TARADCOM
R and D LABORATORY
TECHNICAL REPORT

NO. 12503

U.S. ARMY TARADCOM

NATO REFERENCE MOBILITY MODEL, EDITION I

USERS GUIDE
VOLUME I

CONTRACT NO. DAAK30-77-C-0027
OCT 1979

by
PETER W. HALEY
TARADCOM
M. PETER JURKAT
PETER M. BRADY, JR.
STEVENS INSTITUTE OF TECHNOLOGY
DAVIDSON LABORATORY
HOBOKEN, NJ 07030

Approved for public release; distribution unlimited.

U.S. ARMY TANK-AUTOMOTIVE RESEARCH AND DEVELOPMENT COMMAND
Warren, Michigan 48090

Approved for public release, distribution unlimited

Technical Report 12503

NATO REFERENCE MOBILITY MODEL, EDITION I
USERS GUIDE

VOLUME I

OPERATIONAL MODULES

DA Project 1L162601AH91

Prepared by

Stevens Institute of Technology
Davidson Laboratory
Castle Point Station
Hoboken, NJ 07030

Contract DAAK30-77-C-0027

OCT 1979

for

US Army Tank-Automotive Research & Development Command
Warren, MI 48090

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF APPENDICES

I INTRODUCTION AND OVERVIEW*

The NATO Reference Mobility Model (NRMM) is a collection of equations and algorithms designed to simulate the cross-country movement of vehicles. It was developed from several predecessor models, principally AMC-74 (Jurkat, Nuttall and Haley (1975)). This report, in several volumes, provides some background and motivation for most aspects of the model, and presents documentation for the coded version now available through the U. S. Army Tank-Automotive Research and Development Command (TARADCOM).

A. Background

Rational design and selection of military ground vehicles requires objective evaluation of an ever-increasing number of vehicle system options. Technology, threat, operational requirements, and cost constraints change with time. Current postures must be reexamined, new options evaluated, and new trade-offs and decisions made. In the single area of combat vehicles, for example, changes in one or another influencing factor might require trade-offs that run the gamut from opting for an air or ground system, through choosing wheels, tracks or air cushions, to designating a new tire.

The former Mobility Systems Laboratory of the then U. S. Army Tank-Automotive Command (TACOM) and the U. S. Army Engineer Waterways Experiment Station (WES) are the Army agencies responsible for

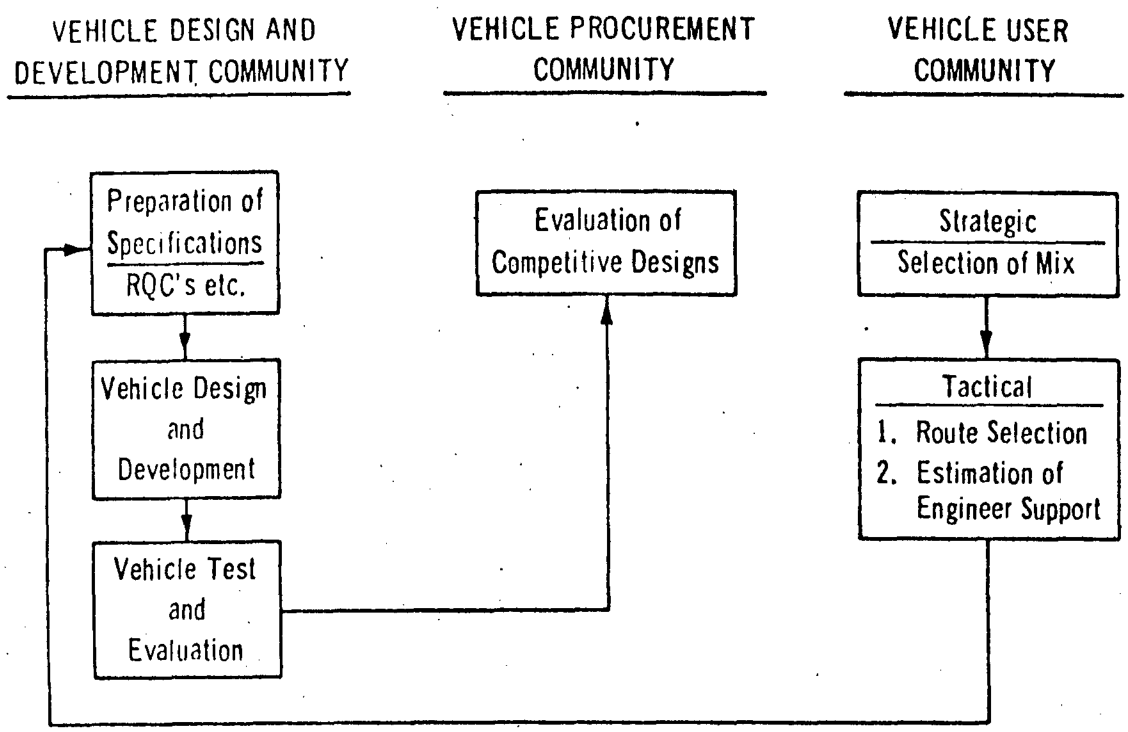
* This chapter is adapted from Jurkat, Nuttall and Haley (1975).

conducting ground mobility research. In 1971, a unified U. S. ground mobility program, under the direction of the then Army Materiel Command (AMC), was implemented that specifically geared the capabilities of both laboratories to achieve common goals.

As a first step in the unified program, a detailed review was made of existing vehicle mobility technology and of the problems and requirements of the various engineering practitioners associated with the military vehicle life cycle. One basic requirement was identified as common to all practitioners surveyed: the need for an objective analytical procedure for quantitatively assessing the performance of a vehicle in a specified operational environment. This is the need that is addressed to a substantial extent by the INRMM and its predecessors.

In theory, a single methodology can serve some of the needs of all major practitioners, provided it relates vehicle performance to basic characteristics of the vehicle-driver-terrain system at appropriate levels of detail.

Three principal categories of potential users of the methodology were identified: the vehicle development community, the vehicle procurement community, and the vehicle user community (Figure I.A.1). The greatest level of detail is needed by the design and development engineer (vehicle design and development community) who is interested in subtle engineering details--for example, wheel geometry, sprung masses, spring rates, track widths, etc.--and their



PROSPECTIVE USERS OF VEHICLE PERFORMANCE PREDICTION METHODOLOGY

FIGURE I-A-1

interactions with soil strength, tree stems of various sizes and spacings, approach angles in ditches and streams, etc. At the other end of the spectrum is the strategic planner (user community), who is interested in such highly aggregated characteristics as the average cross-country speed of a given vehicle throughout a specified region--the net result of many interactions of the engineering details with features of the total operational environment. Between these two extremes, is the person responsible for selection of the vehicles who must evaluate the effect of changes of major subsystems or choose from concepts of early design stages. To be responsive to the needs of all three user communities, the methodology must be flexible enough to provide compatible results at many levels and in an appropriate variety of formats.

Interest in a single, unified methodology applicable to the needs of these three principal users led to the creation of a cross-country vehicle computer simulation combining the best available knowledge and models of the day. Much of this knowledge was collected in Rula and Nuttall (1971). The first realization of the simulation was a series of computer programs known as the AMC-71 Mobility Model, called AMC-71 for short (US ATAC(1973)). This model first became operational in 1971; it was published in 1973. It was conceived as the first generation of a family whose descendants, under the evolutionary pressures of subsequent research and validation testing results, application experiences, and growing user requirements, would be characterized by greater accuracy and applicability. A relatively current status report may be found in Nuttall, Rula and Dugoff (1974).

The first descendant, known as AMC-74, is the basis for the INRMM. It is documented in Jurkat, Nuttall and Haley (1975). The following is a description of this model.

B. Modeling Off-Road Vehicle Mobility

In undertaking mobility modeling, the first question to be answered was the seemingly easy one: What is mobility? The answer had been elusive for many years. Semantic reasons can be traced to the beginnings of mobility research, but there was also a pervasive reluctance to accept the simple fact that even intuitive notions about a vehicle's mobility depend greatly on the conditions under which it is operating. By the mid-1960s, however, a consensus had emerged that the maximum feasible speed-made-good* by a vehicle between two points in a given terrain was a suitable measure of its intrinsic mobility in that situation.

This definition not only identified the engineering measure of mobility, but also its dependence on both terrain and mission. When, at a suitably high resolution, the terrain involved presents the identical set of impediments to vehicle travel throughout its extent, mobility in that terrain (ignoring edge effects) is the vehicle's maximum straight-line speed as limited only by those impediments. But when, as is typically the case, the terrain is not so homogeneous, the problem immediately becomes more complex. Maximum speed-made-good then becomes an interactive function of terrain variations, end points specified, and the path selected. (Note that the last two constitute at least part of a detailed mission statement.) As a way to achieve a useful simulation in this complicated situation the INRMM deliberately

*Speed-made-good between two points is the straight-line distance between the points divided by total travel time, irrespective of path.

simplifies the real areal terrain into a mosaic of terrain units within each of which the terrain characteristics are considered sufficiently uniform to permit use of the simple, maximum straight-line speed of the vehicle to define its mobility in, along, or across that terrain unit. A terrain unit or segment specified for a road or trail is, similarly, considered to have uniform characteristics throughout its extent.

Maximum speed predictions are made for each terrain unit without concern for whether or not distances within the unit are adequate to permit the vehicle to reach the predicted maximum. This vehicle and terrain-specific speed prediction is the basic output of the model. The model, in addition, generates data that may be used to predict operational vibration levels, mission fuel consumption, etc., and can provide diagnostic information as to the factors limiting speed performance in the terrain unit.

The speed and other performance predictions for all terrain units in an area can be incorporated into maps that specify feasible levels of performance that a given vehicle might achieve at all points in the area. At this point, the output is reasonably general and is essentially independent of mission and operational scenario influences. The basic data constituting the maps must usually be further processed to meet the needs of specific users. These needs vary from relatively simple statistics or indices reflecting overall vehicle compatibility with the terrain, to extensive analyses involving detailed or generalized missions. None of these so called post-processors is included as part of the INRMM.

C. Overall Structure of the INRMM

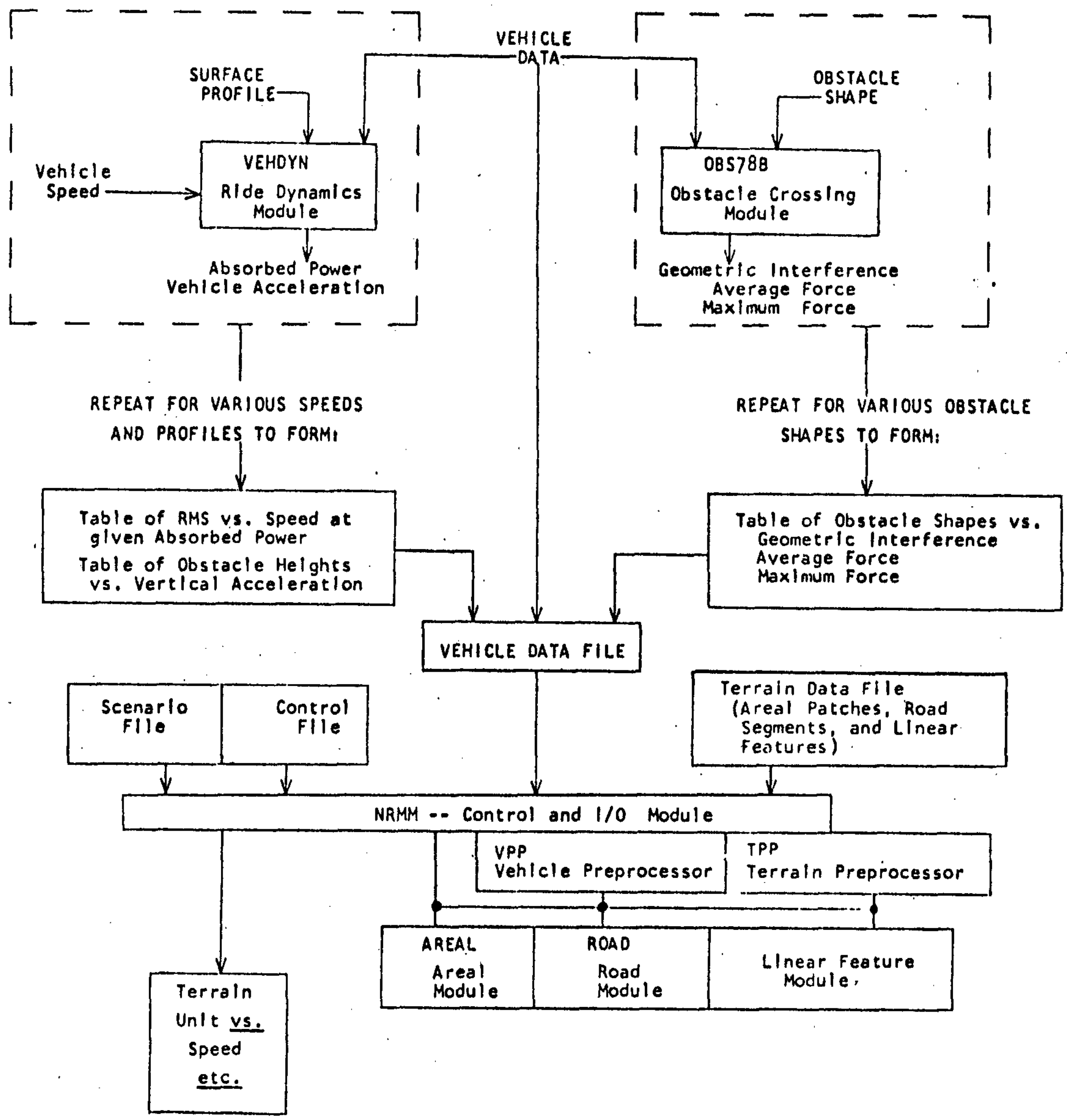
In formulating AMC-71, it was recognized that its ultimate usefulness to decision makers in the vehicle development, procurement, and user communities would depend upon its realism and credibility. (See Nuttall and Dugoff (1973).) These perceived requirements led to several more concrete objectives related to the overall structure of the model. It was determined that the model should be designed to:

1. Allow validation by parts and as a whole.
2. Make a clear distinction between engineering predictions and any whose outcome depends significantly upon human judgment, with the latter kept visible and accessible to the model user.
3. Be updated readily in response to new vehicle and vehicle-terrain technology.
4. Use measured subsystem performance data in place of analytical predictions when and as available and desired.

These objectives, plus the primary goal of supporting decision making relating to vehicle performance at the several levels, clearly dictated a highly modular structure that could both provide and accept data at the subsystem level, as well as make predictions for the vehicle as a whole. The resulting gross structure of the model is illustrated in Figure I.C.1.

At the heart of the model are three independent computational modules, each comprised of analytical relations derived from laboratory and field research, suitably coupled in the particular type of operation. These are:

FIGURE 1.C.1 - GENERAL ORGANIZATION OF THE INITIAL NATO REFERENCE MOBILITY MODEL



1. The Areal Module, which computes the maximum feasible speed for a single vehicle in a single areal terrain unit (patch).

2. The Linear Feature Module, which computes the minimum feasible time for a single vehicle, aided or unaided, to cross a uniform segment of a significant linear terrain

feature such as a stream, ditch, or embankment (not currently available).

3. The Road Module, which computes the maximum feasible speed of a single vehicle traveling along a uniform segment of a road or trail.

These Modules and the Terrain and Vehicle Preprocessors are collected in a computer program called NRMM and are described in this volume.

These three Modules may be used separately or together. Alternately, INRMM has the ability to simulate travel from terrain unit to terrain unit in the sequence given by the terrain input file. In this mode, known as the traverse mode, sufficient output data can be provided so that the user may calculate acceleration and deceleration times and distances between and across terrain unit boundaries, and thereby determine actual travel time and speed-made-good over a chosen route.

All three modules draw from a common data base that describes quantitatively the vehicle, the driver, and the terrain to be examined in the simulation. The general content of the data base is shown in Table I.C.1.

TABLE I.C.1

Terrain, Vehicle, Driver Attributes Characterized in INRMM Data Base

Terrain	Vehicle	Driver
Surface Composition Type Strength	Geometric characteristics	Reaction Times
	Inertial characteristics	Recognition distance
Surface Geometry Slope Altitude Discrete Obstacles Roughness Road Curvature Road Width Road Superelevation	Mechanical characteristics	Acceleration and impact tolerances
		Minimum acceptable speeds
Vegetation Stem Size Stem Spacing		
Linear Geometry Stream cross section Water velocity Water depth		

D. Model Inputs and Preprocessors

1. Terrain

For the purposes of the model, each terrain unit is described at any given time by values for a series of 22 mathematically independent terrain factors for an areal unit (including lake and marsh factors), 10 for the cross section of a linear feature to be negotiated, and 9 to quantify a road segment . General-purpose terrain data also include separate values for several terrain factor values that vary during the year. For example, at present such general data for areal terrain include four values for soil strength (dry, average, wet, and wet-wet seasons) and four seasonal values for recognition distances in vegetated areas. Similar variations in effective ground roughness, resulting from seasonal changes in soil moisture (including freezing) and in the cultivation of farm land, can be envisioned for the future. Further details on the terrain factors used are given in Rula and Nuttall (1975).

As discussed earlier, the basic approach to representing a complex terrain is to subdivide it into areal patches, linear feature segments, or road segments, each of which can be considered to be uniform within its bounds. Besides supplying actual values for the terrain factors, this concept may be implemented by dividing the range of each individual terrain factor value into a number of class intervals, based upon considerations of vehicle response sensitivity and practical measurement and mapping resolution problems. A patch or a segment is then defined by the condition that the class interval designator for each factor involved is the same throughout. A new patch or segment is defined whenever one or more factors fall into a new class interval.

Before being used in the three computational Modules, the basic terrain data are passed through a Terrain Data Preprocessor, called TPP, in the Computer Program NRMM. This preprocessor does three things:

1. Converts as necessary all data from the units in which they are stored to inches, pounds, seconds and radians, which are used throughout the subsequent performance calculations.
2. Selects prestored soil strengths and visibility distances according to run specifications, which are supplied as part of the scenario data (see below).
3. Calculates from the terrain measurements in the basic terrain data a small number of mathematically dependent terrain variables used repeatedly in the computational modules.

2. Vehicle

The vehicle is specified in the vehicle data base in terms of its basic geometric, inertial, and mechanical characteristics. The complete vehicle characterization as used by the performance computation modules includes measures of dynamic response to ground roughness and obstacle impact, and the clearance and traction requirements of the vehicle while it is negotiating a parametric series of discrete obstacles.

The model structure permits use at these points of appropriate data derived either from experiments or from supporting stand-alone simulations used as preprocessors. Available as modules of the INRMM is a two-dimensional ride and obstacle crossing Dynamics Module for obtaining requisite dynamics responses (currently called VEHDYN and described in Volume III) and an Obstacle Module for computing obstacle crossing traction requirements and interferences (currently called OBS78B and described in Volume II). Both derive some required information from the basic vehicle data base, and both, when used, constitute stand-alone vehicle data preprocessors.

There is also a Vehicle Data Preprocessor called VPP (integral to NRMM) which, like the Terrain Data Preprocessor, has three functions:

1. Conversion of vehicle input data to uniform inches, pounds, seconds, and radians.
2. Calculation, from the input data, of controlling soil performance parameters and other simpler dependent vehicle variables subsequently used by the computational modules, but usually not readily measured on a vehicle or available in its engineering specifications.
3. Computation of the basic steady-state traction versus speed characteristics of the vehicle power train, from engine and power train characteristics.

As in the case of dynamic responses and obstacle capabilities, the last item, the steady-state tractive force-speed relation, may be input directly from proving ground data, when available and desired.

3. Driver

The driver attributes used in the model characterize the driver in terms of his limiting tolerance to shock and vibration and his ability to perceive and react to visual stimuli affecting his behaviour as a vehicle controller. While these attributes are identified in Figure I.C.1 and Table I.C.1 as part of the data base INRMM provides for their specific identification and user control so that the effects of various levels of driver motivation, associated with combat or tactical missions, for example, can be considered.

4. Scenario

Several optional features are available to the user of the INRMM (weather, presumed driver motivation, operational variations in tire inflation pressure) which allow the user to match the model predictions to features or assumptions of the full operational scenario for which predictions are required. Model instructions which select and control these options are referred to as scenario inputs.

The scenario options include the specification of:

1. Season, which, when seasonal differences in soil strength constitute a part of the terrain data, allows selection of the soil strength according to the variations in soil moisture with seasonal rainfall, and

2. Weather, which affects soil slipperiness and driving visibility, (including dry snow over frozen ground and associated conditions).

3. Several levels of operational influences on driver tolerances to ride vibrations and shock, and on driver strategy in

negotiating vegetation and using brakes.

4. Reasonable play of tire pressure variations to suit the mode of operation--on-road, cross-country, and in sand.

E. Stand-Alone Simulation Modules

As indicated above, the Model is implemented by a series of independent Modules. The Terrain and Vehicle Preprocessors, already described, form two of these. Two further major stand-alone simulation Modules will now be outlined.

1. Obstacle-crossing Module-OBS78B

This Module determines interferences and traction requirements when vehicles are crossing the kind of minor ditches and mounds characterized as part of the areal terrain; it is described fully in Volume II. It is used as a stand-alone Preprocessor Module to the Areal Module of INRMM.

The Obstacle-crossing Module simulates the inclination and position, interferences, and traction requirements of a two-dimensional (vertical center-line plane) vehicle crossing a single obstacle in a trapezoidal shape as a mound or a ditch. The module determines a series of static equilibrium positions of the vehicle as it progresses across the obstacle profile. Extent of interference is determined by comparison of the obstacle profile and the displaced vehicle bottom profile. Traction demand at each position is determined by the forces on driven running gear elements, tangential to the obstacle surface, required to maintain the vehicle's static position. Pitch compliance of suspension elements is not accounted for but frame articulation (as at pitch joints, trailer hitches, etc) is permitted.

The Obstacle-crossing Module produces a table of minimum clearances (or maximum interferences) and average and maximum force required to cross a representative sample of obstacles defined by combinations of obstacle dimensions varied over the ranges appropriate for features included in the areal terrain description. This simulation is done only once for each vehicle. Included in the INRMM Areal Module is a three-dimensional linear interpolation routine which, for any given set of obstacle parameters, approximates from the derived table the corresponding vehicle clearance (or interference) and associated traction requrements. Obviously, the more entries there are in the table, the more precise will be the determination.

2. Ride Dynamics Module- VEHDYN

The Areal Module examines as possible vehicle speed limits in a given terrain situation two limits which are functions of vehicle dynamic perceptions: speed as limited by the driver's tolerance to his vibrational environment when the vehicle is operating over continuously rough ground, and speed as limited by the driver's tolerance to impact received while the vehicle is crossing discrete obstacles. It is assumed that the driver will adjust his speed to ensure that his tolerance levels will not be exceeded.

The Ride Dynamics Module of INRMM, called VEHDYN and described in Volume III, computes accelerations and motions at the driver's station (and other locations, if desired) while the vehicle is operating at a given speed over a specific terrain profile. The profile may be continuously, randomly rough, may consist solely of a single discrete obstacle, uniformly spaced obstacles of a specific height or may be anything in between. From the computed motions, associated with driver modeling and specified tolerance criteria, simple relations are developed for a given vehicle between relevant terrain measurements and maximum tolerable speed. The terrain measurement to which ride speed is related is the root mean square (rms) elevation of the ground profile (with terrain slopes and long-wavelength components removed). The terrain descriptors for obstacles are obstacle height and obstacle spacing.

