = 5T                                      CMD  85
C     R(5) = ATOL     DEFAULT = 0                                       CMD  86
C                                                                       CMD  87
  190 IF(S(5).NE.A(49).OR.S(6).NE.A(30).OR.S(7).NE.A(45)) RETURN        CMD  88
      T = 1                                                             CMD  89
      DO 200 I = 1,6                                                    1CMD 90
  200 S(I) = 0                                                          1CMD 91
      S(7) = A(2)                                                       CMD  92
      C(1) = A(3)                                                       CMD  93
      CALL RMVZER(S,K)                                                  CMD  94
C                                                                       CMD  95
C     FIND FROM                                                         CMD  96
C                                                                       CMD  97
      C(2) = A(44)                                                      CMD  98
      C(3) = A(32)                                                      CMD  99
      C(4) = A(35)                                                      CMD 100
      C(5) = A(37)                                                      CMD 101
      CALL COMFND(C,S,5,NM,PS,1)                                        CMD 102
      R(2) = VARARY(NM)                                                 CMD 103
C                                                                       CMD 104
C     FIND TIMES                                                        CMD 105
C                                                                       CMD 106
      C(2) = A(30)                                                      CMD 107
      C(3) = A(41)                                                      CMD 108
      C(4) = A(37)                                                      CMD 109
      C(5) = A(45)                                                      CMD 110
```

```
      C(6) = A(31)                                                      CMD 111
      C(7) = A(1)                                                       CMD 112
      CALL COMFND(C,5,7,NM,PS,50)                                       CMD 113
      R(3) = VARARY(NM)                                                 CMD 114
C                                                                       CMD 115
C     FIND ATOL                                                         CMD 116
C                                                                       CMD 117
      C(2) = A(49)                                                      CMD 118
      C(3) = A(30)                                                      CMD 119
      C(4) = A(35)                                                      CMD 120
      C(5) = A(38)                                                      CMD 121
      C(6) = A(1)                                                       CMD 122
      CALL COMFND(C,5,6,NM,PS,0)                                        CMD 123
      R(5) = NM                                                         CMD 124
C                                                                       CMD 125
C     FIND PTOL                                                         CMD 126
C                                                                       CMD 127
      C(2) = A(34)                                                      CMD 128
      NV = 0                                                            CMD 129
      IF (NP .EQ. 0) NV = 5                                             CMD 130
      CALL COMFND(C,5,6,NM,PS,NV)                                       CMD 131
      R(4) = NM                                                         CMD 132
C                                                                       CMD 133
C     FIND ON                                                           CMD 134
C                                                                       CMD 135
      C(2) = A(35)                                                      CMD 136
      C(3) = A(36)                                                      CMD 137
      CALL COMFND(C,5,3,NM,PS,0)                                        CMD 138
      R(1) = NM                                                         CMD 139
      IF (NM .EQ. 401) R(1) = 0                                         CMD 140
      ER = 0                                                            CMD 141
      RETURN                                                            CMD 142
C                                                                       CMD 143
C     INORDER STATEMENT                                                 CMD 144
C                                                                       CMD 145
  210 IF (S(5).NE.A(46).OR.S(6).NE.A(45).OR.S(7).NE.A(32)) RETURN       CMD 146
      T = 2                                                             CMD 147
      ER = 0                                                            CMD 148
      RETURN                                                            CMD 149
C                                                                       CMD 150
C     MODULENAME STATEMENT                                              CMD 151
C     R(1) - R(5) WILL CONTAIN THE MODULE NAME IN COMPRESSED FORM       CMD 152
C                                                                       CMD 153
  220 IF (S(5) .NE. A(38) .OR. S(7) .NE. A(36)) RETURN                  CMD 154
      T = 3                                                             CMD 155
      DO 230 I = 1,80                                                   1CMD 156
      K = I                                                             1CMD 157
      IF (S(I) .EQ. A(I)) GO TO 240                                     1CMD 158
  230 CONTINUE                                                          1CMD 159
      CALL ERROR(18)                                                    CMD 160
      ER = 1                                                            CMD 161
      RETURN                                                            CMD 162
  240 K = K + 1                                                         CMD 163
      KS = K + 15                                                       CMD 164
      IF (KS .GT. 80) KS = 77                                           CMD 165
```

```
      M = 0                                                             CMD 166
      DO 260 I=K,KS,4                                                   1CMD 167
      N = N + 1                                                         1CMD 168
      KT = I+3                                                          1CMD 169
      IF (KT .GT. 80)  KT = 80                                          1CMD 170
      L = 1                                                             1CMD 171
      DO 250 J = I,KT                                                   2CMD 172
      NA(L) = S(J)                                                      2CMD 173
  250 L = L + 1                                                         2CMD 174
      CALL NAMCMP(NA,NM,NC)                                             1CMD 175
      R(M) = NM                                                         1CMD 176
      IF (I+4 .GT. 80) GO TO 270                                        1CMD 177
      IF (S(I+4) .EQ. C) GO TO 270                                      1CMD 178
  260 CONTINUE                                                          1CMD 179
  270 ER = C                                                            CMD 180
      RETURN                                                            CMD 181
C                                                                       CMD 182
C     RUN MOD STATEMENT                                                 CMD 183
C      R(1) - R(5)  CONTAIN THE NAME OF THE MODULE TO BE RUN            CMD 184
C                                                                       CMD 185
  280 IF (S(5) .NE. A(35) .OR. S(6) .NE. A(46))  RETURN                 CMD 186
      T = 4                                                             CMD 187
      S = C                                                             CMD 188
      K = 6                                                             CMD 189
      GO TO 240                                                         CMD 190
C                                                                       CMD 191
C     GET STATEMENT                                                     CMD 192
C     R(1) - R(40) CONTAIN LOCATION OF ARGUMENT                         CMD 193
C                                                                       CMD 194
  290 IF (S(4) .EQ. A(1) .OR. S(5) .EQ. A(1))  RETURN                   CMD 195
      I = 5                                                             CMD 196
      K = 3                                                             CMD 197
      CM = 0                                                            CMD 198
  300 K = K + 1                                                         CMD 199
      DO 310 I = K,80                                                   1CMD 200
      KL = I-1                                                          1CMD 201
      IF(S(I).EQ. 0) GO TO 320                                          1CMD 202
  310 CONTINUE                                                          1CMD 203
  320 N = CM                                                            CMD 204
      L = C                                                             CMD 205
      DO 330 I = K,KL                                                   1CMD 206
      N = N + 1                                                         1CMD 207
      NA(N) = S(I)                                                      1CMD 208
      IF (S(I) .NE. A(3)) GO TO 330                                     1CMD 209
      NA(N) = 0                                                         1CMD 210
      L = L + 1                                                         1CMD 211
      IF (CM .EQ. 1)  NA(1) = A(7)                                      1CMD 212
      CALL NAM(NA,J)                                                    1CMD 213
      R(L) = J                                                          1CMD 214
      N = CM                                                            1CMD 215
  330 CONTINUE                                                          1CMD 216
      NA(N+1) = 0                                                       CMD 217
      IF (CM .EQ. 1)  NA(1) = A(8)                                      CMD 218
      CALL NAM(NA,J)                                                    CMD 219
      R(L+1) = J                                                        CMD 220
```

```
      ER = 0                                                            CMD 221
      RETURN                                                            CMD 222
C                                                                       CMD 223
C     GIVE STATEMENT                                                    CMD 224
C     R(1) - R(40)  CONTAIN ARGUMENT LIST                               CMD 225
C                                                                       CMD 226
  340 IF (S(5) .EQ.  A(1)) RETURN                                       CMD 227
      T = 6                                                             CMD 228
      K = 4                                                             CMD 229
      CM = 0                                                            CMD 230
      GO TO 300                                                         CMD 231
C                                                                       CMD 232
C     POINT STATEMENT                                                   CMD 233
C     R(1)  CONTAINS NAME LOCATION                                      CMD 234
C                                                                       CMD 235
  350 IF (S(5) .NE. A(30))  RETURN                                      CMD 236
      T = 7                                                             CMD 237
      NA(1) = S(6)                                                      CMD 238
      NA(2) = S(7)                                                      CMD 239
      NA(3) = S(8)                                                      CMD 240
      NA(4) = S(9)                                                      CMD 241
      NA(5) = 0                                                         CMD 242
      CALL NAM(NA,J)                                                    CMD 243
      R(1) = J                                                          CMD 244
      ER = 0                                                            CMD 245
      RETURN                                                            CMD 246
C                                                                       CMD 247
C     COMMON STATEMENT                                                  CMD 248
      NO ARGUMENT RETURNED                                              CMD 249
C                                                                       CMD 250
  360 IF (S(5) .NE. A(35) .OR. S(6) .NE. A(36))  RETURN                 CMD 251
      T = 8                                                             CMD 252
      K = 6                                                             CMD 253
      CM = 1                                                            CMD 254
      GO TO 300                                                         CMD 255
C                                                                       CMD 256
C     READ STATEMENT                                                    CMD 257
C     NO ARGUMENT RETURNED                                              CMD 258
C                                                                       CMD 259
  370 IF (S(5) .EQ. A(1)) RETURN                                        CMD 260
      T = 9                                                             CMD 261
      ER = 0                                                            CMD 262
      RETURN                                                            CMD 263
C                                                                       CMD 264
C     PRINT STATEMENT                                                   CMD 265
C     R(1) - R(40)  CONTAIN PRINT LIST                                  CMD 266
C                                                                       CMD 267
  380 IF (S(5) .NE. A(30)) RETURN                                       CMD 268
      T = 10                                                            CMD 269
      CM = 0                                                            CMD 270
      K = 5                                                             CMD 271
      GO TO 300                                                         CMD 272
C                                                                       CMD 273
C     IF STATEMENT                                                      CMD 274
C     R(1)  CONTAINS FIRST ARGUMENT                                     CMD 275
```

```
C     R(2)  CONTAINS LOGICAL OPERATOR                                  CMD 276
C     R(3) CONTAINS SECOND OPERATOR                                    CMD 277
C     FOR  IF - EXPRESSION STATEMENT 5 CONTAINS EXPRESSION             CMD 278
C     FOR  IF - GO TO STATEMENT R(4)  CONTAINS POINT VAR LOCATION      CMD 279
C                                                                      CMD 280
C     LOOK FOR ,                                                       CMD 281
C                                                                      CMD 282
  390 DO 400 I = 3,80                                                 1CMD 283
      K = I                                                           1CMD 284
      IF (S(I) .EQ. A(3)) GO TO 410                                   1CMD 285
  400 CONTINUE                                                        1CMD 286
      ER = C                                                           CMD 287
      RETURN                                                           CMD 288
C                                                                      CMD 289
C     LOOK FOR = SIGN                                                  CMD 290
C                                                                      CMD 291
  410 DO 420 I = K,80                                                 1CMD 292
      IF (S(I) .EQ. A(1))GO TO 510                                    1CMD 293
  420 CONTINUE                                                        1CMD 294
C                                                                      CMD 295
C     FIND GO TO                                                       CMD 296
C                                                                      CMD 297
      C(1) = A(43)                                                     CMD 298
      C(2) = A(35)                                                     CMD 299
      C(3) = A(30)                                                     CMD 300
      C(4) = A(35)                                                     CMD 301
      CALL COMFND(C,S,4,NM,PS,0)                                       CMD 302
      IF (PS .GT. 0) GO TO 430                                         CMD 303
      CALL ERROR (19)                                                  CMD 304
      ER = 1                                                           CMD 305
      RETURN                                                           CMD 306
  430 R(4) = NM                                                        CMD 307
      T = 11                                                           CMD 308
C                                                                      CMD 309
C     CHECK FOR ISLT   R(2) = 1                                        CMD 310
C                                                                      CMD 311
  440 C(1) = A (41)                                                    CMD 312
      C(2) = A(31)                                                     CMD 313
      C(3) = A(38)                                                     CMD 314
      C(4) = A(30)                                                     CMD 315
      CALL COMFND(C,S,4,NM,PS,0)                                       CMD 316
      IF (PS .EQ. 0)  GO TO 450                                        CMD 317
      R(3) = NM                                                        CMD 318
      R(2) = 1                                                         CMD 319
      GO TO 480                                                        CMD 320
C                                                                      CMD 321
C     CHECK FOR ISEQ    R(2) = 2                                       CMD 322
C                                                                      CMD 323
  450 C(3) = A(45)                                                     CMD 324
      C(4) = A(33)                                                     CMD 325
      CALL COMFND (C,S,4,NM,PS,0)                                      CMD 326
      IF (PS .EQ. 0) GO TO 465                                         CMD 327
      R(3) = NM                                                        CMD 328
      R(2) = 2                                                         CMD 329
      GO TO 480                                                        CMD 330
```

```
C                                                                       CMD 331
C     CHECK FOR ISGT  R(2) = 3                                          CMD 332
C                                                                       CMD 333
  460 C(3) = A(43)                                                      CMD 334
      C(4) = A(30)                                                      CMD 335
      CALL COMFND(C,5,4,NM,PS,0)                                        CMD 336
      IF (PS .EQ. 0) GO TO 470                                          CMD 337
      R(3) = NM                                                         CMD 338
      R(2) = 3                                                          CMD 339
      GO TO 480                                                         CMD 340
  470 CALL ERROR(20)                                                    CMD 341
      ER = 1                                                            CMD 342
      RETURN                                                            CMD 343
C                                                                       CMD 344
C     FIND IF                                                           CMD 345
C                                                                       CMD 346
  480 C(2) =A(44)                                                       CMD 347
      CALL COMFND(C,5,2,NM,PS,0)                                        CMD 348
      IF (PS .EQ. 1) GO TO 490                                          CMD 349
      CALL ERROR(21)                                                    CMD 350
      ER = 1                                                            CMD 351
      RETURN                                                            CMD 352
  490 R(1) = NM                                                         CMD 353
      ER = 0                                                            CMD 354
      IF (T .EQ. 11) RETURN                                             CMD 355
C                                                                       CMD 356
C     PUT EXPRESSION IN S                                               CMD 357
C                                                                       CMD 358
      DO 500 I = 1,80                                                   1CMD 359
  500 S(I) = SAV(I)                                                     1CMD 360
      RETURN                                                            CMD 361
C                                                                       CMD 362
C     SAVE EXPRESSION IN SAV                                            CMD 363
C                                                                       CMD 364
  510 K = K + 1                                                         CMD 365
      L = 1                                                             CMD 366
      DO 520 I = K,80                                                   1CMD 367
      SAV(L) = S(I)                                                     1CMD 368
      S(I) = 0                                                          1CMD 369
  520 L = L + 1                                                         1CMD 370
      S(K-1) = 0                                                        CMD 371
      T = 12                                                            CMD 372
      GO TO 440                                                         CMD 373
  530 T = 13                                                            CMD 374
      K = K + 2                                                         CMD 375
      DO 540 I = K,80                                                   1CMD 376
      IF(S(I) .EQ. A(11)) LP = I                                        1CMD 377
  540 IF (S(I) .EQ.A(6)) RP = I                                         1CMD 378
      S(LP) = 0                                                         CMD 379
      S(RP) = 0                                                         CMD 380
      LP = LP + 1                                                       CMD 381
      RP = RP - 1                                                       CMD 382
      NV = 1                                                            CMD 383
      K1 = 1                                                            CMD 384
      DO 560 I = LP,RP                                                  1CMD 385
```