The terrain parameters involved, rms elevation and obstacle height and spacing, are factors quantified in each patch description, and rms elevation is specified for each road segment. Preprocessing of the vehicle data in the ride dynamics module provides an expedient means of predicting dynamics-based speed in the patch and road segment modules via a simple, rapid table-lookup process.

The currently implemented Ride Dynamics Module is a digital simulation that treats vehicle motions in the vertical center-line plane only (two dimensions). It is a generalized model that will handle any rigid-frame vehicle on tracks and/or tires, with any suspension. Tires are modeled using a segmented wheel representation, (see Lessem (1968)) and a variation of this representation is used to introduce first-order coupling of the road wheels on a tracked vehicle by its tracks.

a) Driver model and tolerance criteria.

It has been shown empirically that, in the continuous roughness situation, driver tolerance is a function of the vibrational power being absorbed by the body. (See Pradko, Lee and Kaluza (1966).) The same work showed that the tolerance limit for representative young American males is approximately 6 watts of continuously absorbed power, and the research resulted in a relatively simple model for power absorption by the body. The body power absorption model, based upon shaping filters applied to the decomposed acceleration spectrum at the driver's station, is an integral part of the INRMM two-dimensional dynamics simulation.

In the past, only the 6 watt criterion was used to determine a given vehicle's speed as limited by rms roughness. More recent measurements in the field have shown that with sufficient motivation young military drivers will tolerate more than 6 watts for periods of many minutes. Accordingly, INRMM will accept as vehicle data a series of ride speed versus rms elevation relations, each corresponding to a different absorbed power level, and will use these to select ride-speed limits according to the operationally related level called for by the scenario. The Ride Dynamics Module will, of course, produce the required additional data, but some increased running time is involved.

The criterion limiting the speed of a vehicle crossing a single discrete obstacle, or a series of closely, regularly spaced obstacles, is a peak acceleration at the driver's seat of 2.5-g passing a 30-Hz. filter. Data relating the 2.5-g speed limit to obstacle height and spacing can be developed in the ride dynamics module by inputting appropriate obstacle profiles.

INRMM requires two obstacle impact relations: the first, speed versus obstacle height for a single obstacle (spacing very great); and the second, speed versus regular obstacle spacing for that single obstacle height (from the single obstacle relation) which limits vehicle speed to a maximum of 15 mph. For obstacles spaced at greater than two vehicle lengths, the single-obstacle speed versus obstacle height relation is used. For closer spacings, the least speed allowable by either relation is selected.

3. Main Computational Modules - NRMM

The highly iterative computations required to predict vehicle performance in each of the many terrain units needed to describe even limited geographic areas are carried out in the three main computational modules. Each of these involve only direct arithmetic algorithms which are rapidly processed in modern computers. In INRMM, even the integrations required to compute acceleration and deceleration between obstacles within an areal patch are expressed in closed, algebraic form.

Terrain input data include a flag, which signifies to the model whether the data describes an areal patch, a linear feature segment, or a road segment. This flag calls up the appropriate computational Module.

a) Areal Terrain Unit Module

This Module calculates the maximum average speed a vehicle could achieve and maintain while crossing an areal terrain unit. The speed is limited by one or a combination of the following factors:

1. Traction available to overcome the combined resistances of soil, slope, obstacles, and vegetation.
2. Driver discomfort in negotiating rough terrain (ride comfort) and his tolerance to vegetation and obstacle impacts.
3. Driver reluctance to proceed faster than the speed at which the vehicle could decelerate to a stop within the, possibly limited, visibility distance prevailing in the areal unit (braking-visibility limit).
4. Maneuvering to avoid trees and/or obstacles.
5. Acceleration and deceleration between obstacles if they are to be overriden.
6. Damage to tires.

Figure I.E.1 shows a general flow chart of how the calculations of the Areal Module are organized.

After determination of some vehicle and terrain - dependent factors used repetitively in the patch computation (1),* the Module is entered with the relation between vehicle steady-state speed and theoretical tractive force and with the minimum soil strength that the vehicle requires to maintain headway on level, weak soils. These data

* Numbers in parentheses correspond to numbers in Figure I.E.1.

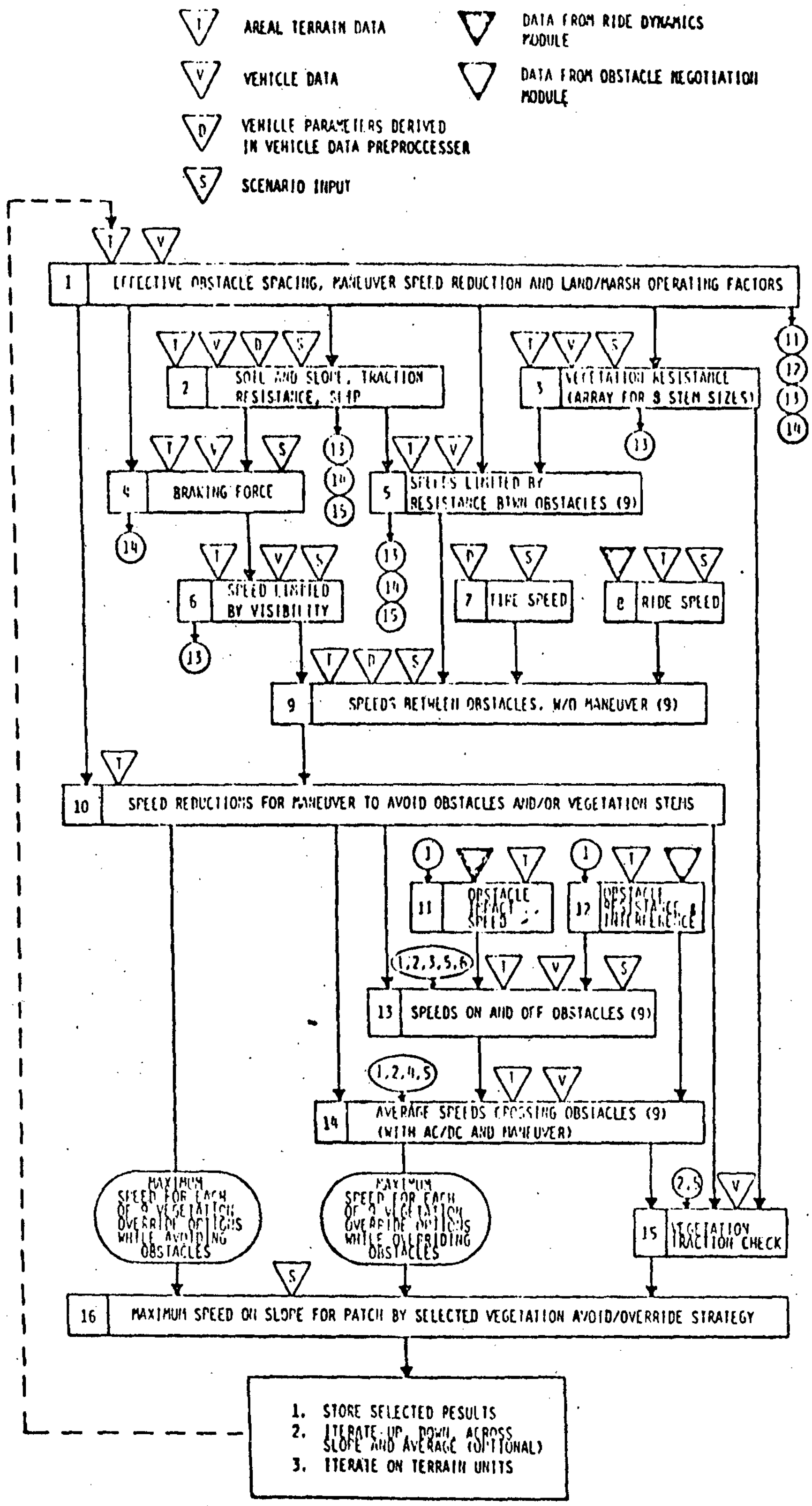


FIGURE I.E.1 -- GENERAL FLOW OF INRMM AREAL MODULE are provided by the vehicle data preprocessor. Soil and slope resistances (2) and braking force limits (4) are computed, and the basic tractive force-speed relation is modified to account for soil-limited traction, soil and slope resistances, and resulting tire or track slip. Forces required to override prevailing tree stems are calculated for eight cases (3): first, overriding only the smallest stems, then overriding the next largest class of stems as well, etc., until in the eighth case all stems are being overridden.

Stem override resistances are combined with the modified tractive force-speed relation to predict nine speeds as limited by basic resistances (5). (The ninth speed corresponds to avoiding all tree stems.)

Maximum braking force and recognition distance are combined to compute a visibility-limited speed (6). Resistance and visibility-limited speeds are compared to the speed limited by tire loading and inflation (7), if applicable, and to the speed limit imposed by driver tolerance to vehicle motions resulting from ground roughness (8). The least of these speeds for each tree override-and-avoid option becomes the maximum speed possible between obstacles by that option, except for degradation due to maneuvering (9).

Obstacle avoidance and/or the tree avoidance implied by limited stem override requires the vehicle to maneuver (or may be impossible).

Using speed reduction factors (derived in 1) associated with avoiding all obstacles (if possible) and avoiding the appropriate classes of tree stems, a series of nine possible speeds (possibly including zero, or NOGO) is computed (10).

A similar set of nine speed predictions is made for the vehicle maneuvering to avoid tree stems only (10). These are further modified by several obstacle crossing considerations.

Possible NOGO interference between the vehicle and the obstacle is checked (12). If obstacle crossing proves to be NOGO, all associated vegetation override and avoid options are also NOGO. If there are no critical interferences, the increase in traction required to negotiate the obstacle is determined (12).

Next, obstacle approach speed and the speed at which the vehicle will depart the obstacle, as a result of the momentarily added resistance encountered, are computed (13). Obstacle approach speed is taken as the lesser of the speed between obstacles, reduced for maneuver required by each stem override and avoid option, and the speed limited by the driver to control his crossing impact (11). Speeds off the obstacle are computed on the basis solely of the soil-and slope-modified tractive force-speed relation (22), i.e. before the tractive force speed relation is modified to account for vegetation override forces, the traction increment required for obstacle negotiation, or any kinetic energy available as a result of the associated obstacle approach speed (13).

Final average speed in the patch for each of the nine tree stem override and avoid options, while the vehicle is overriding patch obstacles, is computed from the speed profile resulting, in general, from considering the vehicle to accelerate from the assigned speed off the obstacle to the allowable speed between obstacles (or to a lesser speed if obstacle spacing is insufficient), to brake to the allowable obstacle approach speed, and to cross the obstacle per se at the computed crossing speed.

Following a final check to ensure that traction and kinetic energy are sufficient for single-tree overrides required (and possible resetting of speeds for some options to NOGO) a single maximum in-patch speed (for the direction of travel being considered relative to the in-unit slope) is selected from among the nine available values associated with obstacle avoidance and the nine for the obstacle override cases. If all 18 options are NOGO, the patch is NOGO for the direction of travel. If several speeds are given, selection is made by one of two logics according to scenario input instructions.

In the past the driver was assumed to be both omniscient and somewhat mad. Accordingly, the maximum speed possible by any of the 18 strategies was selected as the final speed prediction for the terrain unit (and slope direction). Field tests have shown, however, that a driver does not often behave in this ideal manner when driving among trees. Rather, he will take heroic measures to reach some reasonable minimum speed, but will not continue such efforts when those measures involve knocking down trees that he judges it imprudent to attack, even though by doing so he could go still faster. In INRMM, either assignment of maximum speed may be made: the absolute maximum which addresses the vehicle's ultimate potential, or a lesser value which in effect more precisely models actual driver behavior.

If the scenario data specify a traverse prediction, the in-unit speed and other predictions are complete at this point, and the model stores those results specified by the user and goes on to consider the next terrain unit (or next vehicle, condition, etc). When a full areal prediction is called for, the entire computation is repeated three times: once for the vehicle operating up the in-unit slope, once across the slope, and once down the slope. Desired data are stored from each such run prior to the next, and at the conclusion of the third run, the three speeds are averaged. Averaging is done on the assumption that one-third of the distance* will be travelled in each direction, resulting in an omnidirectional mean.

* the average speed, V_{av}, is the harmonic average of the three speeds,i.e.

$$V_{av} = 3/[(1/V_{up}) + (1/V_{across}) + (1/V_{down})]$$

b) Road Module

The Road Module calculates the maximum average speed a vehicle can be expected to attain traveling along a nominally uniform stretch of road, termed a road unit. Travel on super highways, primary and secondary roads, and trails is distinguished by specifying a road type and a surface condition factor. From these characteristics, values of tractive and rolling resistance coefficients for wheeled and tracked vehicles on hard surfaced roads are determined by a table look-up. For trails, surface condition is specified in terms of cone index (CI) or rating cone index (RCI). Traction, motion resistance, and slip are computed using the soil submodel of the Areal Module, with scenario weather factors used in the same way as in making off-road predictions.

The relations used for computing vehicle performance on smooth, hard pavements are taken from the literature (Smith (1970) and Taborek (1957)).

The structure of the Road Module, while much simpler, parallels that of the Areal Module. Separate speeds are computed as limited by available traction and countervailing resistances (rolling, aerodynamic, grade, and curvature), by ride dynamics (absorbed power), by visibility and braking, by tire load, inflation and construction, and by road curvature per se (a feature not directly considered in the Areal Module). The least of these five speeds is assigned as the maximum for the road unit (for the assumed direction relative to the specified grade).

The basic curvature speed limits are derived from American Association of State Highway Officials (AASHO) experience data for the four classes of roads (AASHO (1975)) under dry conditions and are not vehicle dependent. These are appropriately reduced for reduced traction conditions, and vehicle dependent checks are made for tipping or sliding while the vehicle is in the curve.

At the end of a computation, data required by the user are stored. If the model is run in the traverse mode, the model returns to compute values for the next unit; if in the areal mode, it automatically computes performance for both the up-grade and down-grade situations and at the conclusion computes the bidirectional (harmonic) average speed. Scenario options are similar to those for the Areal Module.

F. Acknowledgments

As with any comprehensive compendium covering knowledge in a particular subject area, the results are due to the combined effort of all workers in the discipline. The authors, in this case, are somewhat akin to the scribes of ancient days, recording and organizing the wisdom and folly of those around them.

There are those, however, whose contributions stand out as related to the creation of the Mobility Model itself. The authors w.sh to acknowledge these people explicitly.

Clifford J. Nuttall, Jr., currently with the Mobility Systems Division, Geotechnical Laboratory at the U. S. Army Engineer Waterway Experiment Station (WES) provided the inspiration for many of the submodels, guided the evolution of the content of the entire model, and provided the wisdom and judgement which hopefully kept the various portions in proportion with each other. Additional experience in use of this and predecessor models came from many studies conducted by Donald Randolph at WES. During the model development period, general direction and supervision at WES came from W. G. Schockley, A. A. Rula, E. S. Rush and J. L. Smith.

Peter Haley, from the Tank Automotive Concepts Laboratory, USA TARADCOM, and also the manager of the NATO Reference Mobility Model, in addition to providing overall guidance and judgment did much of the seemingly endless detailed design and testing of the algorithms and code. He was aided in the coding by Thomas Washburn. Direct supervision of the model development at TARADCOM came from Zoltan J. Janosi, who also now serves as Chairman of the Technical Management Committee of the NATO Reference Mobility Model. General supervision during the project was provided by J. G. Parks, O. Renius, and Lt. Col. T. H. Huber. Dr. E. N. Petrick, Chief Scientist of USA TARADCOM, the moving force of the NATO RSI effort in the U. S. Army vehicle community, provided overall guidance and support for this activity. He has been aided in this by Edward Lowe, NATO Standardization and Metrication Officer at TARADCOM.

Newell Murphy of the Mobility Systems Division, WES, provided the driving force behind the current version of the Ride Dynamics Module, supervising its conception, creation, and testing as well as guiding the field work supporting it. Richard Ahlvin of WES and Jeff Wilson of Mississippi State University bore primary responsibility for the production of the sequence of computer programs which have implemented this Module.

The authors also wish to acknowledge the contributions of their colleagues at Stevens Institute of Technology. Jan Nazalewicz was responsible for much of the Obstacle Module. Supervision and guidance during the project came from I. Robert Ehrlich and Irmin O. Kamm.

The arduous task of entering and formatting the text of this report was performed by M. Raihan Ali and Gabriel Totino. Graphics and charts were prepared by Mary Ann McGuire and Christopher McLaughlin. The authors benefited from a careful review of the first draft by Peter Haley. Finally each of the authors notes than any errors are the fault of the other author.

II ALGORITHMS AND EQUATIONS

The INRMM has been implemented in a computer program called NRMM, written in FORTRAN Extended, version 4.6, for the CDC 6600 computer. The description of the Operational Modules which follows occasionally refers to particular aspects of this implementation.

A. Control and I/O Module

The Control and I/O Module (C&I/O) of NRMM consists of a main program and several subroutines which control the flow between vehicle and terrain input and the two operational modules for patches (Areal terrain units) and roads. It is also responsible for output. An overall illustration of the Control and I/O Module is given in Figure II.A.1.

After initialization and setting of variables to their default values, the program opens the files required. It then calls subroutine SCN to read the control variables, which determine how the program is to operate, and the scenario variables, which determine conditions of the simulation.

The program then calls subroutine VEH, which reads the vehicle data, and subroutine VPP, the Vehicle Preprocessor, which calculates vehicle descriptors derived from the vehicle input data, some of which depend on values of scenario variables. Details of these calculations are given in the next section, Section II.B.

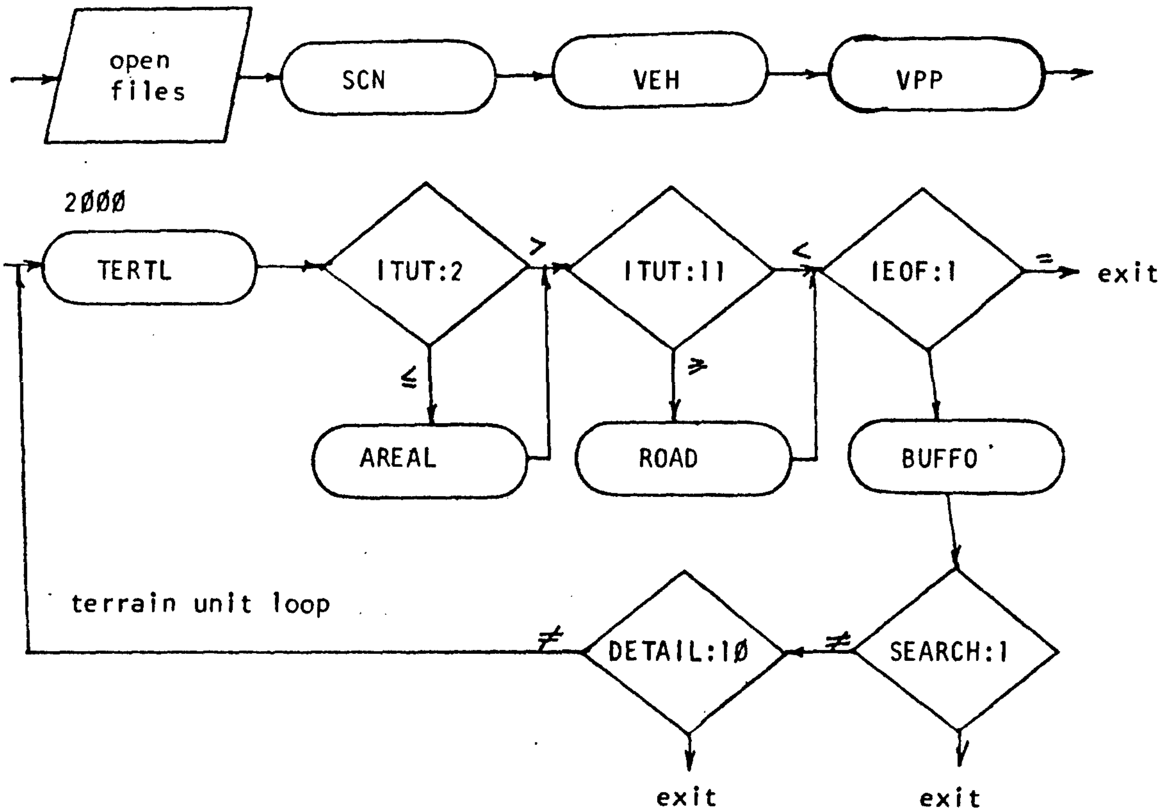


FIGURE II.A.1 -- Structure of Control and I/O Module

The program then enters the terrain loop; that is, it reads the descriptors of the terrain unit under control of subroutine TERTL and its subroutines MAP71, MAP74, MPRD74, calculates several terrain descriptors derived from the primary terrain input data, some of which also depend on values of scenario variables. The program then selects the appropriate operational module, AREAL or ROAD, and calculates the speed the vehicle could be expected to go on that terrain unit. The program returns to TERTL and repeats these calculations if there is more terrain data. If not, the program exits.

There are two options that restrict the loop to a single terrain unit other than if only one terrain unit is present in the terrain input file. Setting control variable SEARCH to 1 indicates that a single terrain unit, whose terrain unit number is given by NTUX, is to be sought in the terrain input file and that the mobility model calculations are to be performed for that one unit only. Control variable DETAIL indicates to what level of detail the output is to be written. The following actions are taken for various values of DETAIL.

1. DETAIL = 1: only the output from BUFFO is written - this consists of terrain unit identification, grade, and maximum and selected speeds only.
2. DETAIL = 2,3 or 4: above output is written. Also control, scenario and vehicle input is echoed, and output from the vehicle Preprocessor is written.
3. DETAIL = 5 : results from almost all intermediate calculations are written. A printer-plot of the tractive effort vs. speed relationship is generated and program execution is terminated after the vehicle preprocessor.
4. DETAIL = 10: Results from almost all intermediate calculations are written.

Setting DETAIL to 10 results in a large volume of output. A check is made in the C&I/O Module and if DETAIL = 10 an exit is made after execution on one terrain unit.

The individual subroutines of the C&I/O Module will now be described briefly.

1. Subroutine SCN - Scenario and Control Input

This subroutine sets default values for the scenario variables (see Section III.D) and then reads the control variables. These consist of flags controlling writing of output for the entire program and individual routines, and the single terrain unit search described above. Then the scenario variables are read, echoed if the appropriate flag (KSCEN) was set to 1, and converted to standard units as necessary.

2. Subroutine VEH - Vehicle Input

This subroutine reads the vehicle parameters as described in Section III.B.1 below and echoes them if the flag KVEH is set to 1.

3. Subroutine VPP - Vehicle Preprocessor Control

This is the Vehicle Preprocessor control program. It consists solely of a series of calls to Subroutines II1, II2,...,II17 and output statement executions as the flags KII1, KII2,...,KII17 are set. If DETAIL = 5, a printer plot describing the vehicle power train is written using Subroutine PLTSET.