```
      IF (S(I) .EQ. A(2)) S(I) = 0                                      1CMD 386
      NA(K1) = S(I)                                                     1CMD 387
      IF (S(I) .NE. 0 .AND.I .NE. RP)  GO TO 550                        1CMD 388
      NA(K1+1) = 0                                                      1CMD 389
      CALL NAM(NA,K2)                                                   1CMD 390
      NV = NV + 1                                                       1CMD 391
      R(NV+1) = K2                                                      1CMD 392
      R(2) = NV-1                                                       1CMD 393
      K1 = 0                                                            1CMD 394
  550 K1 = K1 + 1                                                       1CMD 395
  560 S(I) = 0                                                          1CMD 396
      LP = LP - 2                                                        CMD 397
      L = 0                                                              CMD 398
      DO 570 I = K,LP                                                   1CMD 399
      L = L + 1                                                         1CMD 400
  570 NA(L) = S(I)                                                      1CMD 401
      CALL NAMCMP(NA,NAME,NC)                                            CMD 402
      DO 580 I = 1,20                                                   1CMD 403
      K = I                                                             1CMD 404
      IF (TABL(I,1) .EQ. NAME)  GO TO 600                               1CMD 405
      IF (TABL(I,1) .EQ. 0) GO TO 590                                   1CMD 406
  580 CONTINUE                                                          1CMD 407
      CALL ERROR(27)                                                     CMD 408
      RETURN                                                             CMD 409
  590 TABL(K,1) = NAME                                                   CMD 410
      TABL(K,2) = 0                                                      CMD 411
      R(1) = K                                                           CMD 412
      ER = 0                                                             CMD 413
      RETURN                                                             CMD 414
  600 R(1) = K                                                           CMD 415
      ER = 0                                                             CMD 416
      IF (TABL(K,2) .EQ. NV ) RETURN                                     CMD 417
      CALL ERROR(28)                                                     CMD 418
      ER = 1                                                             CMD 419
      RETURN                                                             CMD 420
C                                                                        CMD 421
C     INITIALIZE STATEMENT                                               CMD 422
C                                                                        CMD 423
  610 DO 620 I = 1,10                                                   1CMD 424
  620 S(I) = 0                                                          1CMD 425
      DO 630 I = 2,10                                                   1CMD 426
      DO 630 J = 1,3,2                                                  2CMD 427
      P1 = I                                                            2CMD 428
      P2 = J                                                            2CMD 429
      IF (EQU(2,P1,P2) .EQ. 0)  GO TO 640                               2CMD 430
  630 CONTINUE                                                          2CMD 431
      CALL ERROR(31)                                                     CMD 432
  640 CALL RMVZER(S,N6)                                                  CMD 433
      DO 650 I = 1,16                                                   1CMD 434
      K = I                                                             1CMD 435
      NA(I) = S(I)                                                      1CMD 436
      S(I) = 0                                                          1CMD 437
      IF (NA(I) .EQ. A(1))  GO TO 660                                   1CMD 438
  650 NA(I+1) = 0                                                       1CMD 439
  660 NA(K) = 0                                                          CMD 440
```

```
      CALL NAM(NA,K)                                                    CMD 441
      CALL RMVZER(S,N6)                                                 CMD 442
      EQU(2,P1,P2) = K                                                  CMD 443
      DO 670 I = 1,16                                                  1CMD 444
      K = I                                                            1CMD 445
      NA(I) = S(I)                                                     1CMD 446
      S(I) = 0                                                         1CMD 447
      IF( NA(I) .EQ. A(3) .OR.NA(I) .EQ. 0)  GO TO 680                 1CMD 448
  670 NA(I+1) = 0                                                      1CMD 449
  680 NA(K) = 0                                                         CMD 450
      CALL NAM(NA,K)                                                    CMD 451
      EQU(2,P1,P2+1) = K                                                CMD 452
      P2 = 4-P2                                                         CMD 453
      P1 = P1 + (3-P2)/2                                                CMD 454
      CALL RMVZER(S,N6)                                                 CMD 455
      IF (N6 .GT. 0)  GO TO 640                                         CMD 456
      EQU(2,1,3) = P2                                                   CMD 457
      T = 14                                                            CMD 458
      ER = 0                                                            CMD 459
      RETURN                                                            CMD 460
  690 IF (S(5) .LE. A(24))  GO TO 700                                   CMD 461
      T = 0                                                             CMD 462
      ER = 0                                                            CMD 463
      RETURN                                                            CMD 464
  700 DO 710 I = 1,4                                                   1CMD 465
  710 S(I) = 0                                                         1CMD 466
      CALL RMVZER(S,K)                                                  CMD 467
      DO 720 I = 1,K                                                   1CMD 468
  720 NA(I) = S(I)                                                     1CMD 469
      NA(K+1) = 0                                                       CMD 470
      CALL NAM(NA,K)                                                    CMD 471
      T = 15                                                            CMD 472
      R(1) = K                                                          CMD 473
      ER = 0                                                            CMD 474
      RETURN                                                            CMD 475
      END                                                               CMD 476
```

```
      SUBROUTINE COMFND(C,S,N,NM,PS,R)                                  CMF   1
      IMPLICIT INTEGER(A-U), REAL(V-Z)                                  CMF   2
      INTEGER*2 EQU(100,10,4),PRTLST(50)                                CMF   3
      COMMON /MOD/ VARARY(620),NAMARY(400),EQU,PRTLST                   CMF   4
      COMMON /GEN/ ALBET(50)                                            CMF   5
      DIMENSION C(16), S(80) , NA(16)                                   CMF   6
      PS = 0                                                            CMF   7
      NM = 401                                                          CMF   8
      ST = 1                                                            CMF   9
  100 DO 110 I = ST,80                                                 1CMF  10
      K = I                                                            1CMF  11
      IF (S(I) .EQ. 0) GO TO 120                                       1CMF  12
      IF (S(I) .EQ. C(1)) GO TO 170                                    1CMF  13
  110 CONTINUE                                                         1CMF  14
  120 IF (R .EQ. 0) RETURN                                              CMF  15
  130 DO 140 I = 402,600                                               1CMF  16
      K = I                                                            1CMF  17
      IF (VARARY(I) .EQ. R) GO TO 160                                  1CMF  18
      IF (VARARY(I) .EQ. 0) GO TO 150                                  1CMF  19
  140 CONTINUE                                                         1CMF  20
      PS = -1                                                           CMF  21
      CALL ERROR(2)                                                     CMF  22
      NM = 0                                                            CMF  23
      RETURN                                                            CMF  24
  150 VARARY(K) = R                                                     CMF  25
  160 NM = K                                                            CMF  26
      RETURN                                                            CMF  27
  170 KS = K + N - 1                                                    CMF  28
      ST = K + 1                                                        CMF  29
      L = 1                                                             CMF  30
      DO 180 I = K,KS                                                  1CMF  31
      IF (S(I) .NE. C(L)) GO TO 100                                    1CMF  32
  180 L = L + 1                                                        1CMF  33
      DO 190 I = K,KS                                                  1CMF  34
  190 S(I) = 0                                                         1CMF  35
      K = KS + 1                                                        CMF  36
      L = 0                                                             CMF  37
      DO 200 I = K,80                                                  1CMF  38
      ST = I                                                           1CMF  39
      IF (S(I) .EQ. 0 .OR. S(I) .EQ. ALBET(3)) GO TO 210               1CMF  40
      L = L + 1                                                        1CMF  41
      NA(L) = S(I)                                                     1CMF  42
  200 S(I) = 0                                                         1CMF  43
  210 NA(L+1) = 0                                                       CMF  44
      CALL NAM(NA,NM)                                                   CMF  45
      PS = 1                                                            CMF  46
      CALL RMVZER(S,K)                                                  CMF  47
      RETURN                                                            CMF  48
      END                                                               CMF  49
```

```
      SUBROUTINE MAININ                                                 MIN   1
      IMPLICIT INTEGER(A-U),REAL(V-Z)                                   MIN   2
      DIMENSION S(80),R(40),NA(16)                                      MIN   3
      LOGICAL CM                                                        MIN   4
      INTEGER*2 MNPG(100,6),PRTLST(100),GETLST(100)                     MIN   5
      COMMON /BUG/ DEBUG                                                MIN   6
      COMMON /MAIN/ VRB(620),NAMS(400),MNPG,PRTLST,GETLST,MAINM(200)    MIN   7
      COMMON /ALL/ VLST(50),MODNAM(30,5)                                MIN   8
      COMMON /GEN/ ALBET(50)                                            MIN   9
      INTEGER*2 EQU(100,10,4)                                           MIN  10
      COMMON /MOD/ VARARY(620),NAMARY(400),EQU                          MIN  11
      DO 100 I = 1,100                                                 1MIN  12
      DO 100 J = 1,6                                                   2MIN  13
  100 MNPG(I,J) = 0                                                    2MIN  14
      EQ = 2                                                            MIN  15
      RC = 0                                                            MIN  16
      NPT = 1                                                           MIN  17
      NGV = 1                                                           MIN  18
      CARD = 0                                                          MIN  19
  110 READ(1,120,END=330) S                                             MIN  20
  120 FORMAT(80A1)                                                      MIN  21
      CARD = CARD + 1                                                   MIN  22
      IF (CARD .EQ. 1 .OR. CARD .EQ. 60) WRITE (3,130)                  MIN  23
  130 FORMAT ('1')                                                      MIN  24
      CMT= 81                                                           MIN  25
      NRT = 0                                                           MIN  26
      NB = 0                                                            MIN  27
      DO 140 I = 1,80                                                  1MIN  28
      K = 81 - I                                                       1MIN  29
      IF(S(K).NE. ALBET(13)) NRT = 1                                   1MIN  30
      IF(S(K).EQ. ALBET(13)  .AND. NRT.EQ.0) NB = K                    1MIN  31
  140 IF(S(K).EQ. ALBET(50)) CMT = K                                   1MIN  32
      IF(CMT .EQ.81) GO TO 160                                          MIN  33
      NTMS = 114- NB                                                    MIN  34
      WRITE(3,150) CARD,(S(I),I = 1,CMT),(ALBET(13),I =1,NTMS),(S(I),I=  MIN  35
     A CMT,NB)                                                          MIN  36
  150 FORMAT(5X,I3,2X,120A1)                                            MIN  37
      GO TO 170                                                         MIN  38
  160 WRITE(3,150) CARD,S                                               MIN  39
  170 DO 180 I = 1,80                                                  1MIN  40
  180 IF(S(I) .EQ. ALBET(13) .OR. I .GE.CMT) S(I) = 0                  1MIN  41
      CALL RMVZER(S,K)                                                  MIN  42
      IF(K.EQ.0) GO TO 110                                              MIN  43
C                                                                       MIN  44
C     CHECK FOR DEBUG                                                   MIN  45
C                                                                       MIN  46
      IF (S(1) .NE. ALBET(46).OR.S(2).NE.ALBET(45).OR.S(3).NE.ALBET(48) MIN  47
     A .OR.S(4).NE.ALBET(29).OR.S(5).NE.ALBET(43)) GO TO 190            MIN  48
      MNPG(1,1) = 1                                                     MIN  49
      GO TO 110                                                         MIN  50
  190 CONTINUE                                                          MIN  51
      CALL COMND(S,T,R,ER)                                              MIN  52
      IF (T .EQ. 0)  GO TO 340                                          MIN  53
      IF (T .EQ. 15)  GO TO 440                                         MIN  54
      IF (T .NE. 0 .AND. ER.EQ. 0) GO TO 210                            MIN  55
```

```
  2CC CALL ERROR(26)                                                    MIN  56
      GO TC 110                                                         MIN  57
  210 GO TO (220,200,230,230,240,200,270,110,280,290,320,200),I         MIN  58
      GO TO 2C0                                                         MIN  59
  22C MNPG(EQ,1) = 1                                                    MIN  60
      MNPG(EQ,2) = R(1)                                                 MIN  61
      MNPG(EQ,3) = R(2)                                                 MIN  62
      MNPG(EQ,4) = R(3)                                                 MIN  63
      MNPG(EQ,5) = R(4)                                                 MIN  64
      MNPG(EQ,6) = R(5)                                                 MIN  65
      EQ = EQ +1                                                        MIN  66
      CO TC 11C                                                         MIN  67
C                                                                       MIN  68
C     RUN MCD STATEMENT                                                 MIN  69
C                                                                       MIN  7C
  230 CALL MDNAME(MNU,R)                                                MIN  71
      MNPG(EQ,1) = 4                                                    MIN  72
      MNPG(EQ,2) = MNU                                                  MIN  73
      MNPG(EQ,3) = RC                                                   MIN  74
      EQ = EQ + 1                                                       MIN  75
      RC = C                                                            MIN  76
      GO TO 110                                                         MIN  77
C                                                                       MIN  78
C     GET STATEMENT                                                     MIN  79
C                                                                       MIN  80
  240 NND = NGV + 40                                                    MIN  81
      MNPG(EQ,1) = 5                                                    MIN  82
      MNPG(EQ,2) = NGV                                                  MIN  83
      K = 1                                                             MIN  84
      CO 25C I = NND,NGV                                               1MIN  85
      IF(R(K) .EQ. C) GO TO 260                                        1MIN  86
      GETLST(I) = R(K)                                                 1MIN  87
  250 K = K + 1                                                        1MIN  88
  260 MNPG(EQ,3) = J                                                    MIN  89
      NGV = J + 1                                                       MIN  90
      EQ = EQ + 1                                                       MIN  91
      GO TO 110                                                         MIN  92
C                                                                       MIN  93
C     POINT  STATEMENT                                                  MIN  94
C                                                                       MIN  95
  270 VARARY(R(1)) = EQ                                                 MIN  96
      CO TO 11C                                                         MIN  97
C                                                                       MIN  98
C     READ STATEMENT                                                    MIN  99
C                                                                       MIN 1CC
  28C RD = 1                                                            MIN 101
      GO TO 110                                                         MIN 102
C                                                                       MIN 103
C     PRINT STATEMENT                                                   MIN 1C4
C                                                                       MIN 105
  290 NND = NPT + 40                                                    MIN 106
      MNPG(EQ,1) = 10                                                   MIN 107
      MNPG(EQ,2) = NPT                                                  MIN 1C8
      K = 1                                                             MIN 109
      CO 3CC I = NPT,NND                                               1MIN 110
```

```
      IF (R(K) .EQ. 0) GO TO 310                                        1MIN 111
      J = I                                                             1MIN 112
      PRTLST(I) = R(K)                                                  1MIN 113
  300 K = K + 1                                                         1MIN 114
  310 MNPG(EQ,3) = J                                                     MIN 115
      EQ = EQ + 1                                                        MIN 116
      NPT = J + 1                                                        MIN 117
      GO TO 110                                                          MIN 118
C                                                                        MIN 119
C     IF --- GO TO  STATEMENT                                            MIN 120
C                                                                        MIN 121
  320 MNPG(EQ,1) = 11                                                    MIN 122
      MNPG(EQ,2) = R(1)                                                  MIN 123
      MNPG(EQ,3) = R(2)                                                  MIN 124
      MNPG(EQ,4) = R(3)                                                  MIN 125
      MNPG(EQ,5) = R(4)                                                  MIN 126
      EQ = EQ + 1                                                        MIN 127
      GO TO 110                                                          MIN 128
  330 CEBLC = MNPG(1,1)                                                  MIN 129
      RETURN                                                             MIN 130
  340 DO 350 I = 1,16                                                   1MIN 131
      K = I                                                             1MIN 132
      NA(I) = S(I)                                                      1MIN 133
      S(I) = 0                                                          1MIN 134
      IF (NA(I) .EQ. ALBET(1))  GO TO 360                               1MIN 135
  350 NA(I+1) = C                                                       1MIN 136
  360 NA(K) = 0                                                          MIN 137
      CALL NAM(NA,K)                                                     MIN 138
      MNPG(EQ,1) = 3                                                     MIN 139
      MNPG(EQ,2) = K                                                     MIN 140
      MNPG(EQ,3) = 401                                                   MIN 141
      MNPG(EQ,4) = 1                                                     MIN 142
      MNPG(EQ,5) = 401                                                   MIN 143
      CALL RMVZER(S,K)                                                   MIN 144
      NPS = 3                                                            MIN 145
      IF (S(1) .EQ. ALBET(5))  GO TO 400                                 MIN 146
  370 IF (S(1) .GE. ALBET(23) .AND. S(1) .NE. ALBET(8))  GO TO 410       MIN 147
      DO 380 I = 1,5                                                    1MIN 148
      K = I                                                             1MIN 149
      NA(I) = S(I)                                                      1MIN 150
      IF (S(I) .EQ. ALBET(5) .OR. S(I) .EQ. ALBET(10))  GO TO 390       1MIN 151
      S(I) = C                                                          1MIN 152
  380 NA(I+1) = C                                                       1MIN 153
  390 NA(K) = 0                                                          MIN 154
      CALL NAM(NA,K)                                                     MIN 155
      MNPG(EQ,NPS) = K                                                   MIN 156
      CALL RMVZER(S,K)                                                   MIN 157
      IF (K .GT. 0 .AND. NPS .LT. 5)  GO TO 400                          MIN 158
      EQ = EQ + 1                                                        MIN 159
      GO TO 110                                                          MIN 160
  400 SGN = 1                                                            MIN 161
      IF (S(1) .EQ. ALBET(5)) SGN = -1                                   MIN 162
      S(1) = C                                                           MIN 163
      CALL RMVZER(S,K)                                                   MIN 164
      MNPG(EQ,4) = SGN                                                   MIN 165
```

```
      NPS = 5                                                           MIN 166
      GO TO 370                                                         MIN 167
  410 NST = 0                                                           MIN 168
      DO 420 I = 1,K                                                    MIN 169
      NA(I) = S(I)                                                      MIN 170
      NA(I+1) = 0                                                       MIN 171
      IF (S(I) .EQ. ALBET(45)) NST = 2                                  MIN 172
      IF (S(I) .GT. ALBET(12) .AND. NST .LE. 0)  GO TO 430              MIN 173
      NST = NST -1                                                      MIN 174
  420 S(I) = 0                                                          MIN 175
  430 CALL NAM(NA,K)                                                    MIN 176
      PNPG(EQ,NPS) = K                                                  MIN 177
      CALL RMV2ER(S,K)                                                  MIN 178
      IF (K .GT.0 .AND. NPS .LT. 5)  GO TO 370                          MIN 179
      EQ = EQ + 1                                                       MIN 180
      GO TO 110                                                         MIN 181
  440 PNPG(EQ,1)  = 2                                                   MIN 182
      PNPG(EQ,2) = R(1)                                                 MIN 183
      EQ = EQ + 1                                                       MIN 184
      GO TO 110                                                         MIN 185
      END                                                               MIN 186
```

```
      SUBROUTINE MANSET                                                 MST   1
      IMPLICIT INTEGER(A-U),REAL(V-Z)                                   MST   2
      INTEGER*2 MNPG(100,6),PRTLST(100),GETLST(100)                     MST   3
      COMMON /MAIN/ VRB(620),NAMS(400),MNPG,PRTLST,GETLST,MAINM(200)    MST   4
      INTEGER*2 EQU(100,10,4)                                           MST   5
      COMMON /MGD/ VARARY(620),NAMARY(400),EQU                          MST   6
      COMMON /BLG/ DEBUG                                                MST   7
      DO 100 I = 1,400                                                  1MST   8
      VRB(I) = VARARY(I)                                                1MST   9
  100 NAMS(I) = NAMARY(I)                                               1MST  10
      DO 110 I = 401,620                                                1MST  11
  110 VRB(I) = VARARY(I)                                                1MST  12
      IF (DEBLG .NE. 1)   GO TO 140                                     MST  13
      WRITE (3,130) (MNPG(1,J),J = 1,6)                                 MST  14
      DO 120 I = 2,100                                                  1MST  15
      IF (MNPG(I,1) .EQ. 0)  GO TO 140                                  1MST  16
  120 WRITE (3,130) (MNPG(I,J),J = 1,6)                                 1MST  17
  130 FORMAT (/,6I4)                                                    MST  18
  140 RETURN                                                            MST  19
      END                                                               MST  20
```

```
      SUBROUTINE STOMOD(MODNAM,MODNUM)
      IMPLICIT INTEGER(A-U),REAL(V-Z)
      INTEGER*2 EQU(100,10,4),PRTLST(50)
      COMMON /MOD/ VARARY(620),NAMARY(400),EQU,PRTLST
      COMMON /GEN/ ALBET(50)
      DIMENSION ST(20),MODNAM(5)
      N5 = MODNUM + 1
      IF (MODNAM(1) .NE. 1)  GO TO 100
      IF(EQU(1,1,1) .LE. 0)  RETURN
      N5 = EQU(1,1,1) + 1
      READ (5'N5) (ST(I),I=1,20)
      NREC = ST(6)
      GO TO 110
  100 READ(5'1) NMOD,NREC
      N4 = NREC
      ST(6) = NREC
  110 TYP = 1
      DO 120 I=1,100,5
      IF(EQU(I,1,1).EQ.0) GO TO 140
      S = I + 4
      WRITE(4'NREC)MODNUM,TYP,(((EQU(J,K,L),L=1,4),K=1,10),J=I,S)
      DO 120 J = I,S
      DO 120 K = 1,10
      DO 120 L = 1,4
  120 EQU(J,K,L) = 0
  130 NREC = NREC + 1
  140 ST(7) = NREC -1
      TYP = 2
      WRITE(4'NREC)MODNUM,TYP,(VARARY(I),I=201,300)
      NREC = NREC + 1
      WRITE(4'NREC)MODNUM,TYP,(VARARY(I),I=301,400)
      NREC = NREC + 1
      TYP = 3
      WRITE(4'NREC)MODNUM,TYP,(VARARY(I),I =401,500)
      NREC = NREC + 1
      WRITE(4'NREC)MODNUM,TYP,(VARARY(I),I=501,600)
      NREC = NREC + 1
      TYP = 4
      WRITE(4'NREC)MODNUM,TYP,(NAMARY(I),I = 1,100)
      NREC = NREC + 1
      WRITE(4'NREC)MODNUM,TYP,(NAMARY(I),I = 101,200)
      NREC = NREC + 1
      WRITE(4'NREC)MODNUM,TYP,(NAMARY(I),I = 201,300)
      NREC = NREC + 1
      WRITE(4'NREC)MODNUM,TYP,(NAMARY(I),I = 301,400)
      NREC = NREC + 1
      TYP = 5
      WRITE(4'NREC)MODNUM,TYP,(PRTLST(I),I=1,50)
      NREC = NREC + 1
  150 IF (MODNAM(1) .EQ. 1)  RETURN
      NMOD = NMOD + 1
      DO 160 I = 1,5
  160 ST(I) = MODNAM(I)
      WRITE(5'N5)(ST(I),I = 1,20)
      WRITE(5'1)NMOD,NREC
      RETURN
      END
```

```
      SUBROUTINE MAINRN                                                 MRN   1
      IMPLICIT INTEGER(A-U),REAL(V-Z)                                   MRN   2
      INTEGER*2 MNPG(100,6),PRTLST(100),GETLST(100)                     MRN   3
      COMMON /MAIN/ V  (620),NM  (400),MNPG,PRTLST,GETLST,MAINM(200)    MRN   4
      COMMON /ALL/ VLST(50),MODNAM(30,5)                                MRN   5
      COMMON /MOD/ VRB(620),NAMS(400)                                   MRN   6
      COMMON /COMMN/ STPCOM,STTREG,STPREG,COM(200)                      MRN   7
      WRITE (3,100)                                                     MRN   8
  100 FORMAT ('1')                                                      MRN   9
      DO 110 I = 1,STPCOM                                              1MRN  10
  110 NM(I) = COM(I)                                                   1MRN  11
      IP = 2                                                            MRN  12
      VUN = 1.23459E-13                                                 MRN  13
  120 IF (MNPG(IP,1) .EQ. 0) GO TO 270                                  MRN  14
      IGO = MNPG(IP,1)                                                  MRN  15
      GO TO (130,290,280,150,180,140,140,140,140,200,230,140),IGO       MRN  16
      IP = IP + 1                                                       MRN  17
      GO TO 120                                                         MRN  18
  130 CALL ITCHK(IP,COM)                                                MRN  19
      IF(COM .EQ. 0) IP = MNPG(IP,3) - 1                                MRN  20
      IP = IP + 1                                                       MRN  21
      GO TO 120                                                         MRN  22
  140 IP = IP + 1                                                       MRN  23
      GO TO 120                                                         MRN  24
  150 MONUM = MNPG(IP,2)                                                MRN  25
      CALL GETMOD(MONUM)                                                MRN  26
      IF (MNPG(IP,3) .EQ. 1) CALL CATAIN                                MRN  27
      IF (MNPG(1,1) .EQ. 1) WRITE (3,160)  (MODNAM(MONUM,I),I=1,5)      MRN  28
  160 FORMAT (' *****',5A4)                                             MRN  29
      CALL RUNMOD                                                       MRN  30
      IF (MNPG(1,1) .EQ. 1)  CALL BUGPRT                                MRN  31
      DO 170 I = 1,STPCOM                                              1MRN  32
  170 V(I) = VRB(I)                                                    1MRN  33
      IP = IP + 1                                                       MRN  34
      GO TO 120                                                         MRN  35
  180 S = MNPG(IP,2)                                                    MRN  36
      ST = MNPG(IP,3)                                                   MRN  37
      K = 1                                                             MRN  38
      DO 190 I = S,ST                                                  1MRN  39
      V(GETLST(I)) = VLST(K)                                           1MRN  40
  190 K = K + 1                                                        1MRN  41
      IP = IP + 1                                                       MRN  42
      GO TO 120                                                         MRN  43
  200 S = MNPG(IP,2)                                                    MRN  44
      ST = MNPG(IP,3)                                                   MRN  45
      IF (MNPG(IP-1,1) .NE. 10) WRITE (3,210)                           MRN  46
  210 FORMAT (/,5X,'CONTROL MODULE')                                    MRN  47
      WRITE(3,220) (NM(PRTLST(I)),V(PRTLST(I)),I = S,ST)                MRN  48
  220 FORMAT(5(5X,A4,' = ',1PE13.6))                                    MRN  49
      IP = IP + 1                                                       MRN  50
      GO TO 120                                                         MRN  51
  230 ISO = MNPG(IP,3)                                                  MRN  52
      IF(V(MNPG(IP,2)).EQ.VUN.OR.V(MNPG(IP,4)).EQ.VUN.OR.V(MNPG(IP,5))  MRN  53
     A .EQ.VUN)  GO TO 300                                              MRN  54
      GO TO (240,250,260),IGO                                           MRN  55
```

```
240 IF (V(MNPG(IP,2)).LT. V(MNPG(IP,4)))IP = V(MNPG(IP,5))-1          MRN  56
    IP = IP +1                                                        MRN  57
    GO TO 120                                                         MRN  58
250 IF (V(MNPG(IP,2)).EQ. V(MNPG(IP,4)))IP = V(MNPG(IP,5))-1          MRN  59
    IP = IP + 1                                                       MRN  60
    GO TO 120                                                         MRN  61
260 IF (V(MNPG(IP,2)).GT. V(MNPG(IP,4)))IP = V(MNPG(IP,5))-1          MRN  62
    IP = IP + 1                                                       MRN  63
    GO TO 120                                                         MRN  64
270 RETURN                                                            MRN  65
280 IF(V(MNPG(IP,3)).EQ.VUN.OR.V(MNPG(IP,5)).EQ.VUN)  GO TO 300       MRN  66
    V(MNPG(IP,2)) = V(MNPG(IP,3))+MNPG(IP,4)*V(MNPG(IP,5))            MRN  67
    IF (MNPG(IP,2) .LE. STPCOM)  VRB(MNPG(IP,2)) = V(MNPG(IP,2))      MRN  68
    IP = IP + 1                                                       MRN  69
    GO TO 120                                                         MRN  70
290 IP = V(MNPG(IP,2))                                                MRN  71
    GO TO 120                                                         MRN  72
300 CALL ERROR(34)                                                    MRN  73
    CALL POLMP(V(1),MNPG(100,6),0)                                    MRN  74
    RETURN                                                            MRN  75
    END                                                               MRN  76
```