4. Subroutine TERTL - Terrain Translator

This subroutine controls the terrain data inputs. Since terrain files are often large and the results of long and expensive effort,

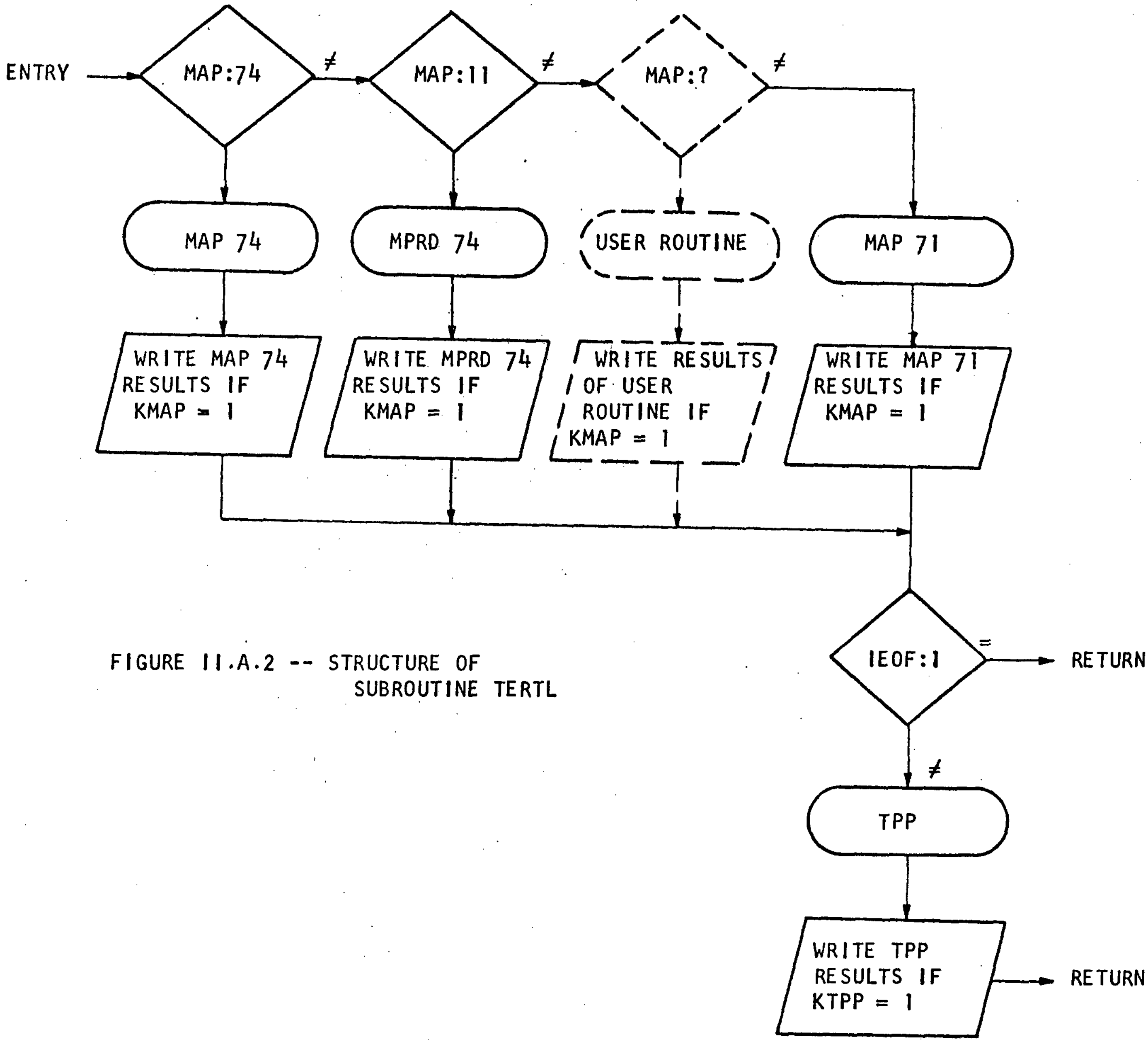


FIGURE II.A.2 -- STRUCTURE OF SUBROUTINE TERTL

users will be reluctant to re-format their existing data to use the NRMM. Instead, the user may write a FORTRAN subroutine which reads the existing files and adjusts the data to satisfy the terrain data requirements of NRMM. Calls to these subroutines are controlled by a Control/Scenario variable called MAP. Three terrain data input routines are currently part of NRMM:

i) Subroutine MAP71, called if MAP is not equal to one of the values below,

ii) Subroutine MAP74, called if MAP = 74,

iii) Subroutine MPRD74, called if MAP = 11.

If the user writes a terrain input routine, called MXXXXX, for an existing data file, a unique, new value of MAP must be assigned and an appropriate IF, CALL and echo write (if desired) need to be added to Section 2., ALGORITHM, in Subroutine TERTL. Care must be taken that this new subroutine, MXXXXX, provides values for the complete list of terrain variables required by NRMM.

Each of the terrain data input subroutines must contain a check for end of terrain data. The existing routines use the CDC run-time FORTRAN function subprogram called EOF which returns a value of 1 if the READ tried to read a record but found an end-of-file instead. In this case the terrain input subroutines set the flag IEOF = 1 which is passed through TERTL to the C&I/O main program; whereupon the NRMM run will terminate.

If another terrain unit was read, TERTL will call Subroutine TPP, the Terrain Preprocessor, described in Section II.C. below, and will write the results if the flag KTPP is set. TERTL then returns.

a) Terrain Input/Translation Subroutine MAP74

This subroutine reads a record for each terrain unit consisting of actual terrain descriptors. These include various values for cone index, grade, obstacle geometry and spacing, surface roughness, spacing of vegetation in eight stem diameter classes, and various values of recognition distance. The particular value of cone index and recognition distance chosen depend on the scenario variables ISEASN and MONTH, respectively. The format of the data to be read by this subroutine is described in Section III.C.2.

The input record also contains NTU, the terrain unit number. If SEARCH = 1 (that is, a particular terrain unit is sought), MAP74 will continue to read terrain unit records, discarding those for which NTU differs from NTUX. Only when a record is read for which NTU = NTUX will MAP74 return to TERTL.

b) Terrain Input/Translation Subroutine MPRD74

This subroutine was designed to read records describing roadway units, including trails compiled for a particular study. For this study the speed limits imposed by horizontal curvature were included as terrain data. Since the Road Module required road curvature as the descriptor, this routine translates the curvature speed limit back to curvature.

The routine first establishes a table relating curvature to maximum speed for four classes of roads: superhighways, primary roads, secondary roads, and trails. Then MPRD74 reads a record of actual values describing the roadway, including cone indexes, grade, recognition distances, surface roughness, curvature speed limit, coefficient of friction, superelevation, and a surface condition factor. The curvature vs. speed table is then interpolated to set a curvature based on the curvature speed of the actual roadway unit. The format of the data to be read by this subroutine is described in Section III.C.3.

The input record also contains NTU, the terrain unit number. If SEARCH = 1 (that is, a particular roadway unit is sought), MPRD74 will continue to read roadway unit records, discarding those for which NTU differs from NTUX. Only when a record for which NTU = NTUX is read will MPRD74 process the data and then return to TERTL.

c) Terrain Input/Translation Subroutine MAP71

This subroutine was designed to read terrain data as class interval designators and to translate these designators into actual terrain descriptors. This is the format of the terrain data files read by an earlier mobility model, AMC71.

The entire range of possible values for each of the terrain descriptors is divided into a sequence of intervals from which a single number, the interval representative, is used for all terrain units whose actual value of that descriptor falls within that interval. This interval is given an integer as its designator. The terrain files read by MAP71 consist of records, one for each terrain unit, containing one or two digit integers designating those intervals into which the terrain descriptors fall.

The subroutine first establishes tables, one for each terrain descriptor, of representatives for each designator. This is done by DATA statements. Then records are read, tests are made for end of input file (and NTUX if SEARCH = 1) and the class interval representatives are loaded into the terrain descriptors by simple table lookup procedures. The format of the data to be read by this subroutine is described in Section III.C.1.

A test is made to check the compatability of obstacle parameters and flags are set for those stem diameter classes not present in the terrain unit. The routine then returns to TERTL.

5. Subroutine AREAL - Areal Module Control

This subroutine is the control program for the Areal Module. It consists solely of a sequence of CALL's to subroutines named IV1, IV2,...,IV21, each call followed by a test of the corresponding flag KIV1, KIV2,...,KIV21 to determine if the results of that subroutine are to be written. The Areal Module is described in section II.D below. The subroutine then returns to the C&I/O main program.

6. Subroutine ROAD - Road Module

Since the Road Module is considerably smaller than the Areal Module no separate control program was written. This subroutine is the entire Road Module and described in Section II.E below.

7. Subroutine BUFFO - Basic Output

This is the basic output subroutine of the C&I/O Module. The particular outputs are described in Section III.E.

B. Vehicle Preprocessor

The Vehicle Preprocessor Module consists of a sequence of subroutines named II1, II2,...,II17, and some additional subroutines named TRAIN, AUTOM, STICK, LINEAR, FIT, APPROX, SOLVER, LINES, RESIDU, PLTSET, SCAL, LIMITS, FIXER and CURPT called by II16. These routines adjust dimensions and calculate derived vehicle descriptors, including the tractive effort vs. speed relationship.

The tractive effort vs. speed relationship, at this stage (i.e., without attenuation for soil limits or slip) also known as the rim pull curve of the vehicle, is fitted by a sequence of quadratic curves for various ranges of speeds. If the program encounters difficulties with the curve fit procedure it will print/plot the points and the curves for user analysis and intervention. Alternately the user may wish to have the points and curves plotted in any case by setting the switch DETAIL to 5.

The individual subroutines comprising the Vehicle Preprocessor will now be described.

1. Subroutine II1 - Units Conversion Routine

This routine changes those vehicle parameters that are not entered in the units of lbs, inches, radians and/or seconds into these units. One exception is the engine revolutions per minute which are converted to revolutions per second, not radians per second.

Those users describing their vehicles in SI units will be required to modify this routine extensively or perhaps write a separate program to change the vehicle data into the U.S. Customary units used in NRMM.

2. Subroutine II2 - Gross Combined Weight

The weight on each suspension assembly is given as part of the vehicle input. In addition, the flag IP(i) = 1 indicates that assembly i is powered and the flag IB(i) = 1 indicates that assembly i is braked. This subroutine sums the weights on all the assemblies into GCW and the weight on the powered and braked assemblies into GCWP and GCWB, respectively. The weight on the non-powered, GCWNP, and non-braked, GCWNB, assemblies is also calculated.

3. Subroutine II3 - Maximum Tire Speed

This subroutine calculates a maximum speed which a wheeled vehicle could travel without destroying the tire. This speed, VTIRE(j), calculated

for j = 1 fine grained soil

j = 2 coarse grained soil

j = 3 highway

depends primarily on tire size, construction, and inflation pressure.*
The formulas are, for each assembly i,

$$s = (b_{ti} - .4b_{ri})/.75$$

* For further explanation on how inflation pressure is used in this Model see Section II.D.3.

$$h = (4.32/s^{2.38})[(W_i/n_i)(1/(d_{ri}+s))]^{1.71}$$

where b_{ti} = section width of tires

b_{ri} = width of rims

w_i = load on entire assembly

n_i = number of tires

d_{ri} = diameter of rims.

For radial tires (ICONST(i)$\neq$ 1)

$$V_{tij} = 100.(p_{ij}/h)^2 \quad \text{in miles per hour}$$

and for bias ply tires (ICONST(i) = 1)

$$V_{tij} = 70.(p_{ij}/h)^{2.25} \quad \text{in miles per hour}$$

where p_{ij} = pressure used in tires on assembly i for

j = 1 fine grained soil

j = 2 coarse grained soil

j = 3 highway

V_{tij} = maximum safe speed for tires on assembly i at pressure j.

V_{tij} is converted to inches per second in II3. The maximum safe tire speed for the vehicle is then

$$VTIRE(j) = \min \{V_{tij} \text{ for all } i\}$$

for tire pressure j.

These relationships are patterned after Eklund (1945) with modifications.

4. Subroutine II4 - Maximum Path Width

In this subroutine the maximum path width of the suspension assemblies is found by subtracting the clearance between the left and the right suspension assembly elements from the tread width and finding the largest such number.

5. Subroutine II5 - Tire Deflection Ratio

For each wheeled assembly i the variable δ_{ij}, or DFLCT(i,j), gives the deflection of the tire at the pressure used for j = 1 fine grained soil, j = 2 coarse grained soil, and j = 3 highway. This routine calculates the deflection ratio, DRAT(i,j), as the ratio of the deflection and the section height, h_i.

6. Subroutine II6 - Characteristic Length

In this routine the characteristic length, l_{ij} or CHARLN(i,j), of a suspension assembly i is set to the track length TRAKLN(i) of assembly i if tracked or

$$l_{ij} = 2(\delta_{ij} d_{ti} - \delta_{ij}^2)^{1/2} \quad \text{if wheeled}$$

where δ_{ij} = DFLCT(i,j)

d_{ti} = diameter of tire on assembly i

7. Subroutine II7 - Ground Contact Area

In this routine the ground contact area, GCA(i,j), for the elements on suspension i is set to

2 * characteristic length * track width   if tracked

or

characteristic length * section width   if wheeled

Since the characteristic length depends on tire pressure j, so does the ground contact area.

8. Subroutine II8 - Controlling Lateral Distance

In this routine, the minimum lateral distance, WTMAX, from the center of gravity to the supporting element of each suspension assembly is found. This represents the maximum lateral support base. Thus for wheeled vehicles

$$WTMAX = \min \{t_i/2 - y_{CG} + (b_i/2)*ID_i \quad \text{for all } i\}$$

where t_i = tread width of suspension i

y_{CG} = lateral distance of center of gravity from vehicle center

b_i = section width of tires on suspension I

ID_i = 0 if singles

1 if duals

and for tracked vehicles

$$WTMAX = \min \{t_i/2 - y_{CG} \quad \text{for all } i\}$$

9. Subroutine II9 - Maximum Rolling Radius

The rolling radius of the tire is calculated from the revolutions per mile, $REVM_i$, which is an input parameter, as

$$RR = \max \quad (12*5280)/(2 *REVM_i)$$

10. Subroutine II10 - Maximum Braking Force

This subroutine calculates the maximum braking force the vehicle can support by summing the product of the braking coefficient, XBRCOF, entered as part of the vehicle data, and the weight on each suspension element for those suspension elements which are allowed to be braked [IB(i) = 1].

This force is to represent the vehicle's ability to arrest its running gear regardless of the running gear ground surface traction coefficient.

11. Subroutine II11 - Horsepower/ton

Here the net horsepower, HPNET, entered as part of the vehicle data is divided by the weight of the vehicle supported on the powered traction elements, CGWP, converted to tons.

12. Subroutine II12 - Vehicle Cone Index in Fine Grained Soil

This routine calculates the single pass Vehicle Cone Index for fine grained soil (VCIFG) for each suspension assembly by applying the equations for all-wheeled and tracked vehicles to a single axle or a pair of tracked elements. For wheeled axles, a separate VCIFG is calculated for the three tire pressures, possibly different, recommended for fine grained soil, coarse grained soil, or roads. For wheeled axles, the following calculations are made:

Contact Pressure Factor: $CPFFG = W_i/(n_i b_i d_i/2)$

where for each axle i

W_i = weight on axle

n_i = number of tires

b_i = section width

d_i = outside diameter of tire

Weight Factor: $WF = .553\ W_i/1000.$ if $W_i \le 2{,}000$ lbs

$= .033\ W_i/1000. + 1.$ if $2{,}000 < W_i \le 13{,}500$

$= .142\ W_i/1000. - .42$ if $13{,}500 < W_i \le 20{,}000$

$= .278\ W_i/1000. - 3.115$ if $20{,}000 < W_i$

Tire Factor: $TF = (10 + b_i)/100.$

Grouser Factor: GF = 1.00 without chains ($ICHAIN_i = 0$)

= 1.05 with chains ($ICHAIN_i = 1$)

Wheel Load Factor: $WLORF = (W_i/1000)/(n_i/2)$

Clearance Factor: $CLF = CLRMIN_i/10$

Engine Factor: EF = 1.00 if less than 10 hp/ton

= 1.05 if 10 hp/ton or more

Transmission Factor: TFX = 1.00 for automatic (ITVAR = 0)

$= 1.05$ for manual (ITVAR $\neq$ 0)

Tire Deflection Factor: $TDF_{ij} = [((1 - \delta_{ij})/b_i)/.85]^{1.5}$

where δ_{ij} = deflection of the tire on axle i when inflated with pressure recommended for

$j = 1$	fine grained soil
$j = 2$	coarse grained soil
$j = 3$	roadway

Mobility Index: XMI = [(CPFFG*WF)/(TF*GF) + WLORF - CLF]*EF*TFX

Vehicle Cone Index (Wheeled, Fine Grained Soil)

$$VCIFG_{ij} = [11.48 + .2\ XMI - 39.2/(XMI + 3.74)]TDF_{ij}$$

For a left- right pair of tracked suspension elements, the following calculations are made:

Contact Pressure Factor: $CPFFG = W_i/(2l_i b_i)$

where for each left-right pair of tracked elements i

W_i = weight supported by pair

l_i = length of tracked element in contact with ground

b_i = width of tracked element

Weight Factor:

WF = 1.0	if	$W_i < 50{,}000$
WF = 1.2	if	$50{,}000 \leq W_i < 70{,}000$
WF = 1.4	if	$70{,}000 \leq W_i < 100{,}000$
WF = 1.8	if	$100{,}000 \leq W_i$

Track Factor: $TF = b_i/100$

Grouser Factor:

GF = 1.0	if grouser height less than 1.5 in.
GF = 1.1	if grouser height is 1.5 in. or more

Bogie Factor: $WLORF = W_i/10/N_i/A_{si}$

where N_i = total number of road wheels on tracks in contact with ground

A_{si} = area of one track shoe (in^2)

Clearance Factor: $CLF = CLRMIN_i/10$

Engine Factor: $EF = 1.00$ if 10 hp/ton or more on element i

$= 1.05$ if less than 10 hp/ton on element i

Transmission Factor: $TFX = 1.0$ if automatic ($ITVAR \neq 0$)

$TFX = 1.05$ if manual ($ITVAR = 0$)

Mobility Factor: XMI = [(CPFFG*WF)/(TF*GF) + WLORF - CLF]*EF*TFX

Vehicle Cone Index (Tracked, Fine Grained Soil)

VCIFG = 7 + .2 XMI - 39.2/(XMI + 5.6)

13. Subroutine II13 - Vehicle Cone Index in Coarse Grained Soil

This routine calculates the single pass Vehicle Cone Index for coarse grained soil (VCICG) for each suspension assembly by applying the equations for an all-wheeled and tracked vehicle to each single axle or each pair of left-right tracked elements. For wheeled axles, a separate VCICG is calculated for the three tire pressures, possibly different, recommended for fine grained soil, course grained soil, and roads.

For wheeled axles, the following calculations are made:

Wheel Diameter Factor: $WDF = 2.$ if $b_i/d_{ri} \geq 2.4$

$WDF = 5.$ if $b_i/d_{ri} < 2.4$

where b_i = nominal tire width

d_{ri} = rim diameter

Contact Pressure Factor:

$$CPFCG = .607p_{ij} + 1.35[(117.*\text{ply rating})/(WDF*b_i + d_{ri})] - 4.93$$

where p_{ij} = pressure of tire on assembly i recommended

for j = 1 fine grained soil

j = 2 coarse grained soil

j = 3 roadways

Contact Area Factor: $CAF = \log_{10}(W_i/CPFCG)$

where W_i = weight on axle i

Strength Factor:

$$STF = .0526(n_i + .0211p_{ij}) - .35CAF + 1.587$$

where n_i = number of tires on axle i

Vehicle Cone Index is then 10 raised to the power STF:

$$VCICG_{ij} = 10^{STF}$$

For tracked assemblies, the VCICG is set equal to zero since it is not used in further calculations.

14. Subroutine II14 - Vehicle Cone Index for Muskeg

This routine calculates the single pass Vehicle Cone Index for muskeg (VCIMUK) as follows:

$$VCIMUK = 13 + .535\ W_i/(b_i + d_i * n_i)$$ for wheeled axles

where W_i = weight on axle i

b_i = section width of tires on axle i

d_i = outside diameter of tires on axle i

n_i = number of tires on axle i

$$VCIMUK = 13 + .0625\ W_i(b_i + l_i)$$ for tracked assemblies

where b_i = track width

l_i = track length on ground

The use of these relationships to model vehicle performance on muskeg is included primarily for comleteness. It is a simplistic model based on data of a single study, Schreiner(1967).

15. Subroutine II15 - Combined Contact Pressure Factor

This subroutine finds the maximum contact pressure factor across suspension assembly. For wheeled assemblies a separate maximum is sought for each pressure setting, j = 1 fine grained soil, j = 2 coarse grained soil, and j = 3 roadway.

16. Subroutine II16 - Power Train

This subroutine controls the calculations used to specify the power train of the simulated vehicle.

The driving, as opposed to braking, characteristics of the vehicle are modeled by a tractive effort vs. speed of vehicle relationship. This relationship is given by a series of quadratics

$$F(v) = c_n v^2 + b_n v + a_n$$

where different values for the constants c_n, b_n, a_n are used for different "gears", n. For computational purposes the gears are really speed ranges. Thus, if $v_o = 0 \leq v_1 \leq v_2 \leq . \ . \ . \leq v_n$ is a non-decreasing sequence of speeds that represent the "gear" intervals, the tractive effort relationship given by the above formula applies for $v_{n-1} \leq v \leq v_n$.

The precise manner of selecting the sequence $\{v_n: n=0,1,...,NG\}$ and the number of gears, NG, consists of several discrete steps. The logic of this calculation is shown in Figure II.B.1.

The first step depends on whether the tractive effort vs. speed was entered as part of the vehicle data. If so, Subroutine FIT is entered immediately to calculate NG, the v_n's, and the coefficients a_n, b_n, and c_n.

If the table of tractive effort vs. speed is to be constructed, it is constructed from basic power train descriptors such as engine torque at given RPM, the torque converter characteristics, the transmission and differential and/or transfer case ratios, and the radius of the drive sprocket or wheels. These calculations are controlled and performed by Subroutine TRAIN and the Subroutines AUTOM and STICK.

a) Subroutine TRAIN - Construction of Tractive Effort vs. Speed Curves

This routine controls the calculations to construct a table of tractive effort vs vehicle speed values. Since no slip or surface characteristics are used, this may be called the "rim pull curve". This routine first loads the speed array, POWER(SPEED,N), with values of forward speed from zero to 100 MPH in half mile-per-hour increments. The variable SPEED is declared integer and given a value 1

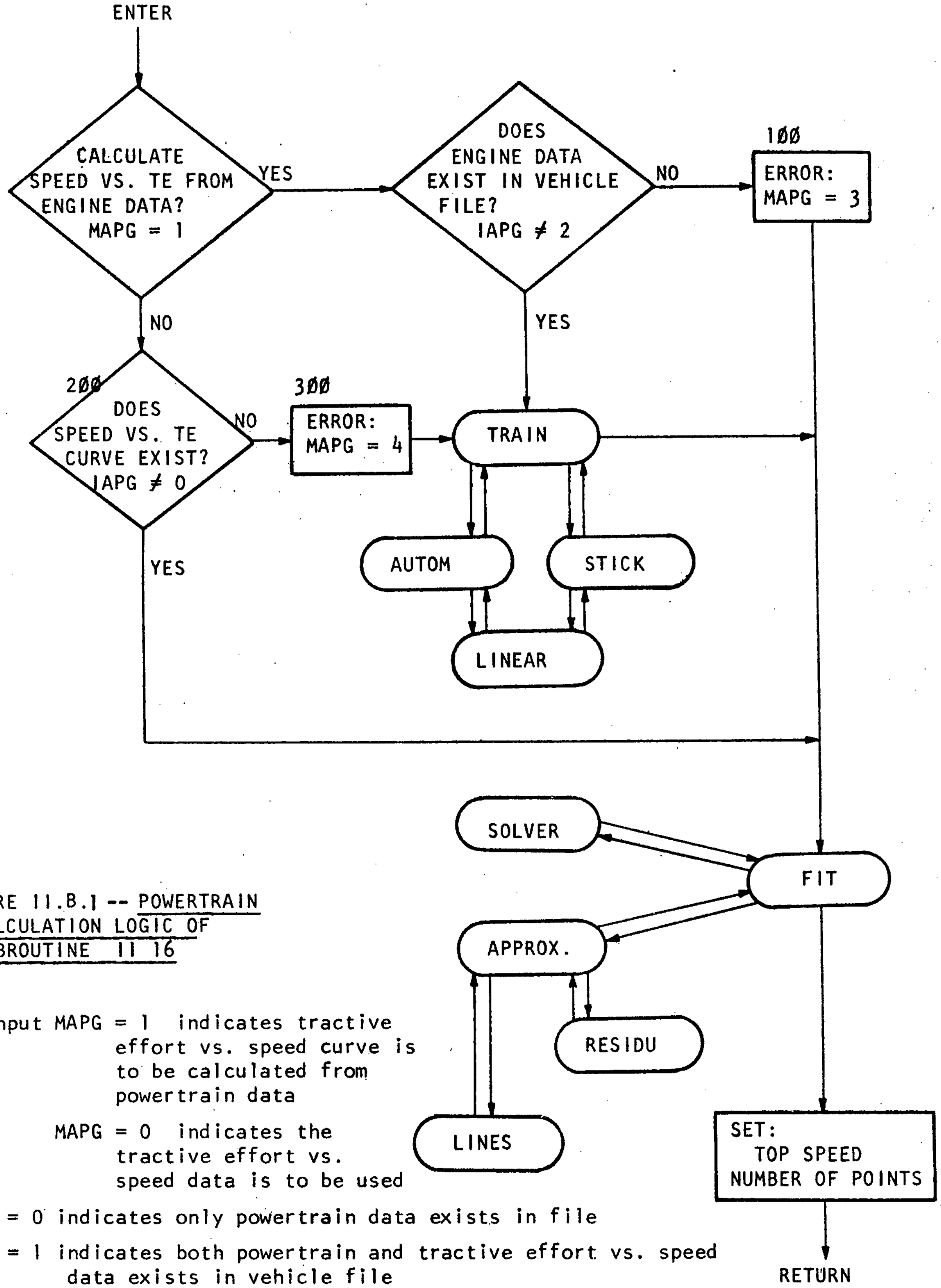


FIGURE II.B.1 -- POWERTRAIN CALCULATION LOGIC OF SUBROUTINE II 16

On input MAPG = 1 indicates tractive effort vs. speed curve is to be calculated from powertrain data

MAPG = 0 indicates the tractive effort vs. speed data is to be used

IAPG = 0 indicates only powertrain data exists in file

IAPG = 1 indicates both powertrain and tractive effort vs. speed data exists in vehicle file

IAPG = 2 indicates only tractive effort vs. speed data exists in vehicle file and used for clarity. POWER(FORCE,N) is initialized at zero. Here the variable FORCE is also declared integer and given the value 2 and used for clarity.