```
      SUBROUTINE ITCHK(P,CON)                                           ICK   1
      IMPLICIT INTEGER(A-U),REAL(V-Z)                                   ICK   2
      INTEGER*2 M   (100,6),PRTLST(100),GETLST(100)                     ICK   3
      COMMON /MAIN/ V(620),N(400),M,PRTLST,GETLST,MAINM(200)            ICK   4
      DIMENSION IT(2,10),VAL(10)                                        ICK   5
      IF(CON .NE. -7  ) GO TO 110                                       ICK   6
      DO 100 I = 1,10                                                  1ICK   7
  100 IT(1,I) = 0                                                      1ICK   8
      RETURN                                                            ICK   9
  110 DO 120 I = 1,10                                                  1ICK  10
      K = I                                                            1ICK  11
      IF (P.EQ.IT(1,I)) GO TO 150                                      1ICK  12
  120 CONTINUE                                                         1ICK  13
      DO 130 I = 1,10                                                  1ICK  14
      K = I                                                            1ICK  15
      IF (IT(1,I).EQ. 0) GO TO 140                                     1ICK  16
  130 CONTINUE                                                         1ICK  17
  140 IT(1,K) = P                                                       ICK  18
      IT(2,K) = 1                                                       ICK  19
      CON = 0                                                           ICK  20
      IF (M(P,2) .EQ. 0) RETURN                                         ICK  21
      VAL(K) = V(M(P,2))                                                ICK  22
      RETURN                                                            ICK  23
  150 IT(2,K) = IT(2,K) + 1                                             ICK  24
      CON = 0                                                           ICK  25
      IF (M(P,2) .EQ. 0)  GO TO 160                                     ICK  26
      V1 = VAL(K)                                                       ICK  27
      V2 = V(M(P,2))                                                    ICK  28
      VAL(K) = V2                                                       ICK  29
      IF (ABS(V1-V2) .LE. V(M(P,6)))CON = 1                             ICK  30
      IF (ABS((V1-V2)/V2)*10 .LE. V(M(P,5))) CON = 1                    ICK  31
  160 IF (IT(2,K) .GE. M(P,4)) CON = 1                                  ICK  32
      IF (CON  EQ. 0) RETURN                                            ICK  33
      IT(1,K  = 0                                                       ICK  34
      IT(2,K) = 0                                                       ICK  35
      RETURN                                                            ICK  36
      END                                                               ICK  37
```

```
      SUBROUTINE GETMOD(MN)                                             GTM   1
      IMPLICIT INTEGER(A-U),REAL(V-Z)                                   GTM   2
      INTEGER*2 EQ (100,10,4),PRTLST(50)                                GTM   3
      COMMON /MOD/ VARARY(620),NAMARY(400),EQ ,PRTLST                   GTM   4
      DIMENSION ST(20),MODNAM(5)                                        GTM   5
      MODNAM(1) = 1                                                     GTM   6
      IF (EQ(1,1,1) .EQ. MN)  RETURN                                    GTM   7
      N5 = MN + 1                                                       GTM   8
      CALL STOMOD(MODNAM,MN)                                            GTM   9
      READ (5'N5) (ST(I),I = 1,20)                                      GTM  10
      NREC = ST(6)                                                      GTM  11
      DO 100 I = 1,100,5                                               1GTM  12
      IF (NREC .GT. ST(7))  GO TO 110                                  1GTM  13
      S = I + 4                                                        1GTM  14
      READ (4'NREC) NAM,TY,( ((EQ(J,K,L),L=1,4),K=1,10),J=I,S)         1GTM  15
  100 NREC = NREC + 1                                                  1GTM  16
  110 READ (4'NREC) NAM,TY,( VARARY(I),I=201,300)                       GTM  17
      NREC = NREC + 1                                                   GTM  18
      READ (4'NREC) NAM,TY,( VARARY(I),I=301,400)                       GTM  19
      NREC = NREC + 1                                                   GTM  20
      READ (4'NREC) NAM,TY,(VARARY(I),I=401,500)                        GTM  21
      NREC = NREC + 1                                                   GTM  22
      READ (4'NREC) NAM,TY,(VARARY(I),I=501,600)                        GTM  23
      NREC = NREC + 1                                                   GTM  24
      READ (4'NREC) NAM,TY,( NAMARY(I),I=1,100)                         GTM  25
      NREC = NREC + 1                                                   GTM  26
      READ (4'NREC) NAM,TY,( NAMARY(I),I=101,200)                       GTM  27
      NREC = NREC + 1                                                   GTM  28
      READ (4'NREC) NAM,TY,( NAMARY(I),I=201,300)                       GTM  29
      NREC = NREC + 1                                                   GTM  30
      READ (4'NREC) NAM,TY,( NAMARY(I),I=301,400)                       GTM  31
      NREC = NREC + 1                                                   GTM  32
      READ (4'NREC) NAM,TY,( PRTLST(I),I=1,50)                          GTM  33
      EQ(1,1,1) = MN                                                    GTM  34
      IF (EQ(2,1,1) .EQ. 1)  GO TO 120                                  GTM  35
      CALL ORDER(INO)                                                   GTM  36
      EQ(2,1,1) = 1                                                     GTM  37
  120 RETURN                                                            GTM  38
      END                                                               GTM  39
```

```
      SUBROUTINE MDNAME(NUM,NAM)                                        MDN   1
      IMPLICIT INTEGER(A-U),REAL(V-Z)                                   MDN   2
      COMMON /ALL/ VLST(50),MODNAM(30,5)                                MDN   3
      DIMENSION NAM(5)                                                  MDN   4
      DO 110 I = 1,30                                                  1MDN   5
      NUM = I                                                          1MDN   6
      IF (MODNAM(I,1) .EQ. 0)  GO TO 120                               1MDN   7
      IF (NAM(1) .NE. MODNAM(I,1))  GO TO 110                          1MDN   8
      DO 100 J = 1,5                                                   2MDN   9
      IF (NAM(J) .NE. MODNAM(I,J))  GO TO 110                          2MDN  10
  100 CONTINUE                                                         2MDN  11
      RETURN                                                           1MDN  12
  110 CONTINUE                                                         1MDN  13
      CALL ERROR(23)                                                    MDN  14
      RETURN                                                            MDN  15
  120 DO 130 I = 1,5                                                   1MDN  16
  130 MODNAM(NUM,I) = NAM(I)                                           1MDN  17
      RETURN                                                            MDN  18
      END                                                               MDN  19
```

```
      SUBROUTINE TABLIN(S)                                              TIN   1
      IMPLICIT INTEGER(A-U),REAL(V-Z)                                   TIN   2
      COMMON /GEN/ A(50)                                                TIN   3
      COMMON /MOC/ VAR    (620),AAM    (400)                            TIN   4
      COMMON /TAB/ TABL(20,2)                                           TIN   5
      DIMENSION S(80),C(16),NSZ(3),V(600),X(100),Y(100),Z(100)          TIN   6
      NSZ(1) = 1                                                        TIN   7
      NSZ(2) = 1                                                        TIN   8
      NSZ(3) = 1                                                        TIN   9
      C(1) = A(36)                                                      TIN  10
      C(2) = A(49)                                                      TIN  11
      C(3) = A(37)                                                      TIN  12
      C(4) = A(45)                                                      TIN  13
      C(5) = A(1)                                                       TIN  14
      CALL COMFNC(C,S,5,NN,PS,0)                                        TIN  15
      DO 100 I = 1,80                                                  1TIN  16
      IF (S(I) .EQ. 0)  GO TO 110                                      1TIN  17
      IF (S(I) .EQ. A(11))  LP = I + 1                                 1TIN  18
  100 IF (S(I) .EQ. A(6)) RP = I - 1                                   1TIN  19
  110 NV = 1                                                            TIN  20
      L = 0                                                             TIN  21
      DO 120 I = LP,RP                                                 1TIN  22
      L = L + 1                                                        1TIN  23
      IF(S(I) .EQ. A(3))  S(I) = 0                                     1TIN  24
      C(L) = S(I)                                                      1TIN  25
      IF (C(L) .NE. 0 .AND. LP .NE. I )  GO TO 120                     1TIN  26
      C(L+1) = 0                                                       1TIN  27
      CALL NUMCMP(C,VR)                                                1TIN  28
      NSZ(NV) = VR                                                     1TIN  29
      L = 0                                                            1TIN  30
      NV = NV + 1                                                      1TIN  31
  120 S(I) = 0                                                         1TIN  32
      NT = NV-1                                                         TIN  33
      WRITE (3,130) NAM(NM),(NSZ(I),I = 1,NT)                           TIN  34
  130 FORMAT (//,5X,'TABLE ',A4,3X,'SIZE = ',T26,I3,T30,I3,T29,',',T34,  TIN  35
     A I3,T33,',')                                                      TIN  36
      WRITE (3,250)                                                     TIN  37
      GO TO (180,160,140),NT                                            TIN  38
  140 WRITE (3,150)                                                     TIN  39
  150 FORMAT ('+',6X,'Z')                                               TIN  40
  160 WRITE (3,170)                                                     TIN  41
  170 FORMAT ('+',19X,'Y')                                              TIN  42
  180 WRITE (3,190)                                                     TIN  43
  190 FORMAT ('+',32X,'X')                                              TIN  44
      NT = NSZ(1)*NSZ(2)*NSZ(3)                                         TIN  45
      NXT = NSZ(1)                                                      TIN  46
      NYT = NSZ(2)                                                      TIN  47
      NZT = NSZ(3)                                                      TIN  48
      READ (1,200,END=300) (X(I),I=1,NXT)                               TIN  49
  200 FORMAT (8E10.5)                                                   TIN  50
      WRITE (3,250)                                                     TIN  51
      WRITE (3,210) (X(I),I=1,NXT)                                      TIN  52
  210 FORMAT ('+',27X,8(2X,1PE11.4),12(/,26X,8(2X,1PE11.4)))            TIN  53
      IF (NYT .EQ. 1)  GO TO 220                                        TIN  54
      READ (1,200,END=300) (Y(I),I=1,NYT)                               TIN  55
```

```
      IF (NZT .EQ. 1)  GO TO 220                                        TIN  56
      READ (1,200,END=300) (Z(I),I=1,NZT)                               TIN  57
  220 NS = 1                                                            TIN  58
      NST = NXT                                                         TIN  59
      WRITE (3,250)                                                     TIN  60
      DO 260 I = 1,NZT                                                 1TIN  61
      IF (NV .GT. 3) WRITE(3,240)  Z(I)                                1TIN  62
      IF (NV .LE. 3) WRITE (3,250)                                     1TIN  63
      DO 260 J = 1,NYT                                                 2TIN  64
      IF (NV .GT. 2) WRITE (3,230) Y(J)                                2TIN  65
      READ(1,200) (V(K),K=NS,NST)                                      2TIN  66
      WRITE (3,210) (V(K),K=NS,NST)                                    2TIN  67
      NS = NST + 1                                                     2TIN  68
  230 FORMAT ('+',14X,1PE11.4)                                         2TIN  69
  240 FORMAT (/,2X,1PE11.4)                                            2TIN  70
      IF (NXT .NE. 8) WRITE (3,250)                                    2TIN  71
  250 FORMAT (' ')                                                     2TIN  72
  260 NST = NS + NXT - 1                                               2TIN  73
      K = 1                                                             TIN  74
      DO 270 I = 1,20                                                  1TIN  75
      IF (TABL(I,1) .EQ. 0)  GO TO 280                                 1TIN  76
      IF (TABL(I,1) .EQ. NAM(NM))  GO TO 290                           1TIN  77
  270 K = I + 1                                                        1TIN  78
      CALL ERROR(27)                                                    TIN  79
  280 TABL(K,1) = NAM(NM)                                               TIN  80
      TABL(K,2) = NV                                                    TIN  81
  290 IF (TABL(K,2) .NE. NV)  CALL ERROR(28)                            TIN  82
      WRITE (6'K)  NT,NXT,NYT,NZT,(V(I),I=1,NT),(X(I),I=1,NXT),(Y(I),I=1 TIN  83
     A ,NYT),(Z(I),I=1,NZT)                                             TIN  84
      RETURN                                                            TIN  85
  300 CALL ERROR(30)                                                    TIN  86
      RETURN                                                            TIN  87
      END                                                               TIN  88
```

```
      SUBROUTINE INTERP(VI,NX,NY,NZ,VT,VX,VY,VZ,VRET,N2)                INT   1
      DIMENSION VI(3),VT(NX,NY,NZ),VX(NX),VY(NY),VZ(NZ)                 INT   2
      CALL GETIND(VI(1),VX,NX,NX1,VXD,N2,1)                             INT   3
      CALL GETIND(VI(2),VY,NY,NY1,VYD,N2,2)                             INT   4
      CALL GETIND(VI(3),VZ,NZ,NZ1,VZD,N2,3)                             INT   5
      NTM = 0                                                           INT   6
  100 VRET = VT(NX1,NY1,NZ1) + VXD*(VT(NX1+1,NY1,NZ1)-VT(NX1,NY1,NZ1))  INT   7
      IF (NY .LE. 1)  RETURN                                            INT   8
      VPT1 = VRET                                                       INT   9
      NY1 = NY1 + 1                                                     INT  10
      VRET = VT(NX1,NY1,NZ1) + VXD*(VT(NX1+1,NY1,NZ1)-VT(NX1,NY1,NZ1))  INT  11
      NY1 = NY1 - 1                                                     INT  12
      VPT2 = VRET                                                       INT  13
      VRET = VPT1 + VYD*(VPT2-VPT1)                                     INT  14
      IF ( NZ .EQ.1 ) RETURN                                            INT  15
      IF (NTM .EQ. 1) GO TO 110                                         INT  16
      VPT3 = VRET                                                       INT  17
      NTM = 1                                                           INT  18
      NZ1= NZ1+ 1                                                       INT  19
      GO TO 100                                                         INT  20
  110 VRET = VPT3 + VZD*(VRET-VPT3)                                     INT  21
      RETURN                                                            INT  22
      END                                                               INT  23
```

```
      SUBROUTINE GETIND(VD,VA,NT,N,VI,NTAB,NVAR)                        GID   1
      DIMENSION VA(NT)                                                  GID   2
      COMMON /TAB/ TABL(20,2)                                           GID   3
      IF (NT .GT. 1)  GO TO 100                                         GID   4
      N = 1                                                             GID   5
      VI = 0                                                            GID   6
      RETURN                                                            GID   7
  100 IF (VD .LT. VA(1)  OR. VD .GT. VA(NT))  GO TO 130                 GID   8
      DO 110 I = 2,NT                                                   GID   9
      N = I-1                                                           GID  10
      IF (VD .LE. VA(I))  GO TO 120                                     GID  11
  110 CONTINUE                                                          GID  12
  120 VI = (VD-VA(N))/(VA(N+1)-VA(N))                                   GID  13
      RETURN                                                            GID  14
  130 CALL ERROR(29)                                                    GID  15
      WRITE (3,140) TABL(NTAB,1),NVAR,VD                                GID  16
  140 FORMAT (' TABLE NAME = ',A4,'  VARIABLE NO =',I2,'   VALUE WAS ',  GID  17
     A 1PE13.6)                                                         GID  18
      N = NT-1                                                          GID  19
      IF(VD .LT. VA(1)) N = 1                                           GID  20
      GO TO 120                                                         GID  21
      END                                                               GID  22
```

```
      SUBROUTINE BUGPRT                                                 BUG   1
      IMPLICIT INTEGER(A-U),REAL(V-Z)                                   BUG   2
      COMMON /HOC/ VARARY(620),NAMARY(400)                              BUG   3
      COMMON /COMMN/ STPCOM,STTREG,STPREG,COM(200)                      BUG   4
      J = 0                                                             BUG   5
      DO 100 I = 1,STPCOM                                              1BUG   6
      IF (COM(I) .EQ. 0)  GO TO 110                                    1BUG   7
  100 J = I                                                            1BUG   8
  110 IF (J .EQ. 0)  GO TO 130                                          BUG   9
      WRITE (3,120)  (COM(I),VARARY(I),I = 1,J)                         BUG  10
  120 FORMAT (' COMMON VARIABLES',/,10(5(5X,A4,' = ',1PE13.6)/))        BUG  11
  130 J = 0                                                             BUG  12
      DO 140 I = STTREG,400                                            1BUG  13
      IF (NAMARY(I) .EQ. 0)  GO TO 150                                 1BUG  14
  140 J = I                                                            1BUG  15
  150 IF (J .EQ. 0)  RETURN                                             BUG  16
      WRITE (3,160) (NAMARY(I),VARARY(I),I = STTREG,J)                  BUG  17
  160 FORMAT (' NON COMMON VARIABLES',/,10(5(5X,A4,' = ',1PE13.6)/))    BUG  18
      RETURN                                                            BUG  19
      END                                                               BUG  20
```

APPENDIX II
MODEL LISTINGS

UTILITY MODEL

```
COMMON GW,WFL,WFL1,WFL2,ERR
COMMON GW1,GW2,WPLO,WPL1
COMMON PTO,PRA,SFC
COMMON GWO,WEM
COMMON WEM,WP,TAS,NEN,YAR,SL,N,DPL,WCR,WFL,CPH,PRA,DGW,MFH
COMMON R,PTO,SFO,F
COMMON INDX,DISC,ALT,OAT
COMMON IDXA,IDXO
COMMON WPL2
POINT ONE
RUN MOD SIZE
GWH = GW
RUN MOD T O ALLOWANCE AND CRUISE
RUN MOD MAX RANGE
RUN MOD TOTAL FUEL
GW = GWH
RUN MOD STAT WEIGHT
RUN MOD GROSS WEIGHT
ITERATE ON GW, FROM ONE, ATOL = 20
WFL1 = WFL
RUN MOD STORE OLD DATA
PRINT ALT,OAT
PRINT GW,WEM,WFL,WPLO,PTC,PRA,SFC
POINT POAT
POINT PALT
RUN MOD NEW SIZE DATA
ITR = 0
POINT TWO
RUN MOD SIZE
GWH = GW
RUN MOD T O ALLOWANCE AND CRUISE
RUN MOD MAX RANGE
RUN MOD TOTAL FUEL
GW = GWH
WFL2 = WFL
RUN MOD ANAL WEIGHT
GW2 = GW
RUN MOD GROSS WEIGHT TWO
GW1 = GW
GWH = GW
RUN MOD T O ALLOWANCE AND CRUISE
RUN MOD MAX RANGE
RUN MOD TOTAL FUEL
GW = GWH
WFL1 = WFL
RUN MOD PAYLOAD
GW = GW2
RUN MOD ERROR
ITR = ITR + 1
IF ITR IS GT MXTR,  GO TO THRE
IF ERR IS GT 20,  GOTO TWO
POINT THRE
PRINT ALT,OAT,ITR
PRINT GW1,$WEM2,WFL1,WPL1,PTC,PRA,SFC
PRINT GW2,WFL2,WPL2
```

```
DGW = GW2
WEM = WEM2
DPL = WPL2
RUN MOD PAPC COSTS
RUN MOD JMAX              FIND MAXIMUM VALUE FOR J - JMAX = 0/DPH
$I = 0
$OCE = 0
POINT FOUR
$I = $I + 1
RUN MOD PAYLOADS
ITAT = 0
POINT FIVE
RUN MOD DGWS
GWH = GW
RUN MOD T O ALLOWANCE AND CRUISE
RUN MOD MAX RANGE
RUN MOD TOTAL FUEL
GW = GWH
ITAT = ITAT + 1
ITERATE ON $WFL,PTOL=1,FROM FIVE
RUN MOD MAINT COSTS
RUN MOD HOVER PROB
RUN MOD MEI
PRINT $OCE,$MEI,$CPFH,$PUF,$PHCV,$PL,$GW,$CLF
IF WN ISGT $I,GO TO FOUR
IDXA = IDXA + 1
ITERATE FROM PALT,TIMES = 5
IDXA = 1
IDXO = IDXO + 1
ITERATE FROM POAT, TIMES = 2
```

```
IDXJ = 1
IDXA = 1
MXTR = 25
NP = 1000
TAS = 150
YAR = 40
SL = 10
N = 5
MCR = 400
DPM = .1
MFH = 100
TABLE NAME = EUF,SIZE = (2)
0          1.5
1.0        1.0
TABLE NAME = BELT , SIZE = (9)
12.2  E+03 60.0 E+03 10.0 E+04 25.5 E+04 54.0 E+04 10.0 E+05 20.0 E+05 44.0 E+05
12.5  E+06
1.5   E-03 1.35 E-03 1.2   E-03 8.5   E-04 6.75  E-04 5.0   E-04 3.6   E-04 3.5   E-04
5.6   E-04
TABLE NAME = HEAD , SIZE = (7)
10000.     47667.5    373320.    783597.    3.5887 E+6 1.1164E+7 1.0    E+8
1.2    E-2 6.542 E-3 2.904 E-3 2.237 E-3 1.819 E-3 1.274 E-3 0.0
TABLE NAME = PUF,SIZE = (2)
0          1.5
.33333     .33333
TABLE NAME = TMAX,SIZE = (14)
0          500.       1000.      2000.      3000.      4000.      5000.      6000.
7000.      8000.      9000.      10000.     11000.     12000.
49.77      45.25      44.44      43.85      42.5       40.55      39.64      37.78
35.33      35.        25.67      21.11      12.75      12.5
TABLE NAME = NPRO,SIZE = (14)
0          .1995      .3295      .5308      .6809      .7769      .8462      .8941
.9338      .9635      .9792      .9919      .9998      1.0
0          500.       1000.      2000.      3000.      4000.      5000.      6000.
7000.      8000.      9000.      10000.     11000.     12000.
TABLE NAME = PYLD,SIZE = (11)
1.0        2.0        3.0        4.0        5.0        6.0        7.0        8.0
9.0        10.0       11.0
40.0       50.0       80.0       100.0      110.0

TABLE NAME = PUF ,SIZE = (11)
1.0        2.0        3.0        4.0        5.0        6.0        7.0        8.0
9.0        10.0       11.0
15.0       25.0       30.0       25.0       5.0