The engine torque vs. engine speed relationship is stored in the array ENGINE where the speed values are located in ENGINE(RPM,N) and the torque in ENGINE(TORQUE,N). Here RPM =1 and TORQUE = 2, both declared integer. If the vehicle is fitted with an engine-to-transmission transfer gear box, this relationship is modified to represent the torque vs. speed at the output shaft of this gear box.

Subroutine TRAIN then calls AUTOM for simulation of an automatic transmission or STICK for simulation of a manual transmission. Upon return, diagnostic output is written if called for.

(1) Subroutine AUTOM - Tractive Effort vs. Speed of Vehicle with Automatic Transmission and Torque Converter

The following calculations are performed for each transmission gear ratio and vehicle speed:

The routine first converts the vehicle speed to torque converter output speed by dividing by 2pi times the wheel/sprocket radius and multiplying by the final drive and transmission gear ratios. A trial value of engine RPM is then chosen and the resulting torque converter speed ratio is calculated. From the input data, a torque converter input speed is estimated (by linear interpolation of the torque converter input speed vs. speed ratio data). The square of the ratio of this torque converter input speed to the engine output speed [which physically must be one but may not be due to the trial value of engine speed not being physically realizable] is then multiplied by the input torque at which the torque converter relationships apply to yield a torque converter input torque. From the engine data, an engine output torque is estimated by linearly interpolating the engine speed vs. torque relationship (also input data). The mismatches between engine output torque and torque converter input torque and engine output speed and torque converter input speed (physically both of which must be the same) are used to adjust the estimated engine speed higher or lower. This adjustment is performed by following a binary iteration scheme.

Once an engine speed at which both the engine output torque matches the estimated torque converter input torque and the engine output speed matches the torque converter input speed has been determined, the torque converter torque ratio at the specified speed ratio is used to calculate a torque converter output torque which when multiplied by gear ratios and efficiencies of the transmission and final drive and divided by the moment arm of the driving wheel/sprocket yields a driving force (tractive effort).

For each gear, the above calculations are done for every speed in the vehicle speed array. For each speed, the maximum tractive effort among those for various gears is chosen.

(2) Subroutine STICK - Tractive Effort vs. Speed of Vehicle with Manual Transmission

For each speed and each gear, the vehicle speed is transformed into an engine speed by dividing by the circumference of the driving wheel/sprocket and multiplying by the final drive and transmission gear ratios. An engine torque is then estimated by linearly interpolating the engine speed vs. torque relationship. This torque is then transformed into a driving force (tractive effort) by multiplying by the transmission and final drive gear ratios and efficiencies and dividing by the moment arm of the driving wheel/sprocket.

For each speed, the maximum tractive effort among those for the various gears is chosen.

b) Subroutine FIT - Quadratic Curve Fit to Tractive Effort vs. Speed Relationship

The Tractive Effort vs. Speed relationship above may be visualized as a sequence of points on a plot of those two variables. This routine determines the coefficients of a sequence of quadratics $F = a + bv + cv^2$ and the minimum and maximum speeds for which each of these quadratics fit the relationship plotted above. Each speed range for which a different set of coefficients (a,b,c) must be used

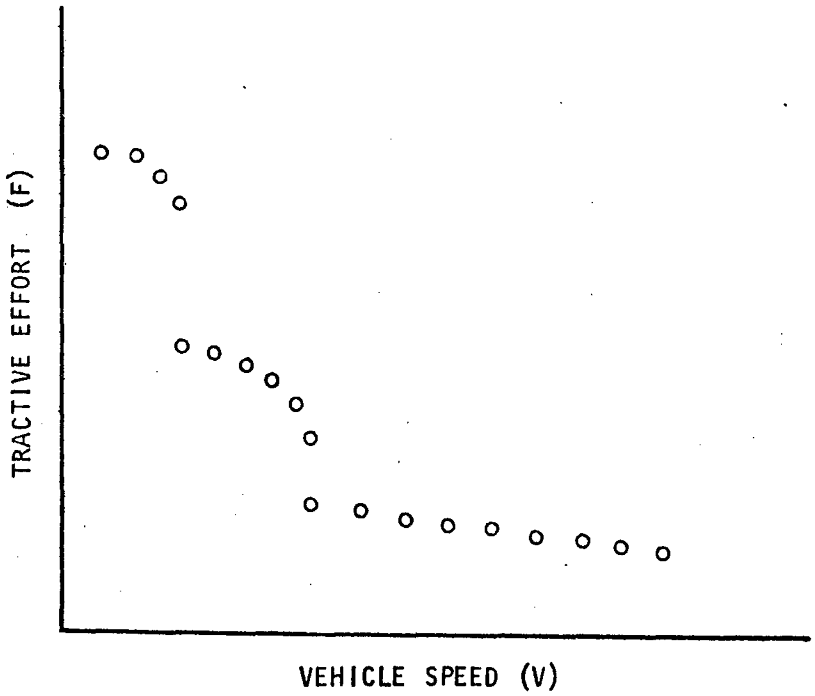


FIGURE II.B.2 -- Plot of Tractive Effort vs. Speed Array

will be called a "gear" since for some vehicle transmissions the speed range may actually correspond to the speeds for which a particular transmision gear is used.

From the plot it is immediately apparent for what speed ranges each gear should be defined. This is due to the powerful pattern recognition capability of the human eye and brain. An efficient computer algorithm which has the same capability for a sequence of number pairs stored in computer memory is difficult to develop. Although the algorithm of this subroutine is capable of successfully distinguishing maximum and minimum speeds for each gear for large class of engine/transmission combinations there are still occasional tractive effort vs. force relationships for which the quadratic fit procedure will not be satisfactory. In these cases the program will terminate and print-plot the relationship for human intervention.

The procedure used here is, starting with three points, to sequentially fit quadratics using a least squares criterion and to test if the next point falls within a range of 2% of the tractive effort predicted by extrapolation of the fitted curve. If it does, this (next) point is included in the current gear and the procedure is repeated for the following point. If it does not, a new gear is started.

When all the points "belonging" to gear n have been found the coefficients of the least squares fitted quadratic (ATF,BTF,CTF) are calculated by use of a matrix inversion routine called SOLVER and the minimum, VGV(n,1), and maximum, VGV(n,5), speeds of the gear are set. Three speed values are interpolated at regular spacing [VGV(n,2), VGV(n,3), and VGV(n,4)] and the values of tractive effort for these five speeds are calculated [TRACTF(n,1),...,TRACTF(n,5)]. A new gear, n+1, is then started from the last speed value VGV(n,5).

When these calculations are complete a subroutine called APPROX calculates the difference between the quadratic approximation to the points and the straight lines fitted between any two adjacent points within the speed range of the gear. Subroutine LINES and RESIDU are used here. RESIDU checks if the difference is large and sets an error indicator if it is. This test is used to avoid anomalous fits such as illustrated in Figure II.B.3.

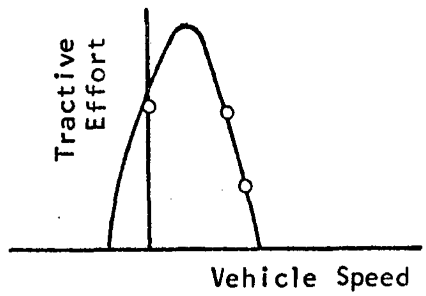


FIGURE II.B.3. -- Possible Anomaly in Quadratic Equation Fit to a Gear

When this occurs, the program produces a print-plot of the tractive effort vs. speed relationship (both points and fitted curve) and writes a message indicating where the problem occured and a suggestion that additional points be included in the tractive effort vs. speed array. The subroutines which are used to produce the print - plot are PLTSET, PNTRLT, SCAL, RESCAL, LIMITS and CURPLT.

If the basic tractive effort vs. speed relationship was originally entered as point pairs, all that is required is that additional point pairs be inserted as indicated. Otherwise, a possible solution is to enter the tractive effort vs. speed pairs as read from the printer-plot produced above as vehicle input data with the additional points inserted and to set MAPG to 0.

17. Subroutine II17 - Rotating Mass Factor

This routine calculates a factor which simulates the inertial mass of the rotating parts which have to be accelerated when the entire vehicle is accelerated. The factor varies depending on the gear in which the transmission is engaged. The formula is

$$\text{Rotating Mass Factor}_{NG} = 1 + m_{F1} + m_{F2} * (r_{NG})^2$$

where NG = gear number

m_{F1} = .14 if there is a tracked assembly on the vehicle

= .03 otherwise

$m_{F2} = [.008(id)^{1.68} n_e]/n_c W$

i = 2 if the engine is a two cycle diesel

= 1 otherwise

d = displacement in cubic inches

n_e = number of engines

n_c = number of cylinders

W = gross combined weight of vehicle

$r_{NG} = (F_{NG} r_w)/(\eta Q_m)$

F_{NG} = tractive effort at center speed of gear NG

r_w = rolling radius of driving wheel or sprocket radius

η = .7 if there is a tracked assembly

= .9 otherwise

Q_m = maximum torque of engine regardless of gear.

C. Terrain Preprocessor

The Terrain Preprocessor is a short subroutine whose primary purpose is to adjust dimensions of the incoming terrain data, to select specific terrain values from optional ones based on scenario variables, calculate some derived terrain descriptors and adjust the terrain for a snow cover if so called for.

First the obstacle dimensions, recognition distance, radius of curvature, and stem spacing are converted to inches. Grade and obstacle approach angle are converted to radians. One of the RCI's given for dry, normal and wet season is selected based on the value of the scenario variable ISEASN.

An elevation correction factor for engine performance is calculated using the equation

$$ECF = 1 - .04e/1000.$$

where e = elevation of the terrain unit in feet, an input variable.

There are terrain situations where, even though the input data indicates obstacles are present in the patch, their effect on vehicle performance is negligible. In this case a flag, IOBS, is set to 1 indicating a patch bare of obstacles. This is done when

1. obstacle spacing is greater than 197 feet
2. obstacle approach angle is from 179 to 181 degrees.

Several parameters describing obstacle spacing are given by the equations below. A test is made to determine the consistency of the three input obstacle parameters in case of a mound since the class interval method of obstacle definition (e.g. as read by MAP71) could result in obstacles that cannot be physically realized. In this case the obstacle base width, OBW, is altered to be consistent with the other two parameters α_o and h_o. The resulting obstacle base width is designated here by w_o.

The ground level width of the obstacle:

$$w_{og} = w_o + 2h_o \text{ABS}(\cos\alpha_o/\sin\alpha_o) \quad \text{for trenches}$$
$$= w_o \quad \text{for mounds}$$

where w_o = base width of obstacle

h_o = obstacle height

α_o = obstacle approach angle

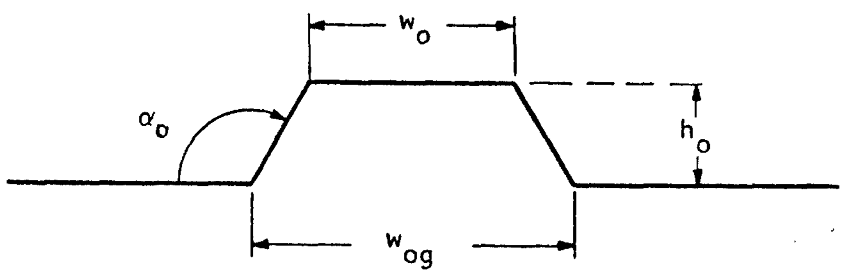


FIGURE II.C.1 -- Side View of an Obstacle

The top of a mound or bottom of a trench is the minimum obstacle width.

$$w_o = w_o \quad \text{for trench}$$
$$= w_o - 2h_o * \text{ABS}(\cos\alpha_o/\sin\alpha_o) \quad \text{for mound}$$

The maximum extent across an obstacle:

$$d_o = (w_{og}^2 + l_o^2)^{1/2}$$

where l_o = length of obstacle

It is assumed here that the ground level plan of the obstacles base is rectangular with width w_o and length l_o.

The mean obstacle approach width:

$$\bar{w}_{oa} = 2(l_o + w_{og})/\pi$$

It is assumed here that the rectangle represented by the base of the obstacle may be oriented at any angle to the approach path of the vehicle and that this approach angle is uniformly distributed between 0 and $\pi/2$. The mean approach width is derived by:

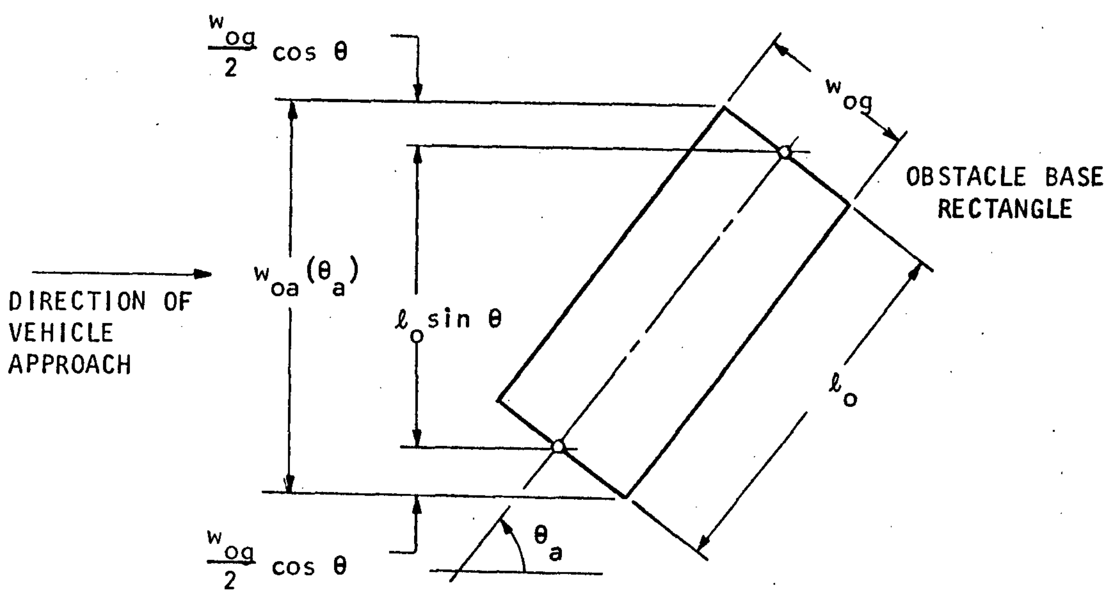


FIGURE II.C.2 -- Vehicle Approach to Obstacle

where $w_{oa}(\theta_a)$ = obstacle approach width

$_a$ = vehicle approach angle

$$\bar{w}_{oa} = (2/\pi) \int_0^{\pi/2} w_{oa}(\theta)d\theta$$

Note that for obstacle crossing, both in the Obstacle Module and in the Areal Module, it is assumed that θ_a = 90 degrees. The calculations here are used in the calculations of speed reduction due to maneuvering around obstacles in Areal Module Subroutine IV1.

Average Terrain Unit Area per Obstacle:

$$A_o = \pi(S_o/2)^2$$

where S_o = average obstacle spacing.

This average spacing is calculated by counting the number of obstacles in a large circular area of diameter D. Obstacle spacing is then

$$S_o = [(\pi/N)(D/2)^2]^{1/2}$$

where N = number of obstacles in a circular area of diameter D. S_o is an input terrain variable. Since S_o can be interpreted as the circle diameter which on average contains one obstacle, the area per obstacle is given by the above formula.

For regularly spaced obstacles which cannot be avoided

$$w_{og} = d_o = \bar{w}_{oa} = A_o = 0.$$

If the scenario variable ISNOW calls for a snow cover of z_s inches, the "snow machine" is used. This is a portion of code which sets the surface/soil type to signal snow (IST = 4), reduces the height of obstacles $z_s\gamma/.8$ where γ = snow specific gravity, and attenuates the surface roughness by a factor of $\gamma/.4$ for snow depth below twice the surface roughness, RMS, and by a factor of

$$1 - 1/2(1 - \gamma/.4)\ (z_s/RMS)$$

for snow depths above twice RMS.

No snow is permitted on water covered terrain units and the above obstacle attenuation is not performed for roads and trails (which have no obstacles). The Terrain Preprocessor then returns to the Control and I/O Module.

D. Areal Module

The Areal Module, similar to the Vehicle Preprocessor, is a series of subroutines which are called sequentially by subprogram AREAL of the Control and I/O Module. This section will describe these subroutines in the order that they are called.

1. Subroutine IV1 - Obstacle Spacing and Area Denied

This subroutine calculates data for factors which are used in Subroutines IV2,IV15,IV16,IV17 and IV18 to model the average speed lost due to the increased time it would take a vehicle to maneuver around obstacles and vegetation. This speed loss is related to the size of the vehicle and the density (inverse spacing) of the obstacles and vegetation.

Vegetation, in NRMM, is categorized into NI categories and ranked from small to large stem diameter values. For the data files read by MAP71 and MAP74, NI = 8 and the spacing for a class is that for all the vegetation in that and higher classes. The data files yield a value for the average spacing of the vegetation in each stem diameter class. (The word "class" has been in common use for "category" in this field.) The vehicle/driver speed will be selected from those achieved under a variety of possible avoidance and override strategies. These strategies are all combinations of avoiding and overriding obstacles and avoiding or overriding vegetation in certain size classes as described below.

Within the program these various speed estimates are indexed by stem diameter classes i = 1,2,...,NI. Thus, SRFV(i) stands for the speed reduction factor due to overriding vegetation in stem diameter classes 1,2,...,i-1 and avoiding stem diameter classes i,i+1,...,NI. Furthermore, SRFO(i) stands for the speed reduction factor due to avoiding obstacles while overriding vegetation in stem diameter classes 1,2,...,i-1 and avoiding those in classes i,i+1,...,NI. Eventually these 2NI factors will be applied (although not as explicitly calculated variables in the program) to 2NI speeds for each of 3 slope crossing conditions (uphill, level and downhill) if cross country simulation is called for (NTRAV = 3). This results in 6NI speed estimates. If NTRAV = 1, only one slope crossing condition is estimated and 2NI speeds are calculated.

This calculation of these speed reduction factors is based on the concept that each obstacle or tree to be avoided can be translated into an area of the terrain that is denied to the vehicle, or more precisely above which the CG of the vehicle cannot go. For instance, for a tree with a diameter of d and a vehicle with a width w the area denied by that one tree is a circle of radius (d+w)/2 centered at the tree. If many of the areas denied are scattered at random over the terrain unit, any one traverse will be forced to deviate from a straight line for maneuvers around the scattered obstructions. The length of the path is increased when the density of the obstructions and/or the size of the vehicle is increased. Both of these factors increase the area denied. For purposes of NRMM an empirical relation between speed reduction and overall area denied is used to account for obstruction avoidance.

In this routine the area denied due to the obstacles, ADO, and the area denied due to avoiding vegetation in classes i,i+1,...,NI, PAV(i), are calculated. Several checks are made for various conditions.

Obstacle avoidance is considered first. If the terrain unit is bare of obstacles, NEVERO is set to 2 indicating that the obstacle override calculations are to be skipped and ADO = 0, indicating no area is denied due to obstacles. The routine then considers vegetation avoidance.

Alternately, if the obstacles are so arranged that they are unavoidable, such as in rice paddies, ADO = 100. and the effective obstacle spacing OBSE = OBS, the actual obstacle spacing.

If obstacles are potentially avoidable, consideration is given as to whether they are small enough to fit under the vehicle or must be bypassed. The variable WI represents the minimum width between running gear elements of the vehicle, i.e., an obstacle no wider than WI will fit under the vehicle if it is no higher than CL, the ground clearance of the vehicle. If the obstacle is wider than WI it will not fit between the running gear and the effective width of the obstacle, EWDTH, is the width of the vehicle, WDTH, plus the width of the obstacle, OAW. The effective obstacle spacing is then OBSE = AREAO/EWDTH, the area assigned to each obstacle divided by the effective obstacle width.

If the obstacle fits between the running gear elements, the effective width of the obstacle is the path width of a single running gear element, PWTE, plus the obstacle width, OAW. The effective obstacle spacing is then calculated as above. A check is made if the obstacle is higher than the ground clearance; if it is, NEVERO = 1 to indicate no obstacle override since the obstacle is too narrow to support the vehicle and too high to fit under it. Then the effective width and spacing is calculated from the full vehicle width, WDTH.

The area denied by the obstacles is then calculated by

ADO = 100 (area denied by a single obstacle)/(terrain unit area per obstacle).

The terrain unit area per obstacle is modeled as a circle whose diameter is the effective obstacle spacing. This area is $\pi(OBSE/2)^2$.

The area denied by a single obstacle is modeled by surrounding the rectangular obstacle base by a band the half-width of the vehicle and summing the areas of all the regions indicated in the figure II.D.1.

Thus the area denied by a single obstacle is

$$OBL*WA + 2(OBL*WDTH/2) + 2(WA*WDTH/2) + \pi(WDTH/2)^2.$$

The last term is the sum of the four quarter circles which form a full circle of radius WDTH/2.