TABLE NAME = TPRO,SIZE = (24,14)
-23.23     -17.79     -12.22     -6.67      -3.89      -1.11      1.67       4.44
7.22       10.0       12.78      15.56      18.33      21.11      23.89      26.67
29.44      32.22      35.0       37.78      40.56      43.33      46.11      48.89
0          500.       1000.      2000.      3000.      4000.      5000.      6000.
7000.      8000.      9000.      10000.     11000.     12000.
0          0          0          0          0          0          0          0
.0096      .0213      .0373      .0568      .0755      .1040      .1444      .2147
.3242      .5143      .7236      .9697      .0416      .9810      .9971      1.0
0          0          0          0          0          0          0          .0062
```

.0139	.0271	.0445	.0665	.0925	.1251	.1642	.2817
.4117	.5707	.7302	.8535	.9360	.9614	1.0	1.0
0	0	0	0	0	0	0	.0053
.0115	.0222	.0362	.0557	.0830	.1280	.2043	.3158
.4556	.6175	.7524	.8514	.9130	.9611	1.0	1.0
0	0	0	0	0	0	0	.0080
.0100	.0297	.0480	.0726	.1076	.1660	.2941	.4295
.5796	.7268	.8373	.9290	.9743	.9900	1.0	1.0
0	0	0	0	0	0	.0053	.112
.0233	.0414	.0636	.1055	.1497	.2522	.3730	.5280
.6791	.8048	.8530	.9475	.9857	1.0	1.0	1.0
0	0	0	0	0	0	.0040	.0149
.0305	.0535	.0844	.1382	.2149	.3276	.4692	.6252
.7554	.8568	.9268	.9712	1.0	1.0	1.0	1.0
0	0	0	0	0	0	.0083	.0209
.0400	.0670	.1183	.1896	.2907	.4196	.5567	.6746
.8063	.8837	.9417	.9303	1.0	1.0	1.0	1.0
0	0	0	0	0	.0030	.0120	.0262
.0443	.0821	.1473	.2410	.3687	.5206	.6060	.7833
.8674	.9278	.9548	1.0	1.0	1.0	1.0	1.0
0	0	0	0	0	.0054	.0149	.0264
.0596	.1241	.2141	.3545	.5224	.6852	.7970	.9963
.9327	.9630	.9919	1.0	1.0	1.0	1.0	1.0
0	0	0	0	0	.0033	.0102	.0561
.1296	.2151	.3367	.5046	.6751	.6027	.5763	.9355
.9639	.9837	1.0	1.0	1.0	1.0	1.0	1.0
0	0	0	0	.0059	.0316	.1398	.2663
.4235	.6050	.7603	.9244	.9704	.9966	.9983	1.0
1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
0	0	.0018	.0144	.0640	.2031	.3702	.5186
.5777	.6551	.9542	.9948	.9976	1.0	1.0	1.0
1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
0	0	.0088	.1420	.2892	.4377	.6238	.7995
.9281	.9790	.9998	1.0	1.0	1.0	1.0	1.0
1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
0	.07	.078	.4800	.5900	.700	.7900	.8600
.9500	.9500	1.0	1.0	1.0	1.0	1.0	1.0
1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0

TABLE NAME = MAP,SIZE =(8,4)

	1.0	E-32.0	E-3 3.0	E-3 5.0	E-3 7.0	E-3
6.0	6.0	8.0	10.0			
428.0	E-4653.0	E-4670.0	E-4690.0	E-4694.0	E-4700.0	E-4
518.0	E-4559.0	E-4585.0	E-4611.0	E-4642.0	E-4663.0	E-4
475.0	E-4532.0	E-4570.0	E-4600.0	E-4645.0	E-4680.0	E-4
459.0	E-4517.0	E-4530.0	E-4627.0	E-4680.0	E-4710.0	E-4

TABLE NAME = MAPS,SIZE = (6,4)

	1.0	E-3 2.0	E-3 3.0	E-3 5.0	E-3 7.0	E-3
4.0	6.0	8.0	10.0			
66.0	58.0	52.0	47.0	41.0	35.0	
84.0	69.0	62.0	57.0	50.0	45.0	
93.0	79.0	69.0	64.0	56.0	51.0	
118.0	88.0	76.0	69.0	59.0	53.0	

TABLE NAME = TAS,SIZE =(8,4,12)

	1.0	E-32.0	E-33.0	E-34.0	E-35.0	E-36.0	E-37.0	E-3
4.0	6.0	8.0	10.0					

.06	.065	.07	.075	.08	.09	.10	.12
.14	.16	.18	.20				
126.0	92.0	73.5					
154.1	107.0	83.9	68.5				
142.8	111.2	81.6					
43.1	56.4						
137.8	103.7	86.0	74.2	66.0	59.2		
142.0	118.0	93.9	81.2	72.3	62.0		
159.0	122.0	94.0	79.8				
93.0	80.5	70.5	62.5	52.5	45.0		
146.4	113.1	94.8	83.9	75.3	70.1	65.1	60.0
149.0	125.6	102.5	89.0	80.3	73.8	67.6	62.4
172.5	129.8	105.0	89.5	79.0	70.0	58.5	
110.0	90.0	80.3	73.0	57.3	62.5	57.0	53.0
153.2	120.2	102.0	90.5	82.2	76.0	71.3	67.5
175.0	132.1	105.5	95.5	86.5	80.0	75.0	70.2
175.5	136.2	111.8	95.5	86.5	78.6	72.9	67.1
119.2	99.0	87.6	80.0	74.5	70.0	66.0	52.1
159.3	126.5	105.0	96.2	88.1	82.1	76.5	72.8
179.0	135.0	116.0	101.4	91.7	85.0	80.0	75.5
178.0	148.0	117.5	102.5	92.4	84.3	78.8	73.8
134.5	112.0	99.6	90.7	84.6	80.0	76.0	72.5
169.4	136.8	118.2	105.2	97.0	90.5	84.8	81.0
185.5	148.5	125.5	111.5	100.6	93.1	87.7	83.4
181.5	151.5	127.0	111.5	101.5	92.7	87.2	82.5
145.2	120.2	109.9	99.6	92.7	87.4	83.0	79.5
177.7	146.1	126.2	113.0	104.3	97.5	91.5	87.4
190.0	158.1	133.2	119.2	108.2	100.0	95.2	89.4
183.5	157.5	134.3	119.8	109.8	100.1	93.5	88.8
161.0	140.1	124.8	112.5	104.0	97.5	94.5	90.5
191.0	162.0	135.8	125.6	116.8	108.4	102.1	97.8
199.0	169.4	146.2	130.9	119.7	111.7	104.9	99.2
186.0	164.5	146.0	131.5	120.5	111.5	104.5	98.7
173.2	154.9	136.2	125.7	119.4	110.0	104.1	99.2
202.5	170.5	151.0	130.2	120.0	110.0	111.3	105.0
206.8	180.0	155.6	140.6	129.0	120.0	114.2	107.5
189.0	168.5	151.5	139.5	128.5	119.6	112.7	107.0
184.8	161.7	146.2	134.6	125.2	116.3	112.5	107.0
211.0	182.5	160.0	145.0	133.5	125.5	119.0	113.5
213.4	187.3	165.5	149.5	137.0	128.0	121.3	114.7
192.0	171.0	156.7	144.5	135.0	126.3	119.7	114.7
195.0	170.7	155.2	142.2	133.0	125.3	119.3	113.8
218.0	189.	167.5	152.5	141.0	132.7	126.0	120.7
218.5	192.5	171.6	155.5	143.6	134.5	127.2	120.1
196.0	174.0	159.0	148.0	138.0	129.2	122.6	117.3
200.0	178.3	162.4	149.0	139.5	131.5	125.1	119.6
225.0	196.0	174.7	159.0	147.0	138.2	131.5	126.0
225.0	198.6	177.1	161.6	149.3	140.1	132.1	124.8
196.0	176.0	161.5	150.5	141.0	133.0	126.0	120.5

TABLE NAME = MAG.SIZE = (8,4,10)

	1.0	E-3 2.0	E-33.0	E-34.0	E-3 5.0	E-36.0	E-37.0	E-3
4.0	6.0	8.0	10.0					
70.0	80.0	90.0	100.0	110.0	120.0	130.0	140.0	
150.0	160.0							

528.0	E-4065.0	E-4651.0	E-4731.0	E-4768.0	E-4800.0	E-4833.0	E-4871.0	E-4
526.0	E-4560.0	E-4591.0	E-4630.0	E-4668.0	E-4700.0	E-4730.0	E-4773.0	E-4
504.0	E-4535.0	E-4571.0	E-4601.0	E-4643.0	E-4676.0	E-4714.0	E-4749.0	E-4
521.0	E-4556.0	E-4590.0	E-4626.0	E-4663.0	E-4698.0	E-4735.0	E-4769.0	E-4
445.0	E-4095.0	E-4748.0	E-4800.0	E-4850.0	E-4901.0	E-4952.0	E-1006.	E-4
520.0	E-4570.0	E-4423.0	E-4676.0	E-4733.0	E-4775.0	E-4840.0	E-4890.0	E-4
436.0	E-4536.0	E-4590.0	E-4643.0	E-4698.0	E-4750.0	E-4800.0	E-4853.0	E-4
464.0	E-4547.0	E-4599.0	E-4651.0	E-4704.0	E-4761.0	E-4804.0	E-4848.0	E-4
470.0	E-4740.0	E-4819.0	E-4893.0	E-4965.0	E-4100.0	E-4115.	E-4193.	E-4
520.0	E-4594.0	E-4651.0	E-4745.0	E-4821.0	E-4893.0	E-4970.0	E-4049.	E-4
478.0	E-4554.0	E-4630.0	E-4700.0	E-4783.0	E-4850.0	E-4934.0	E-4013.	E-4
476.0	E-4551.0	E-4626.0	E-4704.0	E-4779.0	E-4850.0	E-4950.0	E-4020.	E-4
706.0	E-4838.0	E-4903.0	E-4100.	E-4104.	E-4120.	E-4137.	E-4149.	E-4
530.0	E-4030.0	E-4736.0	E-4837.0	E-4930.0	E-4143.	E-4153.	E-4251.	E-4
475.0	E-4586.0	E-4685.0	E-4736.0	E-4893.0	E-4100.	E-4163.	E-4220.	E-4
494.0	E-4563.0	E-4673.0	E-4778.0	E-4853.0	E-4100.	E-4115.	E-4228.	E-4
784.0	E-4904.0	E-4100.	E-4125.	E-4273.	E-4100.	E-4155.	E-4433.	E-4
552.0	E-4880.0	E-4818.0	E-4958.0	E-4104.	E-4123.	E-4177.	E-4505.	E-4
480.0	E-4513.0	E-4752.0	E-4883.0	E-4103.	E-4100.	E-4135.	E-4461.	E-4
450.0	E-4556.0	E-4735.0	E-4880.0	E-4101.	E-4100.	E-4133.	E-4478.	E-4
505.0	E-4972.0	E-4112.	E-4125.	E-4147.	E-4100.	E-4143.	E-4201.	E-4
550.0	E-4748.0	E-4821.0	E-4105.	E-4126.	E-4142.	E-4162.	E-4179.	E-4
490.0	E-4903.0	E-4838.0	E-4101.	E-4120.	E-4100.	E-4152.	E-4180.	E-4
490.0	E-4900.0	E-4825.0	E-4101.	E-4120.	E-4141.	E-4100.	E-4203.	E-4
573.0	E-4106.	E-4128.	E-4149.	E-4172.	E-4195.	E-4224.	E-4237.	E-4
415.0	E-4823.0	E-4105.	E-4127.	E-4150.	E-4175.	E-4190.	E-4220.	E-4
505.0	E-4733.0	E-4750.0	E-4113.	E-4142.	E-4100.	E-4127.	E-4222.	E-4
471.0	E-4700.0	E-4938.0	E-4117.	E-4143.	E-4101.	E-4227.	E-4273.	E-4
556.0	E-4119.	E-4174.	E-4175.	E-4201.	E-4229.	E-4253.	E-4100.	E-4
493.0	E-4935.0	E-4120.	E-4151.	E-4174.	E-4199.	E-4223.	E-4100.	E-4
533.0	E-4813.0	E-4110.	E-4135.	E-4150.	E-4199.	E-4243.	E-4100.	E-4
401.0	E-4785.0	E-4110.	E-4115.	E-4120.	E-4230.	E-4314.	E-4100.	E-4
1045.	E-4136.	E-4162.	E-4204.	E-4237.	E-4200.	E-4271.	E-4100.	E-4
324.0	E-4104.	E-4137.	E-4176.	E-4201.	E-4230.	E-4202.	E-4100.	E-4
579.0	E-4921.0	E-4125.	E-4140.	E-4200.	E-4230.	E-4272.	E-4100.	E-4
523.0	E-4851.0	E-4135.	E-4196.	E-4257.	E-4317.	E-4100.	E-4100.	E-4
1163.	E-4156.	E-4193.	E-4234.	E-4278.	E-4321.	E-4100.	E-4100.	E-4
806.0	E-4105.	E-4150.	E-4204.	E-4250.	E-4300.	E-4100.	E-4100.	E-4
638.0	E-4105.	E-4147.5	E-4193.	E-4240.	E-4290.	E-4100.	E-4100.	E-4
590.	E-4106.	E-4187.	E-4300.	E-4100.	E-4100.	E-4100.	E-4100.	E-4

TABLE NAME = SECP , SIZE = (19)

.2	.25	.30	.35	.40	.45	.50	.55
.6	.65	.70	.75	.80	.85	.90	.95
1.0	2.0						
2.12	1.92	1.52	1.40	1.32	1.25	1.20	1.16
1.13	1.10	1.08	1.06	1.04	1.03	1.02	1.01
1.0	1.0						

```
  MODULE NAME = SIZE
COMMON GW,PTO,PRA,SFO,Q,F,R,NEN,S,CB,NMR,TS,DL
COMMON VRC,PM
COMMON ALT,OAT
COMMON RM,MOW,VM,HP2,HP1,EN
PR = (1-ALT/145300)**5.255                                  PRESSURE RATIO
TR = (OAT+273.16)/288.16                                    TEMPERATURE RATIO
DR = PR/TR                                                  DENSITY RATIO
PTO = (.051*GW*(DL/DR)**.41+2.424E-5*VRC*GW)/(1-PM/100)     POWER AT TAKEOFF
PRA = PTO/(PR*(1-2.09*(TR-1)))                              TOTAL ENGINE RATED HP
SFO = 1.136*(PRA/NEN)**-.105                                SFC AT RATED HP
Q = 310*PTO*TS**-1*GW**.5*DL**-.5          TORQUE AT ENGINE OUTPUT BUT MR RPM
F = .035*GW**.667                          CURRENT UTILITY HELO W/O EXTERNAL STORES
R = .5642*GW**.5*DL**-.5                                    MR RADIUS
S = DL/BL                  SOLIDITY
CB = 3.1416*R*S/NMR                MAIN ROTOR CHORD
RM = R
MOW = GW
VM = TS
HP2 = PRA
HP1 = PTO
EN = NEN
```

```
ALT = 4000, CAT=35, VRC=500, PM=5, NEN=2, CL = 8,TS = 700
GW = 10000
BL = 80
NMR = 4
```

```
MODULE NAME = T O ALLOWANCE AND CRUISE
COMMON GW,K,PRA,E,PTO,DISC,WFL,SFO
COMMON TTIM
ALT = 0
TIM = 8
WFL = 0
PR = (1-ALT/145300)**5.255          PRESS RATIO
FL = 0.15 * PRA * PR * TIM/60          FUEL INCR
WFL=WFL+FL                          TOTAL FUEL USED
GW = GW-FL                          FINAL GW
TTIM = TIM
ALT = 4000
DAT = 35
SM = 10
KPP = .9174
DIS = 50
TAS = 130
RA = .9
PR = (1-ALT/145300)**5.255          PRESS RATIO
TR = (DAT+273.15)/288.16            TEMP. RATIO
DR = PR/TR                          DENS. RATIO
DLN = GW/(3.1416*R**2*DR)
PA = PRA*(PR*(1-2.08*(TR-1)))          MAX. HP. AVAIL.
FOW = F*DR/GW                       DRAG-WEIGHT PARAM
P  = PA * RA
IF P IS GT PTO,P=PTO
SAS = 70-SM+15.7*(15-DLN)/((FOW*1000)**.25983)          STALL KTAS
TAS = SAS
IF TAS IS GT 160, TAS = 160
POW=TABLE PRQ(FOW,DLN,TAS)
PRQ = POW * GW/KPP
IF PRQ IS LT P,P = PRQ
POW = P * KPP/GW
TAS = TABLE TAS(FOW,DLN,POW)
TIM = 60*DIS/TAS
PWF = P/(PR*TR**.5*PRA)
SSF = TABLE SFCP(PWF)
FL = TIM * P * SFO * SSF/60
WFL = WFL+FL
GW = GW-FL
TTIM = TTIM + TIM
```

```
MODULE NAME = MAX RANGE
COMMON Gw,R,PRA,E,PTO,DISC,WFL,SFO,WFL
COMMON TTIM
INITIALIZE SAR=0,TAS=80,DEL=5,FL1 = 0
INITIALIZE TIM1 = 0
ITERATE ON TAS,ATOL=1              A/S FOR MAX RANGE
PR = (1-ALT/145300)**5.255          PRESS RATIO
TR = (OAT+273.16)/288.16            TEMP. RATIO
DR = PR/TR                          DENS. RATIO
DLN = Gw/(3.1416*R**2*DR)
PA = PRA*PR*(1-2.03*(TR-1))         MAX. HP. AVAIL.
FOW = F*DR/Gw                       DRAG-WEIGHT PARAM
P = .9*PA
IF P IS GT PTO,P=PTO
POW=TABLE PRQ(FOW,DLN,TAS)
PRQ=POW*Gw/KPP
PRF=PRQ/(PRA*PR*TR**.5)
SSF=TABLE SFCP(PRF)
SAR1=SAR
SAR = TAS/(SSF*SFO*PRQ)
DSAR = SAR - SAR1
SAS = 70-SM+15.7*(15-DLN)/((FOW*1000)**.2598)        STALL KTAS
IF TAS IS GT SAS,DEL = 0
IF DSAR IS GT 0,TAS = TAS+DEL
TIM = 60 * DIS/TAS
FL = TIM * PRQ * SFO * SSF / 60
WFL = WFL + FL - FL1
FL1 = FL
TTIM = TTIM + TIM - TIM1
TIM1 = TIM
```

```
ALT = 4000
OAT = 35
SM = 10
KPP = .9174
DIS = 160
```

```
MODULE NAME = TOTAL FUEL
COMMON WFL,CAP
FL = (RCP/100)*WFL*(1-(RCP/100))        FUEL INCREMENT
WFL = WFL+FL
CAP = WFL/6.5
```

RCP = 10