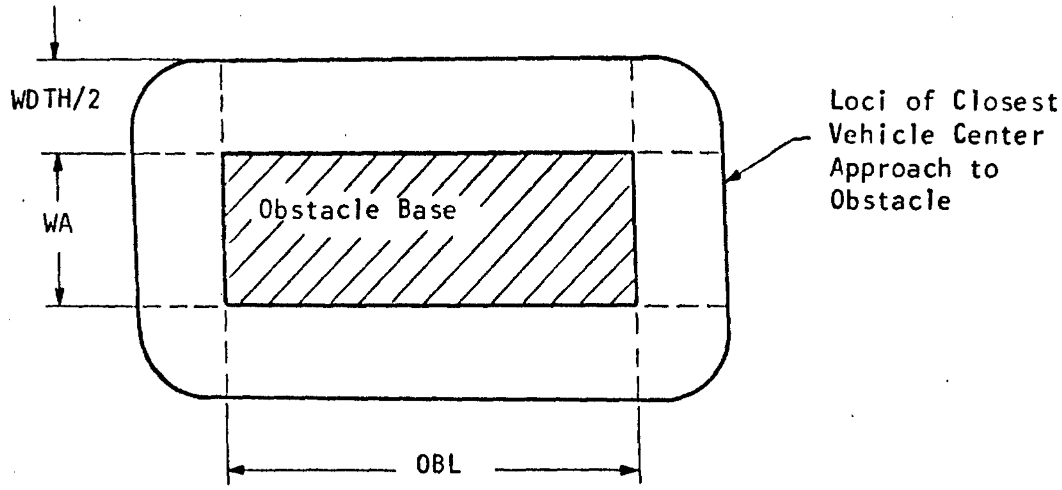


FIGURE II.D.1 -- Area Denied Due to an Obstacle

To calculate the area denied by the vegetation to be avoided the density of stems in each stem diameter class is calculated first. Since the input data gives the average spacing of vegetation in a stem diameter class and greater, the spacing of individual classes has to be separated. This is given by:

$$d_i = (4/\pi)[1/s_i^2 - 1/s_{i+1}^2] \qquad i = 1,...,NI-1$$

where d_i = density of stems in stem diameter class i

s_i = average spacing of stems in stem diameter classes i,i+1,...,NI.

NI = number of stem diameter classes

The percentage of the area denied by avoiding stems in stem diameter classes i, i+1,...,NI is given by

$$PAV_i = 100/s_i^2 [\sum_{j=i}^{NI} d_{vj} \, d_j / \sum_{j=1}^{NI} d_j \; - WDTH]$$

where d_{vj} = diameter of stems in stem diameter class j

WDTH = width of the vehicle.

The total area denied due to avoiding obstacles and stems in stem diameter classes i,i+1,...,NI is given by

$$ADT_i = ADO + PAV_i(100-ADO)/100$$

This formula takes some account of the possibility that the vegetation grows on obstacles. No additional area would be denied due to avoiding such vegetation.

2. Subroutine IV2 - Land/Marsh Operating Factors

This routine accounts for terrain units which may be water covered. The effect of the water may be to support part of the weight of the vehicle, reducing the effective normal forces on the running gear needed for traction. At the limit, the vehicle may be fully supported and will be swimming.

The routine first screens the terrain unit type. If it is dry, the float indicator is set to zero, r_{water} (WRATIO in the code) is set to one indicating the full load of the vehicle is on the running gear, and the under water drag area (DAREA) is set to zero. The routine then sets the tire pressure indicator, JPSI, and exits.

If the terrain unit is not dry, a test is made to determine if the water is sufficiently deep to prohibit fording. This can occur for a nonswimming vehicle and will result in a no go indicator, NOGOWD, to be set indicating the vehicle cannot proceed across this terrain unit. For a swimming vehicle this will result in a fully floating operation for which the selected terrain unit speed is set to the swimming speed, VSS, modified by vegetation avoidance. Water covered patches are assumed bare of obstacles.

If the vehicle can ford, that is, when the water is sufficiently deep to be noted but not deep enough to lift the vehicle clear off the ground or to stop its progress, the buoyancy is calculated by linear interpolation in a table of water depth (WDPTH) vs. weight reduction ratio (WRAT) yielding a value of r_{water} below 1.0 which, when applied to vehicle weight terms will reduce ground contact pressure. A final calculation results in the frontal area of the vehicle subjected to the water drag forces for later incorporation into the driving/braking force calculations. Before exit this routine sets JPSI, the tire pressure index. See description of the next routine for an explanation.

3. Subroutine IV3 - Pull and Resistance Coefficients

In this routine the draw-bar pull over weight (DOW) and resistance over weight (RTOW) coefficients for the given vehicle and the soil of the current terrain unit are calculated. Separate pull and resistance coefficients are calculated for each suspension assembly, thus allowing the simulation of half tracks and tracked vehicles pulling wheeled trailers as well as combinations with large weight variations between axles. Also, different resistance coefficients are calculated for each assembly when it is braked and/or powered (DOWPB,RTOWPB) as opposed to towed in a free wheeling mode (RTOWT). The basic equations are included in Rula and Nuttall (1971) with revisions by Turnage (1972).

Four surface types are included: fine grained soils, coarse grained soils, muskeg and snow. For fine grained soils, provisions are made to simulate the effect on traction of slippery soil surface conditions due to recent rainfall, flooding and or standing water. Separate slipperiness effects are included for CH soils, which are largely impervious to water, and for other, more pervious fine grained soils. Coarse grained soils, muskeg and snow are assumed to never have slipperiness conditions caused by standing water, flooding or rainfall.

Where soil is very soft (soil strength exceeding vehicle cone index by at most 20) slipperiness is not a factor. If this excess (RCIX) is greater than 20 the pull coefficients are reduced by an exponential relationship detailed below. At a certain level, given by RCIX > RCIS, the reduction factor becomes constant indicating a "skating condition" on an extremely hard surface. The routine also accounts for the presence (NPAD = 1) or absence (NPAD = 0) of track road pads.

As described in the various routines of the Vehicle Preprocessor above [Sections II.B.3, II.B.5, II.B.6, II.B.7, II.B.12, and II.B.13], allowance is made for vehicles with central inflation pressure systems by allowing changes in tire pressure due to various soil conditions. The input data may contain up to three different tire pressures for use on fine grained soils, coarse grained soils, and highways. The scenario variable NOPP indicates how these should be used. If NOPP = 0, a vehicle which can change its inflation pressure is being simulated and the pressure appropriate for the terrain surface is used. On fine grained soil, muskeg, snow and water covered terrain units the fine grained soil pressure is used. If NOPP is not 0, the tire pressure is set to the inflation pressure to be used for all terrain units regardless of the type. The variable JPSI, set in routine IV2, indicates the pressure to be used: JPSI = 1 for fine grained soil pressure, JPSI = 2 for coarse grained soil pressure, and JPSI = 3 for highway pressure.

In the current subroutine, JPSI indexes the value of VCIFG, DRAT, CPFFG, CHARLN, and GCA to be used. These are, respectively, the vehicle cone indices for fine grained soil, the tire deflection ratio, the contact pressure factor for fine grained soil, the characteristic lengths of the traction elements, and the ground contact area. All of these values depend on tire deflection which depends on the inflation pressure.

This subroutine uses three other subroutines, FGSTR, FGSPC, and FGSPR. These will now be described.

Subroutine FGSTR calculates the fine grained soil towed motion resistance (RTOW) for a suspension assembly. The routine first checks if the assembly is both powered and braked. If so, the assumption is made that it will never be free rolling (towed) so RTOWT, the returned coefficient, is set to zero. If the assembly may be towed and is tracked an error list is written on unit LUN1 and the program is halted since towed, tracked assemblies are not simulated. For towed, wheeled assemblies

$$W' = W_i \, r_{water}/n_i$$

where W_i = weight on axle i

r_{water} = weight reduction ratio due to buoyancy (=1. for dry terrain units)

n_i = number of wheels on axle i

Then

$$\beta = [RCI b_i d_i \, (\delta_{ij})^{1/2}]/[W' \, (1 - .5 b_i/d_i)]$$

where RCI = rated cone index for assembly

b_i = section width of tire

d_i = outside diameter of tire

δ_{ij} = deflection ratio for pressure j

and

$$RTOW = 1 - .3412\beta \qquad \text{for } \beta \leq 2$$

$$RTOW = .04 + .2/(\beta - 1.35) \qquad \text{for } \beta > 2$$

This concludes Subroutine FGSTR.

Subroutine FGSPC calculates the fine grained soil pull coefficient (DOW). It depends on the contact pressure factor (CPF) and the excess rating cone index (RCIX) as follows:

$$RCIX = RCI - VCIFG_{ij}$$

For tracked assemblies and CPF $\leq$ 4

$$DOW = .544 + .0463RCIX - [(.544 + .0463RCIX)^2 - .0702RCIX]^{1/2}$$

for RCIX < 0, DOW = .076RCIX

For tracked assemblies and CPF $\geq$ 4

$$DOW = .455 + .0392RCIX - [(.455 + .0392RCIX)^2 - .0526RCIX]^{1/2}$$

for RCIX < 0, DOW = .056RCIX

For wheeled assemblies and CPF < 4

$$DOW = .3885 + .0265RCIX - [(.3885 + .0265RCIX)^2 - .0358RCIX]^{1/2}$$

for RCIX < 0, DOW = .046RCIX

For wheeled assemblies and CPF $\geq$ 4

$$DOW = .379 + .0219RCIX - [(.379 + .0219RCIX)^2 - .0257RCIX]^{1/2}$$

for RCIX < 0, DOW = .033RCIX

This concludes Subroutine FGSPC.

Subroutine FGSPR calculates the motion resistance coefficient (RTOWPB) of a powered/braked assembly. Similarly to the previous routine, it depends on excess RCI and the contact pressure factor as follows:

For tracked assemblies:

$$RTOWPB = .045 + 2.3075/(RCIX + 6.5)$$

for $RCIX < 0$ and $CPF < 4$ $\quad RTOWPB = .4 - .072RCIX$

for $RCIX < 0$ and $CPF \geq 4$ $\quad RTOWPB = .4 - .052RCIX$

For wheeled assemblies and $CPF < 4$

$$RTOWPB = .035 + .861/(RCIX + 3.249)$$

for $RCIX < 0$ $\quad RTOWPB = .3 - .043RCIX$

For wheeled assemblies and $CPF \geq 4$

$$RTOWPB = .045 + 2.3075/(RCIX + 6.5)$$

for $RCIX < 0$ $\quad RTOWPB = .4 - .029RCIX$

This completes Subroutine FGSPR.

Returning to Subroutine IV3, initially a test is made to determine the soil type and a transfer is made to the appropriate portion of code which is described next.

a) Fine Grained Soil

For fine grained soils the excess RCI is calculated with respect to the $VCIFG_{ij}$ for assembly i and tire pressure j (if wheeled).

$$RCIX = RCI - VCIFG_{ij}$$

For an assembly never driven nor braked the powered resistance ($RTOWPB_i$) and pull coefficient ($DOWPB_i$) are set to zero and the towed resistance coefficient ($RTOWT_i$) is calculated by a call to Subroutine FGSTR. For powered or braked assemblies and a dry terrain unit successive calls to FGSPC, FGSPR and FGSTR are used to calculate $DOWPB_i$, $RTOWPB_i$ and $RTOWT_i$, respectively.

If the terrain unit is wet, the value of NSLIP indicates the extent of surface water according to Table II.D.1. For wheeled assemblies, the factor

$$x = \delta_{ij}/.4 - .375$$

where δ_{ij} = deflection ratio of tires on assembly i at inflation j, is used to account for the beneficial effects of high inflation pressure, which helps to maintain the "circular" shape of the tire and thereby improves the tire's ability to break through the slippery layer.

Table II.D.1

Slipperiness Conditions and Parameters

NSLIP	Meaning
1	less than 1" rain with no free water
2	less than 6 hours rain with no free water
3	more than 6 hours rain with no free water
4	less than 1" rain with free surface water
5	less than 6 hours rain with free surface water
6	more than 6 hours rain with free surface water

CH Soils Impervious to Water

	Tracked Assemblies		Wheeled Assemblies	
NSLIP	DOWCS	RCIS	DOWCS	RCIS
1	.5	200	.35	300
2	.3	150	.25x	150
3	.3	200	.2x	200
4	.1	200	.15x	150
5	.1	300	.15x	150
6	.15	500	.15	100

All Other Fine Grained Soils

	Tracked Assemblies		Wheeled Assemblies	
NSLIP	DOWCS	RCIS	DOWCS	RCIS
1	.45	100	.3	80
2	.3	100	.1	80
3	.2	100	.1	80
4	.1	100	.1x	80
5	.1	100	.1	80
6	.15	100	.1	80

These relationships are not used for excess rating cone index, RCIX, less than or equal to 20. In that case the assumption is made that the soil is weak and plastic in relation to the load to be imposed on it by the vehicle and therefore surface water will not have a significant effect on traction and resistance. The coefficients $DOWPB_i$, $RTOWPB_i$ and $RTOWT_i$ are calculated by calls to subroutines FGSPC, FGSPR, and FGSTR, respectively.

For a soil/vehicle combination where the excess rating cone index (RCIX) exceeds 20, values for comparison rating cone index (RCIO) and pull coefficient (DOWCO) are set as follows:

Tracked Assemblies: RCIO = 18. DOWCO = .4

Wheeled Assemblies: RCIO = 20. DOWCO = .55

and RCIS and DOWCS are set according to Table II.D.1.

If RCIX exceeds RCIS (from the table) and the assembly is wheeled, the pull coefficient is set to the table value of DOWCS and $RTOWPB_i$ and $RTOWT_i$ are calculated using subroutines FGSPR and FGSTR, respectively.

In case of a tracked assembly (and RCIX greater than or equal to RCIS), a further distinction is made for the presence of track road pads. If track pads are present (NPAD =1) the same calculations as for wheeled assemblies are made (under the observation that in both cases a rubber/soil interface exists). If there are no pads (NPAD = 0), in order to include the effect of grouser action when no pads are fitted the pull coefficient ($DOWPB_i$) is set to the average of DOWCS (from the table) and D (as calculated by subroutine FGSPC). The resistance coefficients are again calculated by subroutine FGSPR and FGSTR.

For excess rating cone index (RCIX) less than RCIS (from table) the pull coefficient (in this case DOWS) is calculated by the log-linear relation

$$(\log DOWS - \log DOWCS)/(\log DOWCO - \log DOWCS) = (\log RCIX - \log RCIS)/(\log RCIO - \log RCIS)$$

For tracked assemblies with track pads, DOWS from this equation is averaged with the pull coefficient D from Subroutine FGSPC to form $DOWPB_i$. Otherwise DOWS becomes $DOWPB_i$. The resistance coefficients $RTOWPB_i$ and $RTOWT_i$ are calculated by subroutines FGSPR and FGSTR.

b) Coarse Grained Soil

This portion of Subroutine IV3 calculates the pull and resistance coefficients for each assembly when the terrain unit contains coarse grained soil. Dimensionless numerics developed by Turnage (1972) are used instead of the vehicle cone index (VCI). A basic cone index gradient term is calculated from

$$G = .8645CI/3.$$

For tracked vehicles, the towed resistance is set to zero since the NRMM does not model towed tracks. The pull coefficient is calculated from the pi term

$$\pi_t = [.6G(b_i l_i)^{1.5}]/W_i$$

where W_i = weight on assembly i

b_i = track width

l_i = track length on ground

by

$$DOWPB_i = .121 + .258 \log \pi_t \qquad \text{for} \qquad \pi_t \leq 25$$

$DOWPB_i = .339 + .109 \log \pi_t$ for $25 < \pi_t \leq 100$

$DOWPB_i = .481 + .038 \log \pi_t$ for $100 < \pi_t \leq 1000$

$DOWPB_i = .595$ for $1000 < \pi_t$

Then the powered/braked resistance coefficient is calculated from

$RTOWPB_i = .6 - DOWPB_i$.

For wheeled axles, if the axle is never towed, the towed resistance coefficient, $RTOWT_i$, is set to zero. If the axle can be towed, the pi term is calculated by

$$\pi_t = G(b_i d_i)^{1.5} \; i^{1/3} / [(1-\delta_{ij})^3 (1+ b_i/d_i) W_i/n_i]$$

where W_i = weight on axle

n_i = number of wheels on axle

b_i = section width of tires on axle

d_i = diameter of tires on axle

i = axle number (from front)

δ_{ij} = deflection ratio for tires at inflation j

The towed resistance coefficient is then calculated by

$$RTOWT_i = .44 - .01\pi_t + [(.44 - .01\pi_t)^2 + .0002\pi_t + .08]^{1/2}$$

If the wheeled axle can be powered or braked the width and weight are adjusted for the presence or absence of dual wheels. The pi term is

$$\pi_d = G(Bd_i)^{1.5} \delta_{ij} / W[i]^{1/2}$$

where the above notation is used with

$B = b_i$ for singles, $2b_i$ for duals

$W = W_i/n_i$ for singles, $2W_i/n_i$ for duals.

The pull coefficient is then given by

$$DOWPB_i = .53 - 4.5/(\pi_d + 3.7)$$

and the powered/braked resistance is

$$RTOWPB_i = .6 - DOWPB_i.$$

c) Muskeg

This portion of the subroutine IV3 calculates the pull and resistance coefficients for each suspension assembly when the surface of the terrain unit is designated as muskeg or peat. The equations used are basically those developed for fine grained soils when the contact pressure factor is greater than or equal to 4psi.

The excess rating index is calculated from

$$RCIX = RCI - VCIMUK_i$$

where RCI = rating cone index of terrain unit

$VCIMUK_i$ = vehicle cone index calculated for assembly i

A candidate resistance coefficient is calculated by

$$RT = 1. \quad \text{if} \quad RCIX \leq -100$$

$$RT = 1. - .006(RCIX + 100.) \quad \text{if} \quad -100 < RCIX \leq 0$$

$$RT = .045 + 2.3075/(6.5 + RCIX) \quad \text{if} \quad 0 < RCIX$$

The towed resistance coefficient is set by

$RTOWT_i$ = RT if assembly may be towed

= 0 if assembly is always either powered or braked.

The powered resistance coefficient is set by

$$RTOWPB_i = RT \quad \text{if assembly may be powered or braked}$$
$$= 0 \quad \text{if assembly can only be towed.}$$

For unpowered assemblies the pull coefficient is set to zero. For powered assemblies the following cases are distinguished:

$$DOWPB_i = -1. \qquad \text{if} \quad RCIX \leq -100.$$
$$DOWPB_i = -1. + .1(RCIX + 100.) \qquad \text{if} \quad -100. < RCIX \leq 0$$
$$DOWPB_i = .5464 + .1091RCIX - [(.5464 + .1091RCIX)^2 - .192RCIX]^{1/2}$$

if 0<RCIX and the vehicle is tracked with contact pressure factor less than 4 psi

$$DOWPB_i = .3537 + .02258RCIX - [(.3537 + .02258RCIX)^2 - .03071RCIX]^{1/2}$$

in all other cases.

d) Shallow Snow

This portion of subroutine IV3 calculates the pull and resistance coefficients for each suspension assembly when the scenario variables indicate that the terrain unit is covered with a shallow layer of snow. For this model shallow snow is defined as snow covering frozen ground at a depth less than the characteristic length of the tire or less than one third of the track length on ground.

Resistance is based on the force required for bulk movement of snow whereas traction is based on the Coulomb equation. Thus the towed resistance for wheeled vehicles is calculated from

$$RT = (10n_i b_i \gamma z_s)/(Nd_i l_{ij})$$

where n_i = number of wheels on assembly i

N = total number of wheel axles on vehicle

b_i = section width of tire on assembly i

d_i = diameter of tire on assembly i

γ = specific weight of the snow

z_s = snow depth

l_{ij} = characteristic length of tire on assembly i at inflation j (see Section II.B.6))

and for tracked vehicles from

$$RT = (\gamma z_s)/(2l_{ij})$$

where l_{ij} = the characteristic length of the track.

For suspension assemblies that are never towed, $RTOWT_i = 0$. For assemblies that may be powered or braked the above equations are used for $RTOWPB_i$.

The pull coefficients are calculated using

$$TOWMAX = \tan\varphi + (cA_{ij}n_i)/W_i \qquad \text{wheeled assembly}$$

$$TOWMAX = \tan\varphi + (cA_i/W_i) \qquad \text{tracked assembly}$$

where φ = internal angle of friction

c = cohesion

A_i, A_{ij} = ground contact area

n_i = number of wheels on assembly i

W_i = weight supported by assembly i.

The pull coefficient is then set to

$$DOWPB_i = TOWMAX - RT.$$

In all cases (fine grained soil, slippery fine grained soil, coarse grained soil, muskeg or snow) Subroutine IV3 passes on

$RTOWT_i$ = towed resistance coefficient

$RTOWPB_i$ = powered/braked resistance coefficient

$DOWPB_i$ = pull coefficient

for each suspension assembly i.

4. Subroutine IV4 - Summed Pull and Resistance Coefficients

In this routine the individual suspension assembly resistance and pull coefficients are summed to provide overall, average vehicle coefficients. The formulas concerning traction are:

RTOWP = sum of $RTOWPB_i*W_i/GCWP$ for powered assemblies i

DOWP = sum of $DOWPB_i*W_i/GCWP$ for powered assemblies i

RTOWNP = sum of $RTOWT_i*W_i/(GCW-GCWP)$ for unpowered assemblies i

where RTOWP = average powered assembly resistance coefficient

DOWP = average powered assembly pull coefficient

RTOWNP = average un-powered assembly resistance coefficient

W_i = weight supported by assembly i

GCWP = weight supported by all powered assemblies

GCW = gross combination weight

$RTOWPB_i$ = powered resistance coefficient of assembly i

$DOWPB_i$ = pull coefficient of assembly i

$RTOWT_i$ = towed resistance coefficient of assembly i.

The formulas concerning braking are similar to the above except that the summation is over the braked and non-braked assemblies.

5. Subroutine IV5 - Slip Modified Tractive Effort

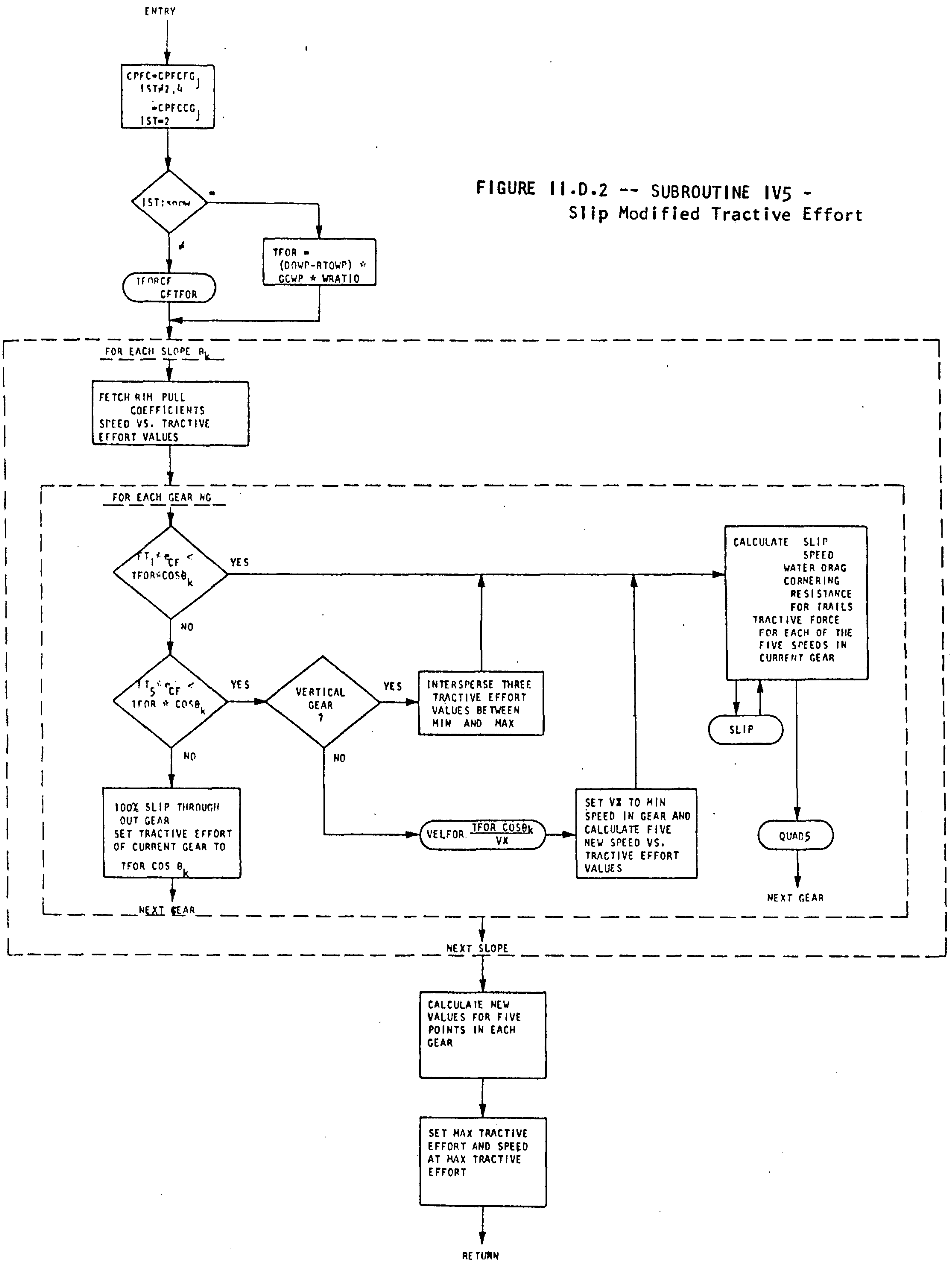
This routine modifies the vehicle tractive effort vs. speed relationship, calculated in Subroutine II16 of the VPP, for slippage of the running gear in the soil. This relationship was stored as the coefficients of a quadratic relating speed to tractive effort in a "gear". Each gear was specified as an interval in the speed range of the vehicle.

This subroutine calls four other subroutines named TFORCF, VELFOR, SLIP, and QUAD5. These routines calculate

1. the soil limited maximum tractive effort (TFOR) available to the vehicle [by TFORCF]
2. the maximum velocity (VX) achievable when just overcoming a given resistance [by VELFOR]
3. the slip of the running gear (SLIP) when operating at a certain pull force coefficient [by SLIP]
4. the least square fitted quadratic to five points under the constraint that the fitted curve must pass through the extreme points of the independent variable, SPEED, [by QUAD5].

Subroutine IV5 first retrieves the appropriate ground contact pressure factor (either $CPFCFG_j$ or $CPFCCG_j$) and, in the case of soil types other than snow, calls subroutine TFORCF to calculate the maximum tractive effort (TFOR) available from the soil. In case of snow cover this is calculated from the pull (DOWP) and powered resistance (RTOWP) coefficients times the effective weight on the powered wheels. The routine then performs the following calculations

FIGURE II.D.2 -- SUBROUTINE IV5 -
Slip Modified Tractive Effort



for each slope (θ_k) and gear (NG).

Gears do not necessarily correspond to real gears in the transmission; they are intervals in the total speed range for which the tractive effort vs. speed curve (rim pull curve) can be well approximated by a quadratic

$$F = a_{NG} + b_{NG}v + c_{NG}v^2 \quad \text{for} \quad v_{1,NG} \leq v \leq v_{5,NG}.$$

Five points $\{(v_{i,NG}, F_{i,NG}), i=1,...,5: v_{i-1,NG} \leq v_{i,NG}, i=2,...5\}$ are given for the curve for each gear NG. A "vertical" gap, as shown in figure II.D.2, at speed $\bar{v}$ is approximated as a gear with $v_{1,NG} = ... = v_{5,NG}$ and five values of F as indicated by the figure. It is generally assumed that $F_{1,NG}$ is the maximum tractive effort in gear NG (i.e. $F_{1,NG} \geq F_{i,NG}$) and that $F_{5,NG}$ is

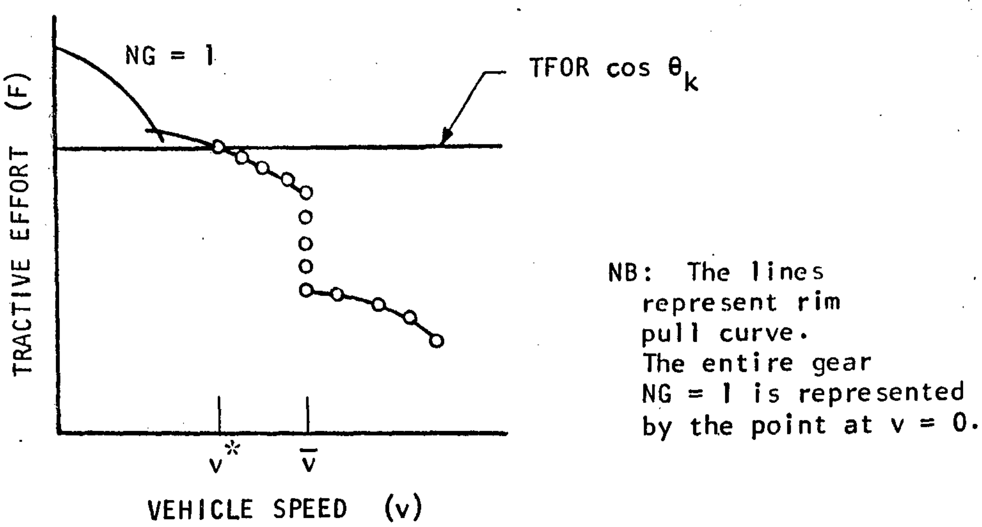


FIGURE II.D.3 -- Tractive Effort vs. Speed Modification for Soil and Slope the minimum tractive effort in gear NG (i.e. $F_{5,NG} \leq F_{i,NG}$).

The routine fetches the coefficients (a_{NG}, b_{NG}, c_{NG}) and the points ($v_{i,NG}$, $F_{i,NG}$) for each gear. These will be modified for slip. First the total tractive effort available from the vehicle in each gear NG, $F_{1,NG}e_{CF}$, corrected by a terrain unit elevation factor e_{CF}, is compared to the slope modified maximum surface traction TFORcosθ_k. If not all the vehicle tractive force can be applied ($F_{1,NG}e_{CF} \geq$ TFORcosθ_k) the minimum surface tractive force $F_{5,NG}e_{CF}$ is compared to the maximum surface traction. If the vehicle tractive effort exceeds the maximum surface tractive force throughout the gear ($F_{5,NG}e_{CF} >$ TFORcosθ_k) the entire gear is approximated by 100% slip, the speeds $v_{i,NG}$ are all set to zero and the tractive effort is set to a constant TFORcosθ_k. The quadratic then reduces to a_{NG} = TFORcosθ_k with $b_{NG}=c_{NG}=0$.

If the soil can support a tractive effort between the minimum and maximum of the gear NG ($F_{5,NG} <$ TFORcos$\theta_k \leq F_{1,NG}$), the subroutine VELFOR is called to determine the speed v* in the interval [$v_{1,NG}$,$v_{5,NG}$] at which the vehicle produces the maximum surface tractive effort. The interval representing the gear NG is now adjusted to be [$v_{1,NG}$ = v*,$v_{5,NG}$], the maximum tractive effort for this gear is reset to

$$F(v^*) = a_{NG} + b_{NG}v^* + c_{NG}(v^*)^2$$

and five new values of tractive effort are calculated at equally spaced speed values from v* to $v_{5,NG}$. These become the new gear {($v_{i,NG}$,$F_{i,NG}$), i=1,...,5: $v_{1,NG}$=v*}. If this gear was a vertical gear, the five points would be equally spaced from $F_{1,NG}$ to $F_{5,NG}$ all at $v_{1,NG}$.

If the maximum available tractive effort in the soil, modified for slip, is greater than the rim pull tractive effort, corrected for elevation, everywhere in the gear ($TFORcos\theta_k > F_{i,NG}e_{CF}$) the computations proceed directly to adjustment of the speeds $V_{i,NG}$ for slip.

The above computations have the effect of limiting the tractive effort vs. speed curve of the vehicle, the rim pull curve, by the maximum tractive effort available from the terrain unit surface material. Each of the five points ($v_{i,NG}$,$F_{i,NG}$) for each gear NG are now individually adjusted for slip, altitude, and extra drag using the equations below. The force coefficient is calculated by

$$y = (F_{i,NG}\ e_{CF}\ r_w)/(GCWPcos\theta_k) - CF$$

where r_w = proportion of vehicle weight on running gear (=1 except possibly for water covered terrain units)

CF = slip curve correction factor (calculated in Subroutine TFORCF)

GCWP = gross combination weight on powered suspension elements.

This ratio is used by Subroutine SLIP to calculate the slip required of the traction elements (SLIPX) to produce force $F_{i,NG}$ for the given vehicle and terrain unit. The vehicle speed is then adjusted by

$v_c = v_{i,NG}$ (1. - SLIPX)

and the new point $(v_c, F_{i,NG} e_{CF})$ replaces $(v_{i,NG}, F_{i,NG})$ in the tractive effort vs. speed curve for the current terrain unit.

For water covered terrain units a hyperbolic drag is calculated from

$$w_D = (.00111 C_D\ A\ v_c^{2})/2$$

where C_D = drag coefficient

A = submerged frontal area

and this drag is subtracted from $F_{i,NG} e_{CF}$.

For terrain units designated as trails, a cornering drag for wheeled assemblies is calculated by

$$F_c = F_e \{\text{sum of } (W_e \cos\theta_k v_{iNG,MPH}/111.1\ R')^2$$
$$[.75/n_1\alpha_1(TFOR/GCWP)] \quad \text{for all wheeled assemblies } 1$$

where F_e = superelevation factor given by

$F_e = 1 - 7.495R'e$

e = superelevation angle

R' = radius of curvature in feet

$v_{iNG,MPH}$ = speed $v_{i,NG}$ in MPH

W_1 = weight on axle 1

n_1 = number of wheels on axle 1

α_1 = cornering stiffness of tires on axle 1

This force is subtracted from $F_{i,NG} e_{CF}$.

The five new tractive effort vs. slip corrected speed points for each gear are now fitted with a quadratic which is constrained to pass through the points for the minimum and maximum slip corrected speeds of the gear using Subroutine QUAD5. Before returning, subroutine IV5 sets the maximum available tractive effort value for each slope, FORMX(K), and the speed at which FORMX(K) occurs, VFMAX(K).

a) Subroutine TFORCF - Soil Limited Tractive Effort

The drawbar pull and traction vs excess rating cone index relationships used in NRMM are based on tests conducted at 20% slip (Turnage (1972)). Subroutine IV5 requires tractive effort at 100% slip.

This routine calculates the slip curve correction factor, CF, and the soil limited tractive effort, TFOR, according to the following formulas:

Fine Grained Soil

Tracked Vehicles

$$CF = DOWP - .758 + RTOWP \qquad \text{for } CPFC < 4$$

$$TFOR = (CF + .82)GCWP$$

$$CF = DOWP - .671 + RTOWP \qquad \text{for } CPFC \geq 4$$

$$TFOR = (CF + .71)GCWP$$

Wheeled Vehicles

$$CF = DOWP - .674 + RTOWP \qquad \text{for } CPFC < 4$$

$$TFOR = (CF + .76)GCWP$$

$$CF = DOWP - .585 + RTOWP \qquad \text{for } CPFC \geq 4$$

TFOR = (CF + .655)GCWP

Coarse Grained Soil

Tracked Vehicles

CF = .074 for rigid track

TFOR = (CF + .568)GCWP

CF = .1 for flexible track

TFOR = (CF + .695)GCWP

Wheeled Vehicles

CF = DOWP - .56 + RTOWP

TFOR = (CF + .575)GCWP

Muskeg

Wheeled Vehicles and CPFC $\geq$ 4

CF = DOWP - .68 + RTOWP

TFOR = (CF + .745)GCWP

All other cases

CF = DOWP - .88 + RTOWP

TFOR = (CF + .91)GCWP

b) Subroutine VELFOR - Maximum Velocity Overcoming a Given Resistance

This routine finds the maximum velocity, v*, that a vehicle can travel while overcoming a given resistance from the tractive effort vs. speed curve after adjustment for soil limited traction and driving element slip.

The tractive effort vs. speed curve is given by

$$F = a_{NG} + b_{NG}v + c_{NG}v^2,$$

a sequence of quadratics in various speed ranges ($v_{1,NG} \leq v \leq v_{3,NG}$), where NG indexes the speed ranges from 1 to NGR, the number of speed ranges (or gears). Note that for this routine there are three speeds given for each gear. This routine in effect solves for maximum v given an F by solving the quadratic equation

$$c_{NG}v^2 + b_{NG}v + (a_{NG} - F) = 0.$$

Let the discriminant be denoted by

$$d^2 = b_{NG}^2 - 4(a_{NG} - F)c_{NG}.$$

For $d^2 < 0$ it must be true that $c_{NG} \neq 0$ and $a_{NG} - F \neq 0$. Then the quadratic has no real solution. If $c_{NG} > 0$, the tractive effort vs. speed curve for the gear NG lies entirely above the value F so the speed, v*, that the vehicle can achieve is set to the maximum in the gear, namely $v^* = v_{3,NG}$. If $c_{NG} < 0$ the entire curve for the gear is below the value F so the vehicle cannot overcome the resistance in the gear NG and thus a lower gear is tested.

For $d^2 = 0$ two cases can occur. For $c_{NG} \neq 0$ there is a unique intersection between the quadratic and the line F = constant. Since a single point intersection (tangency) between the tractive effort vs. speed curve and the line F = constant is, realistically, similar to no intersection at all, decisions like those for the case of $d^2 < 0$ are made when $c_{NG} < 0$ [seek a lower gear] and when $c_{NG} > 0$ [set $v^* = v_{3,NG}$]. When $c_{NG} = 0$, then $b_{NG} = 0$ (since $d^2 = 0$) and thus the tractive effort vs. speed curve for gear NG is a horizontal line through a_{NG}. If $F < a_{NG}$ the vehicle can proceed at maximum speed in the gear [set $v^* = v_{3,NG}$] and if $F > a_{NG}$ the vehicle cannot overcome the resistance F in gear NG and thus a lower gear is sought.

For a positive discriminant, $d^2 > 0$, and $c_{NG} = 0$, the tractive effort vs. speed curve in gear NG is a straight line. If the intersection of this line with F = constant is to the right of the maximum speed, $v_{3,NG}$, in gear NG then if $b_{NG} > 0$ the curve is below F = constant in the speed range NG and the vehicle cannot overcome the resistance F in gear NG and thus a lower gear is sought. If $b_{NG} < 0$ then the curve is above F = constant and $v^* = v_{3,NG}$. If the intersection is to the left of the minimum speed, $v_{1,NG}$, in gear NG then the reverse is true, namely that $v^* = v_{3,NG}$ if $b_{NG} > 0$ and a lower gear is sought if $b_{NG} < 0$. For the case when the intersection occurs at (v,F) and $v_{1,NG} < v < v_{3,NG}$ then $v^* = v$ for $b_{NG} < 0$ and a lower gear is sought for $b_{NG} > 0$.

For a positive discriminant and $c_{NG} \neq 0$ there are two real roots for the quadratic equation, the greater being designated by v = RH while the lesser by v = RL. Three cases may be distinguished, the first for both roots negative. Then in the range of a gear the tractive effort vs. speed curve is either entirely above (indicated by $b_{NG} > 0$) or entirely below (indicated by $b_{NG} < 0$) the line F = constant. The $v^* = v_{3,NG}$ or lower gear is sought, respectively. In the second case only one root is positive, then results similar to the prior $d^2 > 0$, $c = 0$ case are used. The curves for positive v are not straight lines in this case but they are strictly monotonic with less and less curvature for increasing v. In the third case when both RH and RL are positive a test is made as to their relationship to $v_{1,NG}$ and $v_{3,NG}$. The cases are distinguished by the determination of whether the tractive effort vs. speed curve is above or below F = constant. If it is above at $v_{1,NG}$ then the highest root in the interval $v_{1,NG} < v < v_{3,NG}$ is used as v*. If the curve is above at both $v_{1,NG}$ and $v_{3,NG}$, then $v^* = v_{3,NG}$. If the curve is below at $v_{1,NG}$ a lower gear is sought.

If the subroutine cannot find a gear for which the tractive effort exceeds the resistance a final test against the maximum tractive effort, FORMX, is made. If FORMX $\geq$ F, the velocity $v^* = v_{max}$, the velocity at which the vehicle exhibits its maximum tractive effort. If FORMX < F, then $v^* = 0$.

c) Subroutine SLIP - Powered Traction Element Slip for Given Traction Coefficient

This routine uses empirical equations presented in Appendix A of Rula and Nuttall (1971) to determine the longitudinal slip of the powered traction elements in order to produce a given traction (pull) coefficient, y. These relationships are given by the following:

Fine Grained Soil

Tracked

$$S = .0257y - .0161 + .01519/(.8353 - y) \quad \text{for CPFC}<4$$

$$S = .0733y - .0063 + .00734/(.7177 - y) \quad \text{for CPFC}\geq 4$$

Wheeled

$$S = .0621y - .021 + .01888/(.7794 - y) \quad \text{for CPFC}<4$$

$$S = .084y - .016 + .01414/(.6697 - y) \quad \text{for CPFC}\geq 4$$

In case of wheeled vehicles with CPFC $\geq$ 4 on fine grained soil the slip is further reduced by dividing it by 1.1 if the vehicle is equipped with a locking differential.

Coarse Grained Soil

Tracked

$$S = -.0083 + .005312/(.573 - y) \quad \text{for rigid tracks}$$

$$S = 1.074y - .72 + [(1.074y - .72)^2 + .09y + .009]^{1/2} \quad \text{for flexible tracks}$$

Wheeled

$$S = .0074y - .0061 + .00374/(.5785 - y)$$

This last value of S is further reduced by dividing by 1.1 if the vehicle is equipped with a locking differential.

Muskeg

$$S = .0585y - .0106 + .01336/(.964 - y) \quad \text{for tracked vehicle with CPFC} < 4$$

$$S = .1024y - .00864 + .01062/(.7564 - y) \quad \text{all others}$$

In the case of wheeled vehicles with a locking differential, S is further reduced by dividing by 1.1.

Shallow Snow

$$S = .3(1 - [1-y]^{1/2}) \quad \text{for } y < 1$$

$$S = 1 \quad \text{for } y \geq 1$$

In all cases if S lies outside the interval $0 \leq S \leq 1$ it is set to 1.

d) Subroutine QUAD5 - Quadratic Fit to 5 Points

This subroutine uses the least square criterion to fit a quadratic to five points under the constraints that the curve must pass through the points with the lowest and highest value of the independent variable.

6. Subroutine IV6 - Resistance Due to Vegetation

This subroutine calculates the resistance to vehicle motion caused by vegetation when the vehicle attempts to override it. Since vegetation is categorized into NI classes on the basis of stem diameter, separate resistance forces are calculated for each class. If the terrain unit is bare of vegetation all the resistances are set to zero. All resistances are also set to zero for the smallest stem diameter class. Otherwise, for each stem diameter class i beginning at class 2 the force needed to override the largest tree in class i is given by

$$F_{v,i+1} = (56/5.8)*d_{vmi}^3$$

and the force against the vehicle pushbar exerted by such an override attempt is given by

$$F_{vm,i+1} = (40 - b_{PB}/2)d_{vmi}^3.$$

The force required to override all the vegetation in classes i and stem diameters is given by

$$F_{v,i+1} = 12w100[\text{sum of } {}_jd_{vj}^3 \text{ for } j=1 \text{ to } i]$$

where d_{vmi} = stem diameter of the largest stems in class i

d_{vj} = stem diameter of representative stem in class j

b_{PB} = push bar height of vehicle

w = width of vehicle

$_{j}$ = density of stems in class j.

These relationships may be found in Rula and Nuttall (1971) starting on page 157.

7. Subroutine IV7 - Driver Dependent Vehicle Vegetation Override Check

This routine determines the maximum stem diameter class which the driver will try to override. For each class not overridden an indicator is set as to whether it was driver tolerance or pushbar capacity that limited the override. The driver tolerance is based on longitudinal acceleration and is currently limited by 2 g's. Thus if for stem class i, $F_{vmi} > F_{mPB}$, the indicator will be set to no override [IMPACT(i)=1] due to pushbar weakness. If $F_{vmi}/GCW > 2.$, the indicator will be set to no override [IMPACT(i)=2] due to driver limit. If both limits are exceeded IMPACT(i)=3. The maximum stem diameter class to be overridden (indexed by MAXI) will be the largest stem class for which neither limit was exceeded.

8. Subroutine IV8 - Total Resistance Between Obstacles

In this routine the resistance to vehicle motion due to soil, slope and vegetation is summed to produce a resistance between obstacles. For each slope (up,level,down indexed by k) the resistance while overriding a single tree of stem diameter class i is given by

$$R_{T1ki} = GCW\sin\theta_k + (RTOWP*CGWP + RTOWNP*CGWNP)r_w\cos\theta_k + F_{v1i}$$

and the resistance while overriding all vegetation in stem diameter classes i-1 and smaller is

$$R_{Tki} = GCW\sin\theta_k + (RTOWP*GCWP + RTOWNP*GCWNP)r_w\cos\theta_k + F_{vi}$$

where GCW = gross combined weight

GCWP = gross combined weight on powered elements

GCWNP = gross combined weight on unpowered elements

θ_k = slope angle

RTOWP = powered elements resistance coefficient

RTOWNP = unpowered elements resistance coefficient

r_w = proportion of vehicle weight on running gear elements (=1 except possibly for water covered terrain units)

F_{v1i} = force required to override the largest tree in stem diameter class i-1

F_{vi} = force required to override vegetation in stem diameter classes i-1 and smaller.

9. Subroutine IV9 - Speed Limited by Resistance Between Obstacles

This routine uses Subroutine VELFOR (described as part of Subroutine IV5 above) to determine the maximum speed the vehicle could travel while overcoming the resistance R_{T1ki} and R_{Tki} calculated in Subroutine IV8.

If the resistance due to soil, slope and vegetation in a certain stem diameter class is larger than that which the vehicle is capable of overcoming at any speed, the velocity is set to zero and MAXI is lowered to reflect the fact that vegetation override will not be attempted for that class. In traverse mode (NTRAV=1) all velocities for slopes other than the designated slope are set to zero.

10. Subroutine IV10 - Speed Limited by Surface Roughness

Prior to runs of NRMM, a cross plot of speed vs. surface roughness was made from repeated runs of VEHDYN (See Chapter 1. Overview). Each of these cross-plots implies that, for the given surface roughness, the speed given is the maximum speed the vehicle can operate without subjecting the driver to vibrations exceeding a certain level of absorbed power. INRMM allows for several of these plots, giving results for several levels of absorbed power, to be entered as part of the vehicle data. The choice of which one to use is indicated by scenario variable LAC.

In this routine the speed vs. surface roughness array is searched and linearly interpolated for the maximum speed a vehicle could travel without subjecting the driver to vibrations resulting in an absorbed power greater than the limits implied by the choice of LAC.