```
MODULE NAME = STAT WEIGHT
NOORDER
COMMON RM,MOW,VM,HP2,HP1,CAP,EN,S,CB,NMR,BODY,WCR,WEM
COMMON TAIL,WLG1
COMMON WXS,WLS
WENG = 1.396*EN*(HP2/EN)**.733
WBL = .151E4*((NGW*NULT/100)/(AR**.999*VM **.331*DMNR**.65*NOF**.41))   BLADE WT
NGW = 2.145*MOW**.893      BLADE WEIGHT SUPPORTING CALCULATIONS
B = 2 * RM
AR = B**2/AB              BLADE ASPECT RATIO
AB = RM*CB*NMR        TOTAL BLADE AREA(SQ.FT.)
DMNR = DM*NR          DISC LOADING * NUMBER OF ROTORS
DM = MOW/3.141592/RM**2          DISC LOADING
NOF = 14.259*S**.409
WHUB = .0104*(WBL*.5*RM*RPM**2*1E-6)**1.127*QMR**.405*NOF**.058/IY**.059
RPM = 9.549*VM/RM                    MAIN ROTOR RPM
QMR = 5250 *HP1/RPM
IY = .6296E-2*MOW**1.635
FLD = .12*(WBL+WHUB)*BF
WTR = KTR*1E-6*(RT*C*N*VM**2)**1.29                    TAIL ROTOR
C = .092*RM**.651
N = 1.062*RT**.773
WHS = .0049*ATF**.5*LTF**1.78*(DM*10)**.2*TAF**.28
ATF = 1.433*DM**1.53              HORIZONTAL STABILIZER CALCULATIONS
LTF = .96*RM**1.02
KLU = LU*(W+H)
FUS=KFUS*KLU**.03*LM**.05*NGW*QMR**1.15/((MOW*WTRD)**.23* TTR**.42*IX**.38)
LU = .255*RM**1.474    FUSELAGE  LENGTH          FUSELAGE WEIGHT CALCULATIONS
W = 2.53*IX**.115    FUSELAGE WIDTH,MAX
IX = .221E-3*MOW**1.807      ROLL INERTIA
H = .406*SINH**1.226     FUSELAGE HEIGHT, MAX
SINH = .981*RM**.743    = LDIA*SIND--USE SINH WHEN LDIA*SIND UNKNOWN
WTRD = .41*W**1.75    MAIN WHEEL TREAD
TTR = 13392.66*HP2/(RPM*RM)
AFUS = .183*LTRP**.95*TAF**.13*FUS**.44/(QMR/(TTR*LTB))**.45
LTRP = 1.57*RT**.334        TAIL ROTOR PYLON LENGTH
RT = .067*RM**1.22    TAIL ROTOR RADIUS
LTB = 36.07/RM**.273      LENGTH OF TAILBOOM
WPY = .063*PA**.78*TTR**.42*TPY**.24                    PYLON WEIGHT
PA = .36*( 5*RT*LTRP)**.48*TPY**.74          PYLON PROFILE AREA
WLG1 = .1904*KLG*MOW**.87*(AG+2)**.61*WTRD**2.48/IX**.46
FC = .588E-6*NGW**1.07*NOF**.30*VM**1.898*NR**.45/LDIA**.69
LDIA = 1.717*RM**1.005
WFS = .068*KNAC*(FUS+AFUS+WPY)**.25*(WENG/ENF)**1.06*VN**.81/HP2**.55
ENF  = .279*HP2**.235
VN = 26.51*HP2**.257*ENF**.37              NACELLE VOLUME
WPTS = .434E-3*DMNR**1.82*NOF**1.69*(1.23*RM)**3.87/QMR1**.68/RR**.14
QMR1 = 52.50*HP1/RPM       MAIN ROTOR TORQUE/100
RR = 2.5.7/RPM**.174              MAIN TRANSMISSION REDUCTION RATIO
WFAC = .018*EN**.08*WENG**1.93/HP2**.59      ENGINE ACCESSORY WEIGHT
WCS  = .25*WLS     COOLING SYSTEM WEIGHT
WLS = .0145*RR**.5*WPTS**.196*CAP**1.01                    LUBRICATION SYSTEM
WFS = 3.6*CAP**.71       FUEL SYSTEM
WFC = 3.75*ENF**1.63*.3*RM**.61
WSS = 3.16*(SHP/HP2)**1.11*(WENG/ENF)**1.37    STARTING SYSTEM
```

```
SHP = .0245*HP2**.837          STARTING HORSEPOWER
WXS1 = 0.2815*WMR5**.75     MAIN TRANSMISSION
QMR5 = 5250 *HP2/RPM
ACPV = .0127*WMR6**.766          ACCESSORY DRIVE PROVISION
QMR6 = 5250*HP1/RPM     MAIN ROTOR TORQUE
WITR = KITR*WMR2**.625*ITR                    INTERMEDIATE TAIL ROTOR GEARBOX
QMR2 = .07*WMR
WTRB = KTRB*WMR3**.65                                   TAIL ROTOR GEARBOX
QMR3 = .15*WMR
WTRS = KTRS*1E-2*QMR3**.31*(1.23*RM)**1.64
WEDS = 19.197*(QMR/1000)**.17*EN**1.25                  ENGINE DRIVE SHAFT
WRR = .219*QMR**.44*DPM**.10*BRK
DPM = WBL*RM**2/9273.6          MAIN ROTOR BLAD INERTIA
APU = 46.52*ACPV**.0598*TPL     AUXILIARY POWER PLANT GROUP
WFIX = .17DE5*CP**.32*(LL*(W+H))**.14*CM**.04*ENF**2.6/NGM**.62   FIXED WEIGHT
NGM = 2.145*MUM**.893                NORMAL GROSS WEIGHT
CM = 1.39*MUM**.172    USE WEIGHTED VALUES IF KNOWN
WPTR = WBL + WMUB + FLD
TAIL = WTR + WHS
BODY = FUS + AFUS + WPY
ASE = .15 * FC
WFC = FC + ASE
PROP = WENG + SDG + WEAC + WCS1 + WLS1 + WES + WEC + WSS + WUS
LPC = WCS2 + WLS2
WDS = WXS + ACPV + WITR + WTRB + WTRS + LPC + WEDS + WRS + WRB
WXS = WXS1 - WRS
WEA = WIND + WEXM
WEM = WRTR+TAIL+BODY+WLG1+WFC+WES+PROP+WFIX+APU +WING
WCS1 = .15 * WLS
WLS1 = .60 * WLS
WES = 3.599 * CAP**.711
WPS = .18 * WXS1
WIND = .53 * WEAC
WEXM = .47 * WEAC
WING = 4.36 * SW
WCS2 = .1 * WLS
WLS2 = .4 * WLS
```

```
DF = 1        FOLDING ROTOR
ITR = 1         INTERMEDIATE TR GEARBOX
BRK = 1       ROTOR BRAKE
KFUS=  .0328                                                  CONVENTIONAL GEARED
KTR = .183                                                    CONVENTIONAL GEARED
KITR = .2296        STATISTICAL BASELINE CONSTANT    FOR  INTERMEDIATE T. ROT. GB
KTRB = .2279                      TAIL ROTOR GEARBOX CONST--------STAT.  BASELINE
KTRS = .058           TAIL ROTOR DRIVE SHAFT CONST.-----------------STAT. BASELINE
NR = 1
NULT = 4.5
TAF = 13
TPY = 62
KLG = .0329
AG = 1
KNAC = .96
CP = 2
SDG  = 0.0
SW = 0
TPU = 0
```

```
MODULE NAME = GROSS WEIGHT
COMMON WEM,WFL,WCR,WPL,GW
GW = WEM + WFL + WCR + WPL
```

NPL = 2640

```
MODULE NAME = STORE OLD DATA
COMMON WPL,ALT,DAT,Q,GW,WEM,WPLO,ALTO,DATO,QO,GWO,WEMO
WPLO = WPL
ALTO = ALT
DATO = DAT
QO = Q
GWO = GW
WEMO = WEM
$WEMO = WEM
```

```
MODULE NAME = NEW SIZE DATA
COMMON ALT,DAT,VSC,PM,DL,T,WEN,GW
COMMON ALTO,DATO,CO,U,GWC,GW,WEN2,WEP
COMMON WPL,WPLO
COMMON IDXA,IDXO
WPL = WPLO
Q = QO
GW = GWO
WEN = WWENO
ALT = (IDXA-1)*1000
DAT = 45 - 10*IDXO
```

```
MODULE NAME = ANAL WEIGHT
COMMON W,WDKV,WESG,WRTR,WEM,CO,WEM2
COMMON GW,GWJ,BODY
COMMON TAIL,WLG1
COMMON WXS,WDS
DELW = (Q-QO)*12              CHANGE IN TORQUE IN IN-LB
QILB = Q*12                TORQUE IN IN-LB
DEL1 = TABLE DELT(QILB)
DEL2 = TABLE BLAD(QILB)
DEL3 = GW/GWJ-1
WEM2 = WEM +(BODY+TAIL+WLG1)*DEL3 + DEL1*DELC*WDS/WXS + DEL2*DELC
```

```
MODULE NAME = GROSS WEIGHT TWO
COMMON WCM2,WFL1,WPL,WCR,GW
GW = WCM2+WFL1+WPL+WCR
```

```
MODULE NAME = PAYLOAD
COMMON DATU,ALTU,VRC,PM,GW,R,PRA,GW1,WPL1,WEM2
COMMON WFL1,WCR
COMMON WPL2 ,GW2,WFL2
PR = (1-ALTU/145300)**5.255
TR = (DATU+273.16)/288.16
DR = PR/TR
PTU1 = PRA*PR*(1-2.08*(TR-1))
DL1 = GW/(3.1416*R**2)
GW1 = PTU1*(1-PM/100)/(.051*(DL1/DR)**.41+2.424E-5*VRC)
WPL1 = GW1-WEM2-WCR-WFL1
WPL2 = GW2 - WEM2 - WCR - WFL2
```

```
MODULE NAME = ERROR
COMMON ERR,WPL0,WPL1
COMMON GW
GW = GW + 1.5*(WPL0-WPL1)
ERR = ((WPL0-WPL1)*(WPL0-WPL1))**.5
```

```
MODULE NAME = PAPC COSTS
COMMON WEM,NP,TAS,PRA,NEN,YAR,SL,WEM,CPIA,CD
WA = 126 +.749*WEM
CET = 1.21 * WA*(220/NP + .75*NP**-.15)
CL = 1011 * WA**.55*NP**-.39
CM = 9.31E-5*WA*TAS**1.24*NP**-.12
                    CE =PRA*(58 - 6E-3*PRA/NEN)
IF PRA LSUT 3500,CE =NEN*(1.295E5 + 37*(PRA/NEN -3500))
CG = -3760 + 4.75*WA
CP = CET + CL + CM + CE + CG
CI = .1*CP
CA = YAR*CP*SL/NP
CC = 101 + 3.325E-3*WEM
CPOL = 0
CPIA = CP + CI + CA
CD = CC + CPOL
```

```
MODULE NAME = JMAX
$JMAX = 1.0/$CPM
```

```
MODULE NAME = PAYLOADS
COMMON PUF,*PL2,PL,I
PPL = TABLE PYLD(I)
PL = PPL/100*PL2
PUF = TABLE PUF(I)
PUF = PUF / 100
```

```
MODULE NAME = OGWS
$GW = $mEM + $mCR + $WEL + $PL
```

```
MODULE NAME = MAINT COSTS
COMMON OLF,GW,DGW,WEM,CPIA,WFL,SL,CPFH,CD
COMMON WFL,TTIM
OLF = GW/DGW
TN = .577G + 5544/WEM
T = .593*TN/((OLF + .01771)/1.151**3
CFM = TN/T*(-67.6 + .05362*WEM)
COPL = TN/T*(16.14 + .00233*WEM)
IF WEM ISLT 3200, CFM = TN/T*(.00191*WEM**1.323)
IF WEM ISLT 3200, COPL =TN/T*(1.016*WEM**.4)
CMT = CFM + COPL
CPFH = 5(PIA/(12*WFL*SL) + CD + CMT
CFUL = WFL/6.5*.25
CPFH = CPFH + CFUL*50/TTIM
PRINT CPIA,CFM,CD,CFUL,TTIM
```

```
MODULE NAME = HOVER PRCB
COMMON TMAX,JMAX,DPM,DL,GW,DGW,VRC,PRA,PM,AT,PT,A,J,PTD
INITIALIZE J = 0,A = 0,PT = 0,DELT = 1
PM = 1 - J*DPM
H = TABLE HPRD(PM)
TMAX = TABLE TMAX(H)
APR = (1 - H/14*300)**5.255
CON1 = DL * GW**3.435/(DGW*APR)
CON2 = 47.51E-5*VRC*GW
CON3 = PRA*APR*(1-PM/100)
ATR = 0
ATR =1.481-.02452*((CON1*ATR)**.41+CON2)/CON3
ATR =1.461-.02452*((CON1*ATR)**.41+CON2)/CON3
ATR =1.461-.02452*((CON1*ATR)**.41+CON2)/CON3
ATR =1.461-.02452*((CON1*ATR)**.41+CON2)/CON3
ATR =1.461-.02452*((CON1*ATR)**.41+CON2)/CON3
ATR =1.461-.02452*((CON1*ATR)**.41+CON2)/CON3
ATT = ((PTO*(1-PM/100)-2.42E-5*GW*VRC)/(1.051*GW))**(1./.41)/(DL*GW/(DGW*APR))
IF ATT IS LT ATR, ATR = ATT
AT = 286.16 * ATR -273.16
PT1 = PT
AT1 = AT
IF AT1 IS LT -23, AT1 = -23
PT = TABLE TPRD(AT1,H)
A = DPM*(1-(PT+PT1)/2) + A
IF J IS EQ 0 , A = 0
IF AT IS GT TMAX , DELT = 0
IF J IS GT JMAX, J = J - DELT
J = J + DELT
ITERATE ON J , ATOL = .5
```

CONTROL MODULE FOR TRANSPORT MODEL

```
COMMON GW,WFL,WFL1,WFL2,ERR
COMMON GW1,GW2,WPLO,WPL1
COMMON PTO,PRA,SFO
COMMON GWO,WEM
COMMON WEM,NP,TAS,NEN,YAR,SL,N,DPL,WCR,WFL,DPH,PRA,DGW,WFH
COMMON R,PTO,SFO,F
COMMON INDX,DISC,ALT,OAT
COMMON IDXA,IDXO
COMMON WPL2
POINT ONE
RUN MOD SIZE
RUN MOD  T O ALLOWANCE
RUN MOD CLIMB
RUN MOD CRUISE AT MCP
RUN MOD FLIGHT IDLE
RUN MOD CLIMB
RUN MOD MAX RANGE
RUN MOD TOTAL FUEL
RUN MOD STAT WEIGHT
RUN MOD GROSS WEIGHT
ITERATE ON GW, FROM ONE, ATOL = 20
WFL1 = WFL
RUN MOD STORE OLD DATA
PRINT ALT,OAT
PRINT GW,WEP,WFL,WPLO,PTO,PRA,SFO
RUN MOD JMAX
POINT POAT
POINT PALT
RUN MOD NEW SIZE DATA
ITR = 0
POINT TWO
RUN MOD SIZE
RUN MOD  T C ALLOWANCE
RUN MOD CLIMB
RUN MOD CRUISE AT MCP
RUN MOD FLIGHT IDLE
RUN MOD CLIMB
RUN MOD MAX RANGE
RUN MOD TOTAL FUEL
WFL2 = WFL
RUN MOD ANAL WEIGHT
GW2 = GW
RUN MOD GROSS WEIGHT TWO
GW1 = GW
RUN MOD  T C ALLOWANCE
RUN MOD CLIMB
RUN MOD CRUISE AT MCP
RUN MOD FLIGHT IDLE
RUN MOD CLIMB
RUN MOD MAX RANGE
RUN MOD TOTAL FUEL
RUN MOD PAYLOAD
RUN MOD ERROR
ITR = ITR + 1
IF ITR IS GT MXTR, GO TO THRE
IF ERR IS GT 20.  GOTO TWO
POINT THRE
PRINT ALT,OAT,ITR
PRINT GW1,SWEM2,WFL1,WPL1,PTO,PRA,SFO
```