11. Subroutine IV11 - Total Braking Force - Soil/Slope/Vehicle

This routine calculates the total braking force available to the vehicle. Two basic components are calculated. The first is the braking force due to the resistance of the running gear which is always present regardless of whether the brakes are on or not. This force is estimated as

$$X_1 = (RTOWB*GCWB + RTOWNB*GCWNB)r_w$$

where RTOWB = resistance coefficient of braked running gear elements

RTOWNB = resistance coefficient of unbraked running gear elements

GCWB = gross combined weight on braked running gear elements

GCWNB = gross combined weight on unbraked running gear elements

r_w = proportion of vehicle weight supported by running gear elements (=1 except possibly for water covered terrain units).

The other component calculated is due to the retardation force when the brakes are applied. The maximum retardation force available from the terrain unit surface material is calculated by

$$X_2 = (DOWB + RTOWB)GCWBr_w$$

where DOWB is the pull coefficient of the powered wheels. This force, adjusted for slope, is compared to the maximum force the vehicle can exert, X_{BR}, the lesser of the two being used. The total braking is the sum of these two components plus or minus the force due to gravity on slopes. Thus

$$T_{BFk} = GCWr_w\sin\theta_k + X_1\cos\theta_k + \min[X_{BR}, X_2\cos\theta_k]$$

where θ_k = slope angle.

If T_{BFk} is negative [due to $\sin\theta_k$ on downhill slopes] a no-go indicator is set , NOGOBF=1, under the supposition that the vehicle cannot be controlled without sufficient braking. Calculations are continued since the situation being simulated may not actually stop the vehicle, as for instance, on a long straight downhill slope for which the next terrain unit is flat and sufficiently bare of obstructions to allow run-out.

12. Subroutine IV12 - Maximum Braking Force - Soil/Slope/Vehicle/Driver

In this routine the maximum braking force calculated in the previous routine may be attenuated due to simulation of driver actions. It has been observed that drivers do not always use the maximum braking force available either due to choice or lack of skill. In addition drivers may not even use all of that braking force, preferring to always keep some in reserve for "safety's sake". These actions are modeled by two vehicle inputs:

DCLMAX = the maximum braking force (in g's) a driver will use due to comfort or skill (the "lunch box" limit)

SFTYPC = the percentage of the theoretical maximum the driver will actually use "for safety" (e.g. to prevent lockup of wheels).

If the user wishes no restriction on the performance of the vehicle due to driver imposed limits, the scenario variables DCLMAX and SFTYPC should be set to high values.

The braking used in further calculations is given by

$$B_{MXk} = \min[DCLMAX*GCW,\ T_{BFk}*SFTYPC/100]$$

where k = slope index (1 = up, 2 = level, 3 = down).

13. Subroutine IV13 - Speed Limited by Visibility

In this routine the maximum speed at which a driver may proceed to just stop within the visibility distance without exceeding the braking force (calculated in the previous routine) is calculated. The recognition distance is calculated by:

$$D_r = D_v\ h_e/60$$

where D_v = visibility distance

h_e = driver eye height.

The deceleration due to the braking force the driver will actually use is given by

$$a_{ck} = B_{MXk}\ g/GCW$$

where g = acceleration of gravity

k = slope index (1 = up, 2 = level, 3 = down)

B_{MXk} = maximum braking force actually used on slope k.

The maximum speed limited by visibility, V_{vk}, is calculated from the solution of the equation for recognition distance required to stop, which is

$$D_r = t_r V_{vk} + V_{vk}^2/(2B_{MXk})$$

where t_r = reaction time between recognition and application of brakes.

If the braking force B_{MXk} is nonpositive, the speed limited by visibility is set to zero. In addition, if V_{vk} is positive but less than VISMNV, the minimum speed the driver will accept despite full obscuration of his vision, then V_{vk} is set to VISMNV.

14. Subroutine IV14 - Selected Speed Between Obstacles

This routine chooses the minimum among the following speeds for travel between obstacles. This selection is made for each slope - vegetation override class.

$VTIRE_j$ = maximum safe tire speed at inflation pressure j

99% of $V_{SOIL,k,i}$ = velocity limited by soil, slope and vegetation resistance for slope k overriding vegetation in classes i-1 and smaller

V_{RID} = velocity limited by surface roughness

V_{vk} = velocity limited by visibility and braking force on slope k

A separate speed limit, V_{ttki}, is chosen for each slope k (1 = up, 2 = level, 3 = down) and overriding vegetation in class i-1 and smaller. Less than 100% of $V_{SOIL,k,i}$ is used since the velocity - distance curve has an asymptote at $V_{SOIL,k,i}$, which means that a vehicle can never accelerate up to $V_{SOIL,k,i}$.

15. Subroutine IV15 - Maximum Speed Between and Around Obstacles

This routine calculates, for each slope k and vegetation override index i, VBO_{ki}, the maximum speed the vehicle can achieve while traveling between obstacles, and $V_{AVOIDki}$, the speed which the vehicle can achieve in the terrain unit while avoiding all obstacles.

Between obstacles, VBO_{ki} represents the maximum speed a vehicle can travel while overriding vegetation in stem diameter classes i-1 and smaller while avoiding all vegetation in stem diameter classes i and larger by maneuvering. This maneuvering will lower the overall average speed due to path elongation and a variety of other factors. The extent to which this speed is lowered depends on PAV_i, the percentage of area denied due to maneuvering around vegetation in stem diameter classes i and greater. The relationships are:

Tracked Vehicles

for $PAV_i \leq 3\%$ no reduction ($VBO_{ki} = V_{ttki}$)

for $3\% < PAV_i \leq 7\%$

$$S_{MG} = [(392.93 - V_{ttki})PAV_i]/4 + (7V_{ttki} - 3*392.93)/4$$

$$VBO_{ki} = \min[S_{MG}, V_{ttki}]$$

for $7\% < PAV_i \leq 52.5\%$

$$S_{MG} = 453.2 - 8.603PAV_i$$

$$VBO_{ki} = \min[S_{MG}, V_{ttki}]$$

for $52.5\% < PAV_i$ set $VBO_{ki} = 0$

Wheeled Vehicles

for $PAV_i \leq 3\%$ no reduction ($VBO_{ki} = V_{ttki}$)

for $3\% < PAV_i \leq 7\%$

$$S_{MG} = [(450.33 - V_{ttki})PAV_i]/4 + (7V_{ttki} - 3*450.33)/4$$

$$VBO_{ki} = \min [S_{MG}, V_{ttki}]$$

for $7\% < PAV_i \leq 41.3\%$

$$S_{MG} = 542.11 - 13.112PAV_i$$

$$VBO_{ki} = \min [S_{MG}, V_{ttki}]$$

for $41.3\% < PAV_i$ set $VBO_{ki} = 0$

Around and between obstacles, $VAVOID_{ki}$ represents the overall maximum speed the vehicle can travel while overriding vegetation in stem diameter classes i-1 and smaller and avoiding both obstacles and vegetation in stem diameter classes i and greater. The equations used to calculate $VAVOID_{ki}$ are similar to those used above to calculate VBO_{ki} except that ADT_i, the percentage area denied by avoiding obstacles and vegetation in stem diameter classes i and greater, is used in place of PAV_i.

$$VAVOID_{k,i} \quad \text{for } k = 1,2,3 \quad \text{if NTRAV} = 3$$
$$\text{or } k = 1 \quad \text{if NTRAV} = 1$$
$$i = 1,\ldots,NI$$

is the complete set of speeds calculated for driving strategy which avoids obstacles. The next comparable set of speeds is $VOVER_{ki}$, which simulates the driving strategy that overrides obstacles. From these two sets of speeds the overall terrain unit speed is selected with some modifications in the last two routines of the Areal Module, Subroutines IV20 and IV21.

16. Subroutine IV16 - Obstacle Override Interference and Resistance

This routine uses a simplistic three dimensional, linear interpolation routine, subroutine D3LINC, on the arrays developed from the Obstacle Module, OBS78B. prior to execution of the Operational Modules of the INRMM. Three tables were developed from the Obstacle Module

i) FOO vs OBH, OBAA, and OBW

ii) F_{OOMAX} vs OBH, OBAA, and OBW

iii) CLEAR vs OBH, OBAA, and OBW

where OBH = obstacle height (denoted by h_o in TPP)

OBAA = obstacle approach angle (denoted by α_o in TPP)

OBW = obstacle base width (denoted by W_{og} in TPP)

FOO = overall tractive effort to override obstacle

F_{OOMAX} = maximum tractive effort required during override of obstacle

CLEAR = minimum clearance/maximum interference during override of obstacle

The outputs of this routine are FOM, FOMMAX, and CLR, which are the overall tractive effort, maximum tractive effort, and clearance/interference for overriding obstacles in the current terrain unit respectively.

If this routine determines that vehicle/obstacle interference occurs, the flag NEVERO is set to 3.

17. Subroutine IV17 - Driver Dependent Vehicle Speed Over Obstacles

The routine linearly interpolates two arrays developed from the Ride Dynamics Module, VEHDYN, prior to execution of the Operational Modules of the INRMM. These tables give

i) VOOB vs OBH

ii) VOOBS vs OBSE

where OBH = obstacle height (denoted by h_o in TPP)

OBSE = effective obstacle spacing (calculated in subroutine IV1)

VOOB = maximum constant forward velocity during override of an obstacle of height h_o which results in a vertical acceleration on the driver (or some other critical location) limited by a certain g level. Currently 2.5g at the driver's station is used.

VOOBS = maximum constant forward speed during the override of obstacles whose spacing is OBSE. This maximum speed is limited by both the absorbed power criterion and the vertical acceleration criterion on the driver (or some other critical location on the vehicle).

The relationship between VOOBS and OBSE is used here without regard for obstacle height. Current practice is to generate this relationship using VEHDYN with a constant obstacle height for all spacings. The height used should be that which limits the vehicle speed to 15 mph. as determined from the single obstacle relation.

The logic of the routine is that if the average distance between obstacles (OBSE - WA) is less than the wheel base or track length on the ground, only the second array (VOOBS vs. OBSE) is interpolated and the resulting speed is used for VOLA, the maximum obstacle approach speed.

If the distance between obstacles is between the wheelbase or track length on the ground and twice the full wheelbase or track length both arrays are interpolated and the lesser of the two speeds is used for VOLA. If the distance between obstacles is greater than twice the wheelbase or track length on the ground then only the first array (VOOB vs. OBH) is interpolated and the resulting speed is used for VOLA.

18. Subroutine IV18 - Speed Onto and Off Obstacles

This routine determines the maximum approach speed, VA_{ki}, at the first contact with an obstacle and the exit speed, VXT_{ki}, when departing the obstacle after overriding. Possible adjustments to the speed between obstacles, VBO_{ki}, are also made. If the prior routines of the Areal Module have determined that there cannot be override or there can be no gain in speed due to override, an indicator NEVERO is set to 1,2 or 3. In this case the routine sets all the speeds $VA_{ki} = VXT_{ki} = VBO_{ki} = 0$.

If the obstacles are so closely spaced ($TL \geq OBSE - WA$) that the vehicle essentially is always in contact with one of them then an

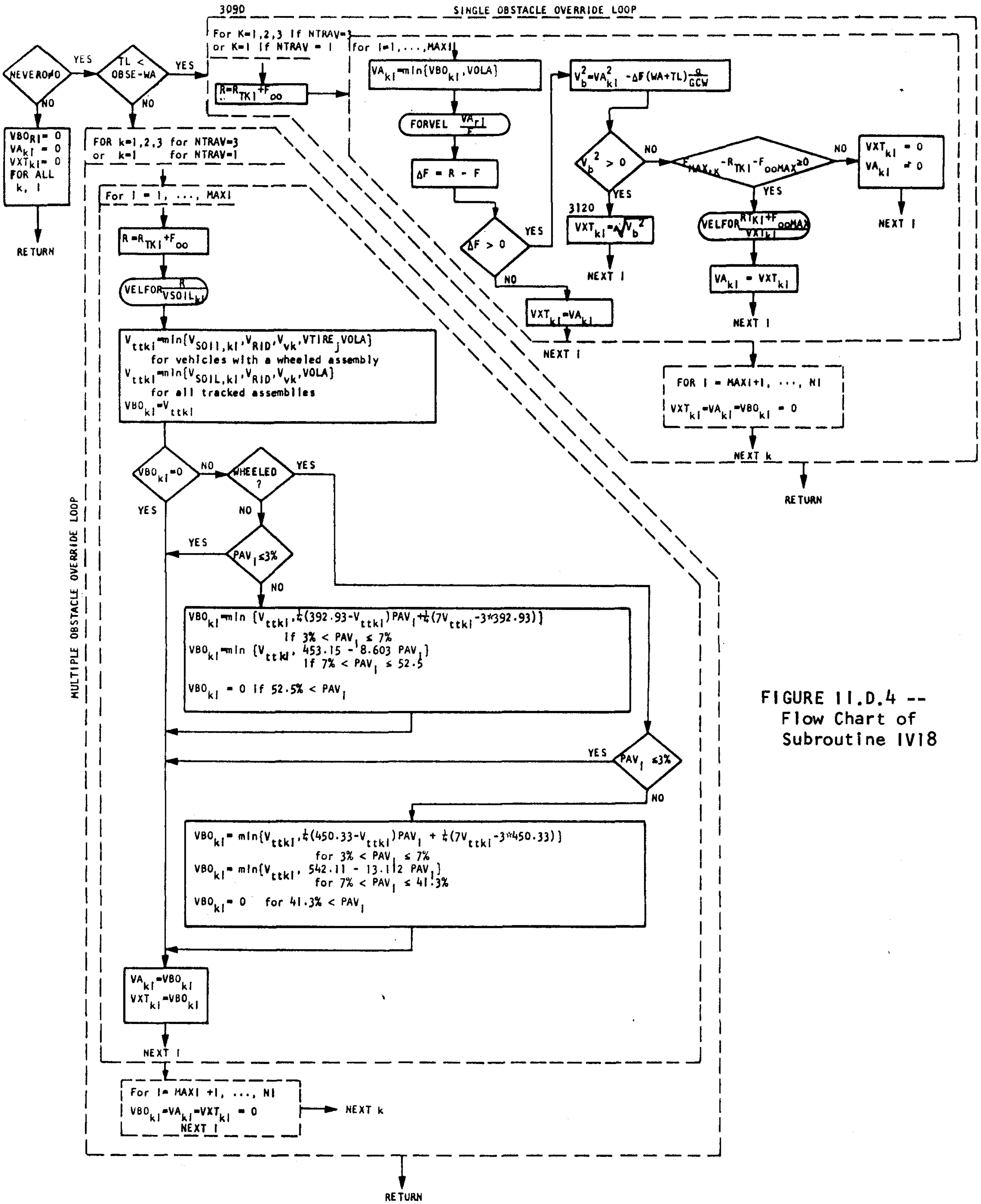


FIGURE II.D.4 --
Flow Chart of
Subroutine IV18

assumption is made that the driver will choose to proceed at a steady speed over the terrain unit (i.e., no acceleration/deceleration between obstacles). This speed, V_{ttki}, is determined to be the maximum speed limited by the driver's comfort level overriding obstacles (VOLA), the general roughness level (V_{RID}), the visibility/braking force limit (V_{vk}), the tire limit ($VTIRE_j$) if there are wheeled suspension elements, and the soil/vegetation/ slope/obstacle resistance limit ($VSOIL_{ki}$) calculated by VELFOR, a subroutine described above in Section II.D.5.b). The resistance used here is that of the soil/vegetation/slope for the terrain unit, plus that resistance due to overriding the obstacle, FOM. The speed "between obstacles", VBO_{ki} is then calculated from V_{ttki} by attenuating V_{ttki} due to possible maneuvering to avoid vegetation in stem diameter classes i and greater. This calculation is the same as that described in Subroutine IV15 above (Section II.D.15)). Then the approach and exit speeds are all set to the same value as the speed between obstacles ($VA_{ki} = VXT_{ki} = VBO_{ki}$). This will guarantee that no acceleration/deceleration can occur since no speed changes across and between obstacles will occur.

For the case where the vehicle can fit entirely between obstacles (TL < OBSE - WA) it is theoretically possible that the vehicle could accelerate to speed VBO_{ki} or some lesser speed after leaving the obstacle at speed VXT_{ki} before braking in order to reduce speed to VA_{ki} when approaching the next obstacle. In this case the three speeds, VA_{ki}, VXT_{ki}, and VBO_{ki} are calculated as follows.

The resistance of soil and slope is given by R_{Tk1}, the soil/slope/vegetation resistance avoiding vegetation in classes 1,2,...,NI. The use of this resistance makes the assumption that obstacles are bare of vegetation.

For each vegetation class i and slope class k the approach speed, VA_{ki}, is set equal to the lesser of the soil/slope/vegetation limited speed (V_{ttki}) and the human tolerance limited obstacle impact speed (VOLA). A subroutine, called FORVEL, is used to evaluate the tractive effort, F, available at speed VA_{ki}. This subroutine searches the various speed ranges or "gears" and then evaluates the appropriate quadratic. A force deficit is calculated by

$$\Delta F = R_{Tk1} + F_{OO} - F.$$

If $\Delta F \leq 0$ then there is enough tractive effort available to overcome both soil/slope and obstacle resistance hence no speed is lost in crossing the obstacle. In this case $VXT_{ki} = VA_{ki}$.

If $\Delta F > 0$, there is not enough tractive effort available. A check is made to see if the obstacle can be overcome by some of the kinetic energy of the vehicle. A terminal speed exiting the obstacle is calculated from

$$V_b^2 = {VA_{ki}}^2 - \Delta F(WA + TL)g/GCW$$

where WA = obstacle width

TL = wheel base or track length on ground

g = acceleration of gravity

GCW = gross combination weight.

If $V_b^2 \geq 0$, there is sufficient kinetic energy available to override the obstacle and the exiting speed is $VXT_{ki} = V_b$.

If $V_b^2 < 0$, there is not sufficient energy available to override the obstacle when approaching it at VA_{ki}. Since there usually is more tractive effort available at lower speeds, a test is made to see if the obstacle can be overriden at any speed by comparing $F_{MAX,k}$ to $R_{Tk1} + F_{OOMAX}$, where

$F_{MAX,k}$ = maximum tractive effort available

F_{OOMAX} = maximum tractive effort required during obstacle override

If $F_{MAX,k} < R_{Tk1} + F_{OOMAX}$, override is not possible and $VA_{ki} = VXT_{ki} = 0$.

If $F_{MAX,k} > R_{Tk1} + F_{OOMAX}$, override is possible and the assumption is made that the vehicle will cross the entire obstacle at the speed, V_{FMAX}, which yields the maximum tractive effort. This is calculated by Subroutine VELFOR. Then $VA_{ki} = VXT_{ki} = V_{FMAX}$.

19. Subroutine IV19 - Average Terrain Unit Speed While Accelerating/Decelerating Between Obstacles

This routine calculates the time it takes for the vehicle to traverse various portions of the terrain unit and the distance it travels during this time. From this time and distance calculation an average speed, $VOVER_{ki}$, is calculated.

The various times and portions are specified by

T_a, X_a = time and distance during acceleration from velocity VXT_{ki} after leaving an obstacle

T_{BO}, X_{BO} = time and distance during constant velocity travel at speed VBO_{ki} between obstacles

T_b, X_b = time and distance during deceleration (braking) from VBO_{ki} (or some lesser speed if $T_{BO} = X_{BO} = 0$) to VA_{ki}, the maximum obstacle approach speed

T_{OO}, X_{OO} = time and distance crossing the obstacle.

If VBO_{ki} cannot be reached before braking must begin then $T_{BO} = X_{BO} = 0$. The various possibilities are indicated in Figure II.D.5.

The routine makes use of three other subroutines:

ACCEL - which calculates the time and distance to accelerate from one speed, V_a, to another $V_b > V_a$. If speed V_b cannot be reached by acceleration an error flag, NV2FLG, is set.

TXGEAR - which is called by ACCEL and calculates the time and distance required in one gear during acceleration

VELFOR - the maximum speed achievable while overcoming a given resistance (described as part of Subroutine IV5 above)

The above times and distances are calculated as follows:

$$X_{OO} = WA + TL$$

$$T_{OO} = 2X_{OO}/(VA_{ki} + VXT_{ki})$$

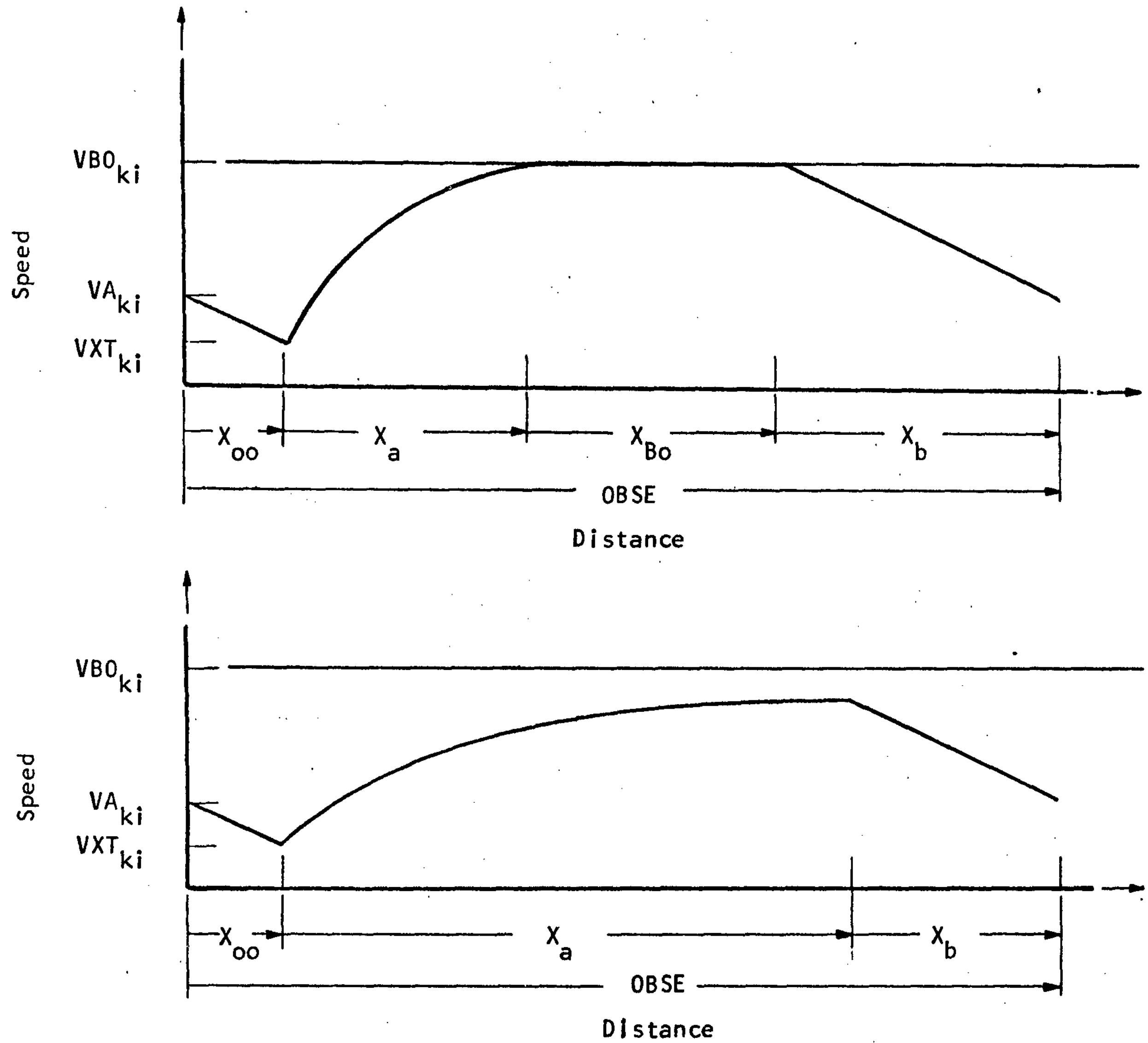


FIGURE II.D.5 -- Possible Speed Profiles Across an Obstacle and Between Obstacles

where WA = obstacle width

TL = wheelbase or track width on ground

$$X_{BO} = OBSE - X_{OO} - X_a - X_b$$

where OBSE = effective obstacle spacing

$$T_{BO} = X_{BO}/VBO_{ki} \qquad \text{if} \quad X_{BO} \geq 0$$

$T_{BO} = 0 \qquad \text{if} \quad X_{BO} < 0$

For any speed $V_M > VA_{ki}$,

$$T_b = [M_v(V_M - VA_{ki})]/B_{MXK}$$

where M_v = vehicle mass

B_{MXK} = maximum braking force

and

$$X_b = (V_M + VA_{ki})T_b/2$$

The overall average terrain unit speed while crossing obstacles is then given by

$$VOVER_{ki} = OBSE/(T_{OO} + T_a + T_b + T_{BO})$$

for slope k = 1 up, 2 level, 3 down if NTRAV = 3

k = 1 if NTRAV = 1

while overriding vegetation in stem diameter class i-1 and smaller and avoiding vegetation in stem diameter class i and larger. These relationships can all be calculated if X_a, T_a, and the final speed after acceleration, denoted above by V_M, are known.