```
60  PRINT GW2,WFL2,WPL2
61  RUN MOD PAPC COSTS
62  SI = 0
63  SOCS = 0
64  POINT FOUR
65  SI = SI + 1
66  RUN MOD PAYLOADS
67  ITAT = 0
68  POINT FIVE
69  RUN MOD OGWS
70  RUN MOD  F C ALLOWANCE
71  RUN MOD CLIMB
72  RUN MOD CRUISE AT MCP
73  RUN MOD FLIGHT IDLE
74  RUN MOD CLIMB
75  RUN MOD MAX RANGE
76  RUN MOD TOTAL FUEL
77  ITAT = ITAT + 1
78  ITERATE ON SWFL,PTOL=1,FROM FIVE
79  RUN MOD MAINT COSTS
80  RUN MOD HOVER PROB
81  RUN MOD MEI
82  PRINT SOCE,SMEI,SCPFH,SPUF,SPHOV,SPL,SGW,SCLF
83  IF SN ISGT SI,GO TO FOUR
84  IDXA = IDXA + 1
85  ITERATE FROM PALT,TIMES = 5
```

CONTROL MODULE FOR OBSERVATION MODEL

```
COMMON GW,WFL,WFL1,WFL2,ERR
COMMON GW1,GW2,WPLC,WPL1
COMMON PTO,PRA,SFO
COMMON GWO,WEM
COMMON WEM,AP,TAS,NEN,YAR,SL,N,CPL,WCR,WFL,DPH,PRA,CGW,MFH
COMMON R,PTO,SFO,F
COMMON INDX,DISC,ALT,CAT
COMMON IDXA,IDXO
COMMON WPL2
POINT ONE
RUN MOD SIZE
RUN MOD MISSION FUEL
RUN MOD STAT WEIGHT
RUN MOD GROSS WEIGHT
ITERATE ON GW, FROM ONE, ATCL = 20
WFL1 = WFL
RUN MOD STORE OLD DATA
PRINT ALT,OAT
PRINT GW,WEM,WFL,WPLO,PTO,PRA,SFO
POINT POAT
POINT PALT
RUN MOD NEW SIZE DATA
ITR = 0
POINT TWO
RUN MOD SIZE
RUN MOD ANAL WEIGHT
RUN MOD MISSION TIME
WFL2 = WFL
GW2 = GW
RUN MOD GROSS WEIGHT TWO
GW1 = GW
RUN MOD MISSION FUEL
WFL1 = WFL
RUN MOD PAYLOAD
GW = GW2
RUN MOD ERROR
ITR = ITR + 1
IF ITR IS GT MXTR,  GO TO THRE
IF ERR IS GT 20, GOTO TWO
POINT THRE
PRINT ALT,OAT,ITR
PRINT GW1,$WEM2,WFL1,WPL1,PTO,PRA,SFO
PRINT GW2,WFL2,WPL2
DGW = GW2
WEM = WEM2
DPL = WPL2
GW = GW2
$EUF = 1
$OCE = 0
RUN MOD PAPC COSTS
RUN MOD MAINT COSTS
RUN MOD MEI
PRINT $OCE,$MEI,$CPFH,$MIST,GW
IDXA = IDXA + 1
ITERATE FROM PALT,TIMES = 5
```

CONTROL MODULE FOR CRANE MODEL

```
COMMON GW,WFL,WFL1,WFL2,ERR
COMMON GW1,GW2,WPLO,WPL1
COMMON PTO,PRA,SFO
COMMON GWO,WEM
COMMON WEM,AP,TAS,MEN,YAR,SL,N,DPL,WCR,WFL,DPH,PRA,DGW,MFH
COMMON K,PTO,SFO,F
COMMON INDX,DISC,ALT,OAT
COMMON IDXA,IDXO
COMMON WPL2
POINT ONE
RUN MOD SIZE
RUN MOD FLIGHT IDLE
RUN MOD T O ALLOWANCE
RUN MOD HOVER
RUN MOD MAX RANGE
GW = GW + DPL
F = F + DF
RUN MOD HOVER
RUN MOD MAX RANGE
RUN MOD TOTAL FUEL
RUN MOD STAT WEIGHT
RUN MOD GROSS WEIGHT
ITERATE ON GW, FROM ONE, ATOL = 20
WFL1 = WFL
RUN MOD STORE OLD DATA
PRINT ALT,OAT
PRINT GW,WEM,WFL,WPLO,PTO,PRA,SFO
POINT POAT
POINT PALT
RUN MOD NEW SIZE DATA
ITR = 0
POINT TWO
RUN MOD SIZE
RUN MOD FLIGHT IDLE
RUN MOD T O ALLOWANCE
RUN MOD HOVER
RUN MOD MAX RANGE
GW = GW + DPL
F = F + DF
RUN MOD HOVER
RUN MOD MAX RANGE
RUN MOD TOTAL FUEL
WFL2 = WFL
RUN MOD ANAL WEIGHT
GW2 = GW
RUN MOD GROSS WEIGHT TWO
GW1 = GW
RUN MOD FLIGHT IDLE
RUN MOD T O ALLOWANCE
RUN MOD HOVER
RUN MOD MAX RANGE
GW = GW + DPL
F = F + DF
RUN MOD HOVER
RUN MOD MAX RANGE
RUN MOD TOTAL FUEL
WFL1 = WFL
RUN MOD PAYLOAD
GW = GW2
```

```
60  RUN MOD ERROR
61  ITR = ITR + 1
62  IF ITR IS GT MXTR, GO TO THRE
63  IF ERR IS GT 20, GOTO TWO
64  POINT THRE
65  PRINT ALT,OAT,ITR
66  PRINT GW1,$WEM2,WFL1,WPL1,PTC,PRA,SFO
67  PRINT GW2,WFL2,WPL2
68  DGW = GW2
69  WEM = WEM2
70  DPL = WPL2
71  RUN MOD PARC COSTS
72  RUN MOD JMAX
73  $I = 0
74  $OCE = 0
75  POINT FOUR
76  $I = $I + 1
77  RUN MOD PAYLOADS
78  ITAT = 0
79  POINT FIVE
80  RUN MOD OGWS
81  RUN MOD FLIGHT IDLE
82  RUN MOD T O ALLOWANCE
83  RUN MOD HOVER
84  RUN MOD MAX RANGE
85  GW = GW + DPL
86  F = F + DF
87  RUN MOD HOVER
88  RUN MOD MAX RANGE
89  RUN MOD TOTAL FUEL
90  ITAT = ITAT + 1
91  ITERATE ON $WFL,PTOL=1,FROM FIVE
92  RUN MOD MAINT COSTS
93  RUN MOD HOVER PROB
94  RUN MOD MEI
95  PRINT $OCE,$MEI,$CPFH,$PUF,$PHCV,$PL,$GW,$GLF
96  IF $N ISGT $I,GO TO FOUR
97  IDXA = IDXA + 1
98  ITERATE FROM PALT,TIMES = 5
99  IDXA = 1
100 IDXO = IDXO + 1
101 ITERATE FROM POAT, TIMES = 2
```

CONTROL MODULE FOR GUNSHIP MODEL

```
COMMON GW,WFL,WFL1,WFL2,ERR
COMMON GW1,GW2,WPLC,WPL1
COMMON PTC,PRA,SFO
COMMON GWO,WEM
COMMON WEM,AP,TAS,NEN,YAR,SL,N,DPL,WCR,WFL,DPH,PRA,CGW,MFH
COMMON R,PTO,SFO,F
COMMON INDX,DISC,ALT,CAT
COMMON IDXA,IDXO
COMMON WPLZ
POINT ONE
RUN MOD SIZE
RUN MOD CRUISE HOVER AND VMAX
RUN MOD MAX RANGE
RUN MOD TOTAL FUEL
RUN MOD STAT WEIGHT
RUN MOD CROSS WEIGHT
ITERATE ON GW, FROM ONE, ATOL = 20
WFL1 = WFL
RUN MOD STORE OLD DATA
PRINT ALT,OAT
PRINT GW,WEM,WFL,WPLO,PTO,PRA,SFO
POINT POAT
POINT PALT
RUN MOD NEW SIZE DATA
ITR = 0
POINT TWO
RUN MOD SIZE
RUN MOD CRUISE HOVER AND VMAX
RUN MOD MAX RANGE
RUN MOD TOTAL FUEL
WFL2 = WFL
RUN MOD ANAL WEIGHT
GW2 = GW
RUN MOD GROSS WEIGHT TWO
GW1 = GW
RUN MOD CRUISE HOVER AND VMAX
RUN MOD MAX RANGE
RUN MOD TOTAL FUEL
WFL1 = WFL
RUN MOD PAYLOAD
GW = GW2
RUN MOD ERROR
ITR = ITR + 1
IF ITR IS GT MXTR,  GO TO THRE
IF ERR IS GT 20,  GOTO TWO
POINT THRE
PRINT ALT,OAT,ITR
PRINT GW1,$WEM2,WFL1,WPL1,PTO,PRA,SFO
PRINT GW2,WFL2,WPL2
DGW = GW2
WEM = WEM2
DPL = WPL2
RUN MOD PAPC COSTS
RUN MOD JMAX
$I = 0
$OCE = 0
POINT FOUR
$I = $I + 1
RUN MOD PAYLOADS
```

```
60  ITAT = 0
61  POINT FIVE
62  RUN MOD OGWS
63  RUN MOD CRUISE HOVER AND VMAX
64  RUN MOD MAX RANGE
65  RUN MOD TOTAL FUEL
66  ITAT = ITAT + 1
67  ITERATE ON $WFL,$TOL=1,FROM FIVE
68  RUN MOD MAINT COSTS
69  RUN MOD HOVER PROB
70  RUN MOD MEI
71  PRINT $OCE,$MEI,$CPFH,$PUF,$PHOV,$PC,$GW,$OCF
72  IF $N ISGT $I,GO TO FCUR
```

```
MODULE NAME = MEI
$PHUV = 1 - $A
$MEI = '/L*$PHUV
$OCE = $OCE+$MEI/$CPFH*$PUF
```

LIST OF SYMBOLS

A	joint probability plot area
ALT	altitude, ft
APR	atmospheric pressure ratio
AR	aspect ratio
AT	atmospheric temperature - °F
ATR	atmospheric temperature ratio
BL	blade loading - lb/ft^2
CA	attrition costs - $
CB	main rotor blade chord - ft
CC	crew costs - $/flight hr
CD	direct costs - $/flight hr
CE	engine cost - $
CET	cost of engineering and tooling - $
CFM	field maintenance cost - $/flight hr
CG	cost of GFE - $
CI	initial spares cost - $
CL	cost of labor - $
CM	cost of materials - $
CMT	total maintenance cost - $/flight hr
COPL	overhaul parts and labor cost - $/flight hr
CP	total aircraft production cost - $
CPFH	total cost per flight hour - $/flight hr
CPIA	production, initial spares and attrition costs - $
CPOL	fuel cost - $/flight hr

LIST OF SYMBOLS (Continued)

DE	design value of endurance - min
DGW	design gross weight - lb
DIS	distance - n mi
DL	disc loading - lb/ft^2
DLN	normalized disc loading - DL/DR
DPH	incremental altitude probability width
DPL	design payload or change in payload - lb
DR	density ratio - ρ/ρ_o
E	endurance - min
EN	number of engines
EUF	endurance utilization frequency
F	equivalent flat plate drag area - ft^2
FL	mission segment fuel load - lb
FOW	ratio of drag area times DR to GW - (F)(DR)/GW
FRC	forward rate of climb - ft/min
GW	gross weight - lb
GW0	single-point design gross weight - lb
GW1	first design point gross weight - lb
GW2	second design point gross weight - lb
H	altitude - ft
HP	horsepower
KPP	power reduction - MRHP/SHP
MEI	mission effectiveness index
MFH	average monthly flight hours - hr

LIST OF SYMBOLS (Continued)

MOW	maximum overload gross weight, lb
MRHP	main rotor horsepower
N	number of payload increments
NEN	number of engines
NMR	number of main rotor blades
NP	number of ships produced (fleet size)
OAT	outside air temperature - °C
OCE	overall cost effectiveness index
OE	operating endurance - min
OLF	overload factor - %
OPL	operating payload - lb
PA	power available - hp
PHOV	hover probability
PM	power margin - %
POW	power-to-weight ratio - MRHP/GW
PR	pressure ratio
PRA	intermediate rated power of engine @ SL, 59°F - hp
PRF	referred power - hp
PRQ	power required for level flight - hp
PT	temperature probability
PTO	power at transmission torque limit - hp
PT1	previous value of PT
PUF	payload utilization frequency
Q	torque limit of transmission - ft-lb

LIST OF SYMBOLS (Continued)

QMR	main rotor torque - ft-lb
R	main rotor radius - ft
RA	maximum continuous power rating factor
RCP	percent fuel for reserve - %
R/C	rate of climb
RPL	relative payload (to design)
S	solidity = α
SAR	specific air range - n mi/lb
SAS	stall limited air speed - kn
SFC	specific fuel consumption - lb/hr-hp
SFO	specific fuel consumption at PRA - lb/hr-hp
SL	system life - years
SM	airspeed margin to stall - kn
SSF	normalized specific fuel consumption
T	actual average MTBF - hr
TAS	true airspeed - kn
TIM	time - min
TMAX	maximum temperature at a given altitude - °F
TN	normal average MTBF - hr
TR	temperature ratio
TS	main rotor tip speed - ft/sec
VRC	vertical rate of climb - ft/min
WA	AMPR weight - lb
WCR	weight of crew - lb

LIST OF SYMBOLS (Continued)

WEM	empty weight of ship - lb
WFL	fuel weight - lb
WPL	payload - lb
YAR	average yearly attrition rate - no./yr
Ω	rotor speed - rad/sec