Several initial checks are made. If $VXT_{ki} = VBO_{ki}$, then the speed between, onto and off the obstacle are all the same and the overall terrain unit speed crossing obstacles is $VOVER_{ki} = VBO_{ki}$. If $VA_{ki} = VXT_{ki} = 0$ then obstacles cannot be crossed and $VOVER_{ki} = 0$.

If $VXT_{ki} < VA_{ki}$, Subroutine ACCEL is called to determine if the vehicle can accelerate from $V_a = VXT_{ki}$ to $V_b = VA_{ki}$; that is, can the vehicle when leaving an obstacle at speed VXT_{ki} accelerate up to the maximum approach speed VA_{ki} in the distance between obstacles? If the distance to do so, X_a, is greater than the space between obstacles, OBSE - WA -TL, or the flag NV2FLG is set, it means the vehicle cannot accelerate to the approach speed VA_{ki} and VELFOR is called to determine if there is any speed at which the vehicle can overcome the soil/slope/vegetation and obstacle resistance, given by $R_{Tk1} + F_{OOMAX}$ (see Figure II.D.6 for the speed profile). If such an override speed exixts, $VOVER_{ki}$ is set to this override speed; otherwise $VOVER_{ki} = 0$.

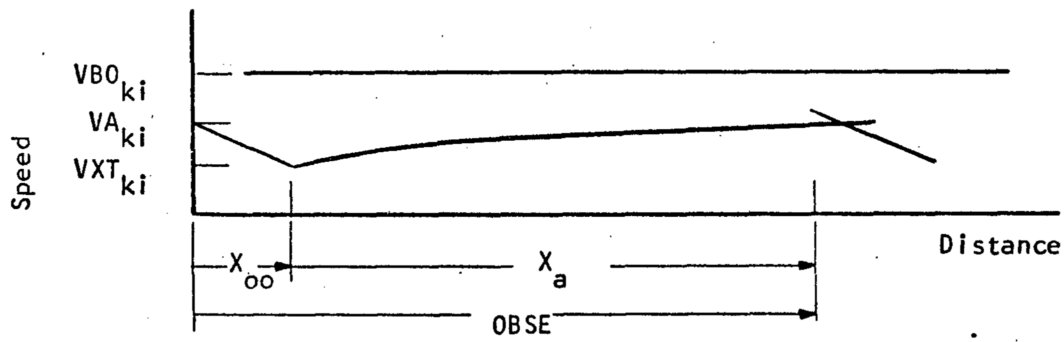


FIGURE II.D.6 -- Speed Profile when Obstacle Approach Speed cannot be Attained

Once it is known that there is enough distance between obstacles and/or the vehicle has enough excess traction to accelerate at least up to the approach speed of the next obstacle, Subroutine ACCEL is called to determine if the vehicle can accelerate to VBO_{ki}, the maximum speed between obstacles, and if so, the distance, X_a, and time, T_a, required. The time, T_b, and distance, X_b, to brake from VBO_{ki} to VA_{ki} are also calculated and if the sum of X_a and X_b is less than the space between obstacles, OBSE - WA - TL, then $VOVER_{ki}$ is calculated from formulas at the beginning of this section. (See Figure II.D.5 for the speed profile of this case.)

If the distance, X_a, to reach VBO_{ki} plus the distance, X_b, to brake back down to VA_{ki} exceeds the distance between obstacles, then the speed VBO_{ki} can never be reached between obstacles before braking has to begin. The lower speed profile of Figure II.D.5 applies to this case. The distance-speed coordinates of the point B have to be determined. The speed coordinate, to be called V_M, must have a value in the interval from VA_{ki} and VBO_{ki} and the distance coordinate, $X_{OO} + X_a$, must have a value between X_{OO} and OBSE $-X_b$. Actually only one number among V_M, X_a, and X_b needs to be determined, since the others can be found from it and other known values.

V_M is the value sought. Successive approximations to V_M are postulated by a binary search within the speed interval from VA_{ki} to VBO_{ki}. For each such postulated value of V_M, the distance X_a to accelerate to V_M from VXT_{ki} and the distance X_b to decelerate from V_M to VA_{ki} are calculated. If $X_{OO} + X_a + X_b > \text{OBSE}$ then V_M is adjusted to a lower value; if $X_{OO} + X_a + X_b < \text{OBSE}$ then V_M is adjusted to a higher value. Ten such adjustment are made (corresponding to a speed precision of 2^{-10} of the difference between VBO_{ki} and VA_{ki}).

The distance coordinate is highly sensitive with respect to the final speed before braking, V_M, in the sense that a small difference in v_M can lead to a large difference in $X_{OO} + X_a$. As a consequence, it may result that even though the speed precision is as stated above, after 10 iterations the distance $X_{OO} + X_a + X_b$ is still significantly different from OBSE.

If the final distance is larger than OBSE, VBO_{ki} is reduced by 1 MPH decrements and the entire search for V_M is repeated. If the distance is smaller, no corrective action is taken since the result is that there will be a distance between obstacle at which the vehicle will travel at the constant speed V_M. Any error caused by this are considered negligible.

A value of $VOVER_{ki}$ is calculated for every combination of slope k (up, level, down or traverse) and vegetation override/avoid strategy i; for vegetation classes and/or obstacles which cannot be overriden $VOVER_{ki} = 0$.

a) Subroutine ACCEL - Time and Distance to Accelerate from One Velocity to Another

This routine calculates the time, T, and the distance, X, required for the vehicle to accelerate from one speed, V_1, to another $V_2 \geq V_1$. From the tractive effort vs speed curve, the "gear" or speed range, n_{g1}, of the initial speed V_1 is found ($V_{n_{g1},1} \leq V_1 \leq V_{n_{g1},3}$). Similarly, the gear, n_{g2} of the final speed, V_2 is found ($V_{n_{g2},1} \leq V_2 \leq V_{n_{g2},3}$).

If $n_{g1} < n_{g2}$, the Subroutine TXGEAR is called repeatedly to calculate the time and distance in each gear from n_{g1} to n_{g2}. If, within any gear, the vehicle cannot accelerate to the final speed of the gear, V_{n3}, or the final speed to be reached, V_2, an error flag, NV2FLG, is set and the highest speed within that gear that the vehicle can achieve is calculated by a Binary search.

b) Subroutine TXGEAR - Time and Distance in a Gear

In this routine the quadratic functions representing tractive effort for vehicle speed are integrated to yield time and distance in each gear.

For illustrative purposes, a single gear, or speed range representing a fixed set of quadratic coefficients is shown on Figure II.D.7.

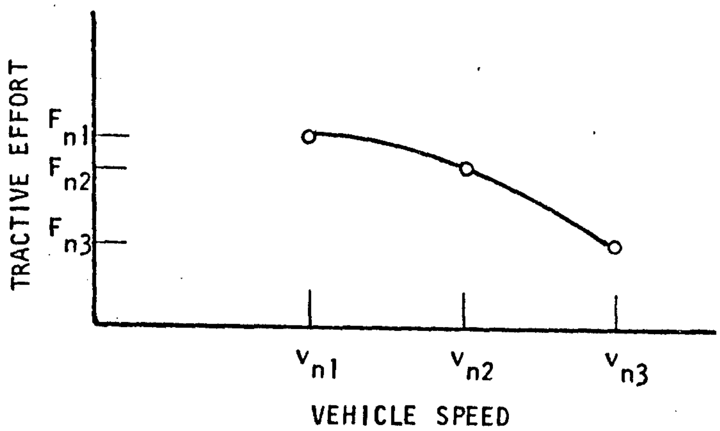


FIGURE II.D.7 -- Representation of Tractive Effort vs. Speed Relation

For this gear, the relationship is given by

$$F = a_n + b_n v + c_n v^2$$

for $v_{n1} \leq v \leq v_{n3}$. Similar sets of constants (a_n, b_n, c_n) were calculated in Subroutine IV5 for all gears n=1,2,. . . ,NGR.

The differential equation to be solved is given by

$$M \dot{v} = F - R_{Tki}$$

where M = vehicle mass

R_{Tki} = soil/slope/vegetation resistance to be overcome.

The excess force $F - R_{Tki}$ is the tractive effort being used to accelerate from v_1 to v_2 within the gear n. Usually $v_1 = v_{n1}$ and $v_2 = v_{n3}$ but it is possible for $v_{n1} \leq v_1 \leq v_2 \leq v_{n3}$.

The above differential equation is separated and integrated thus

$$I = \int_{v_1}^{v_2} (1/[c_n v^2 + b_n v + (a_n - R_{Tki})])dv$$

$$= \int_0^t (1/M)dt = t/M$$

If the discriminant $d_n^2 = b_n^2 - 4c_n(a_n - R_{Tki})$, then the time to accelerate from v_1 to v_2 in gear r is given by

if $d_n^2 < 0$

$$t = 2M/(-d_n^2)^{1/2} \arctan[(2c_n v + b_n)/(-d_n^2)^{1/2}] \Big|_{v1}^{v_2}$$

if $d_n^2 = 0$

$$t = -2M/(2c_n v + b_n) \Big|_{v_1}^{v_2}$$

if $\quad d_n^2 > 0$ and $d_n = +[d_n^2]^{1/2}$

$$t = M/(d_n^2)^{1/2} \ln[(2c_nv+b_n-d_n)/(2c_nv+b_n+d_n)] \Big|_{v_1}^{v_2}$$

These relationships may be read from tables of integrals, e.g., Abramowitz and Stegun (1965).

In the cases $d_n^2 > 0$, the distance to accelerate from v_1 to v_2 can be calculated directly. For $d_n^2 > 0$, the equation for t can be solved for v_2 to yield

$$v_2(t) = 1/2c_n[2d_n/(1 - v_1e^{td_n/M}) - b_n - d_n]$$

and this integrated to yield distance thus

$$X(t) = \int_0^t v_2(t)\,dt$$

$$= 1/c_n[(d_n - b_n)t/2 + M \ln((1 - v_1\, e^{td_n/M})/(1 - v_1))]$$

Similarly for $d_n^2 = 0$

$$v_2(t) = 1/2c_n[2M/(2M/(2c_nv_1 + b_n) - t) - b_n]$$

and

$$X(t) = \int_0^t v_2(t)\,dt$$

$$= (M/c_n)\ln[2M/(2M - t(2c_nv1 + b_n))] - b_nt/2c_n$$

In the case $d_n^2 < 0$, the equation for t can be solved for $v_2(t)$ in terms of v_1 but this formula cannot be integrated in closed form to yield X(t). In this case the gear is approximated by two straight lines fitted from (v_{n1},F_{n1}) to (v_{n2},F_{n2}) and from (v_{n2},F_{n2}) to (v_{n3},F_{n3}). For each of these lines the integral equation

$$\int_{v_1}^{v_2} [1/(bv + (a - R_{Tki}))]dv = \int_0^t [1/M]dt$$

(where the coefficients a and b stand for either set) is solved to yield

$$t = (M/b)\ln[(bv_2 + (a - R_{Tki}))/(bv_1 + (a - R_{Tki}))]$$

$$v_2(t) = (v_1 + (a - R_{Tki})/b)e^{bt/M} - (a - R_{Tki})/b$$

$$X(t) = \int_0^t v_2(t)dt$$

$$= (1/b^2)[(bv1 + a - R_{Tki})M(e^{bt/M} - 1) - (a - R_{Tki})bt]$$

These latter relationships are also used in the case where the quadratic formula for a gear really was a straight line, namely $c_n = 0$.

The only other case for which formulas are neeeded is the case where both $c_n = b_n = 0$. Here the integral equation is

$$\int_{v_1}^{v_2}[1/(a_n - R_{Tki})]dv = \int_0^t [1/M]dt$$

and

$$t = M[(v_2 - v_1)/(a_n - R_{Tki})]$$

$$X(t) = [(a_n - R_{Tki})t^2]/2M + v_1t$$

The various formulas above can, of course, yield negative values if applied without concern as to whether the vehicle can really accelerate from v_1 all the way to v_2 in gear r, and in fact whether the vehicle can accelerate beyond v_1 at all. NV2FLG = 1 will be used to represent the case where the vehicle cannot accelerate beyond v_1 and NV2FLG = 2 will be used to denote the case where v_2 cannot be reached. These questions can be resolved by considering the locations of the two roots of

$$F = c_n v^2 + b_n v + (a_n - R_{Tki}) = 0.$$

As long as the quadratic on the left hand side represents positive values of tractive effort within the domain $v_1 \leq v \leq v_2$, there is excess tractive effort available for acceleration. To determine whether the quadratic is positive within $v_1 \leq v \leq v_2$ the coefficients can be tested; 27 combinations of c_n, b_n, and $(a_n - R_{Tki})$ being positive , zero or negative can result. Let $R_2 = (-b_n + d_n^2)/2c_n$ be the larger root and $R_1 = (a_n - R_{Tki})/(c_n R_2)$ be the smaller. Then NV2FLG = 1 will result from

$c_n > 0,\ b_n < 0,\ a_n - R_{Tki} \leq 0$ and $v_1 \leq R_2$

$c_n > 0,\ b_n < 0,\ a_n - R_{Tki} > 0$
and $v_1 > R_1$ for $d_n^2 > 0$
or $v_1 = R_1$ for $d_n^2 = 0$

$c_n > 0,\ b_n = 0,\ a_n - R_{Tki} < 0$ and $v_1 \leq R_2$

$c_n > 0,\ b_n > 0,\ a_n - R_{Tki} \leq 0$ and $v_1 \leq R_2$

$c_n \leq 0,\ b_n \leq 0,\ a_n - R_{Tki} \leq 0$ all cases

$c_n \leq 0,\ b_n < 0,\ a_n - R_{Tki} > 0$ and $v_1 \geq R_1$

$c_n = 0,\ b_n > 0,\ a_n - R_{Tki} < 0$ and $v_1 \leq R_1$

$c_n = 0,\ b_n > 0,\ a_n - R_{Tki} = 0$ and $v_1 \leq 0$

$c_n < 0,\ b_n \leq 0,\ a_n - R_{Tki} > 0$ and $v_1 \geq R_2$

$c_n < 0,\ b_n > 0,\ a_n - R_{Tki} < 0$
and $v_1 \leq R_1$ or $v_1 \geq R_2$.

NV2FLG = 2 will result from

$c_n > 0$, $b_n < 0$, $a_n - R_{Tki} > 0$
and $v_2 > R_1$ or $v_1 < R_2$
but not $v_1 > R_1$ when $d_n^2 > 0$
or $v_2 = R_1$ when $d_n^2 = 0$

$c_n = 0$, $b_n < 0$, $a_n - R_{Tki} > 0$ and $v_2 \geq R_1$

$c_n < 0$, $b_n \leq 0$, $a_n - R_{Tki} > 0$ and $v_2 \geq R_2$

$c_n < 0$, $b_n > 0$, $a_n - R_{Tki} < 0$ and $v2 \geq R_2$.

Subroutine TXGEAR performs a decision tree on c_n, b_n, d_n^2, R_1 and R_2 to determine if the flag NV2FLG needs to be 0,1 or 2 and then uses the appropriate formulas for t and X(t) to calculate the time and distance in gear n.

20. Subroutine IV20 - Kinematic Vegetation Override Check

Various prior subroutines have calculated speeds limited by various factors such as soil/slope/vegetation resistance, ride roughness, obstacle resistance, maneuvering to avoid obstacles and vegetation, etc. As each new factor was considered the possibility arose that the prior maximum speed for a given slope (up, level, down or traverse) and vegetation override/avoid strategy the speed first calculated on the basis of soil/slope/vegetation resistance had to be reduced. A question now arises if at the final speeds $VAVOID_{ki}$ and $VOVER_{ki}$ there is enough excess traction and kinetic energy to still override vegetation in the stem diameter classes i-1 and smaller. This routine performs this test as follows:

Subroutine FORVEL is called to calculate the excess traction available at speed $VOVER_{ki}$ which is added to the kinetic energy GCW $(VOVER_{ki})^2/2g$, where GCW is the gross combination weight and g is the acceleration of gravity. If this sum is less than the force needed to override the largest stem diameter in class i-1, an indicator $NOGOVO_{ki}$ is reset. If the force is sufficient, $NOGOVO_{ki}$ = 1. Similarly, $NOGOVA_{ki} = 1$ if the excess tractive effort plus the kinetic energy available at speed $VAVOID_{ki}$ is sufficient to override the largest stem in vegetation class i-1; otherwise $NOGOVA_{ki} = 0$.

21. Subroutine IV21 - Maximum Average Speed

In this routine a speed, from all the $VAVOID_{ki}$ and $VOVER_{ki}$, is selected according to several criteria which include:

1. for each k, select the maximum of all $VAVOID_{ki}$ and $VOVER_{ki}$
2. if the maximum speed is less than VWALK,the vehicle will proceed at that maximum speed.
3. if the maximum is greater than VWALK and the resulting acceleration due to overriding vegetation is less than one g and the stem diameter class i is less than that designated by IOVER, then proceed at the maximum
4. if the maximum is greater than VWALK but i > IOVER or the acceleration is greater than g, continue to reduce the stem diameter class to be overriden until the criteria of 3) are met.

After the above are selected for each slope (up, level, down, or traverse) the final speeds calculated are the harmonic average of the individual speeds in each of the slopes. [If any of these individual speeds is zero, the harmonic average is set to zero.] Two speeds are reported by this routine, the maximum speed (VMAX) regardless of vegetation override and the maximum speed (VSEL) which keeps the acceleration below one g and does not override stems in class IOVER or greater.

E. Road Module

The Road Module of the INRMM calculates the maximum speed a vehicle can travel if the terrain unit is a road or a trail. Roads in the INRMM are terrain units characterized by a non-yielding surface with a coefficient of friction and a surface condition factor as well as curvature, superelevation, roughness and slope. There are no obstacles or vegetation on roads. This Module is also used for trails, which are terrain units characterized by yielding soils but otherwise are similar to roads.

For both roads and trails it is assumed that each terrain unit contains sections for up and down or just level or up or down travel. In rolling terrain there are seldom stretches of road that would be classified as distinct terrain units that contain both up/down and level parts.

The major portions of this Module calculate aerodynamic, rolling, cornering and grade resistance, and from this find a speed limited by these resistances. This speed is compared to that limited by ride roughness, sliding and tipping on curves, and a braking/visibility limit. An overall curvature speed limit derived from tests conducted by the American Association of State Highway Officials (AASHO) is applied before a final maximum speed is selected.

These portions are comparable to various subroutines in the Areal Module. The Road Module, being shorter, was coded as a single subroutine with calls to appropriate subroutines of the Areal Module for the yielding soil. These individual sections of the Road Module will now be described.

1. Initialization

Various constants are set, some of which are needed only for compatability with the subroutines of the Areal Module used for trails.

2. Velocity Dependent Resistance

In this section the surface resistance, aerodynamic drag and turning resistance are calculated.

a) Surface Resistance

This calculation differs between roads and trails. For trails Subroutines IV3, IV4, and IV5 of the Areal Module are called to calculate

RTOWxx - the resistance coefficient for powered (RTOWP) non-powered (RTOWNP), braked (RTOWB), non-braked (RTOWNB), powered and braked (RTOWPB), and towed (RTOWT) running gear assemblies,

DOWxx - the pull coefficients of powered (DOWP), braked (DOWB), and powered and braked (DOWPB) running gear assemblies, and

Tractive effort vs. velocity relationship adjusted for the slip of the running gear. this relationship is given as a series of speed intervals (gears) for each of which a quadratic curve ($F = c_n v^2 + b_n v + a_n$) represents the relationship.

See the description of these routines in Section II.D above.

For hard surfaces, primary and secondary roads, resistances are calculated for each of the five velocity, force points of each gear and subtracted from the tractive effort. (See description of Subroutine IV5 in Section II.D.5). A new set of quadratics are then fitted, by Subroutine QUAD5, to yield a resistance modified tractive effort vs. speed relationship.

Initially the tractive effort, adjusted for altitude by the elevation correction factor ECF, is limited by the traction available from the surface given by

$$T_s = \mu_j \text{ GCWP } \cos\theta_k$$

where μ_j = coefficient of friction for surface condition factor j

GCWP = gross combination weight on powered running gear assemblies

θ_k = slope angle.

If T_s is less than tractive effort (rim pull curve) the tractive effort vs. speed curve is modeled as a horizontal line at $F = T_s$ until such speed, v*, that the rim pull curve is exactly equal to T_s. The gears are so adjusted that the first gear is a horizontal line at T_s for speed v such that $0 \leq v \leq v^*$ and the next gear begins at $v_{n1} = v^*$. Five new (speed, tractive effort) points are interpolated from 0 to v^* and v^* to v_{n5}.

b) Aerodynamic Resistance

After adjustment for surface traction limit, the aerodynamic drag at each velocity, v_{ni}, in each gear is calculated from

$$F_{AD} = .0026\ C_D\ A_F\ v_{ni}^2$$

adjusted so that the units match

where C_D = drag coefficient

A_F = frontal area. (ft.^{2})

c) Wheeled Axle Turning Resistance

First the superelevation factor is calculated from

$$e = 1 - 14.95 R_c e_a / 12^*\ v_{ni}^2$$

where R_c = radius of curvature

e_a = superelevation angle.

The turning resistance for any wheeled assemblies is speed dependent and calculated by

$$F_{CC} = (e/n_i C_\alpha)[W_i \cos\theta_K\ v_{ni}^2/111 R_c]^2$$

where i = running gear (wheeled) index

R_c = radius of curvature

W_i = weight on axle i

n_i = number of wheels on axle i

C_α = average cornering coefficient (slope at slip angle = 0 in lateral force vs. slip angle relationship,Lbs./Deg.). The units of C_α do not follow the INRMM standards in this formula. Furthermore, the constant 111, which comes from the formula in Smith (1970), implies that v_{ni} is in miles per hour. The code in NRMM makes a final adjustment to inches per second. Only terms for wheeled assemblies are included in the sum for F_{CC}. These two resistances are subtracted from the tractive effort vs. speed relationship and a new set of quadratics are fitted to the adjusted (speed, tractive effort-resistance) points by Subroutine QUAD5.

d) Tandem Wheel Aligning Resistance

This resistance is calculated by

$$F_{TC} = (e\,\mu_j/2R_c)\sum_i (W_i + W_{i+1})b_i$$

where i = index of front axle of a tandem pair

b_i = spacing between tandem axles.

3. Non-velocity Dependent Resistance

In this category are turning resistance on tracked assemblies and rolling and grade resistance.

a) Tracked Turning Resistance

The resistance of tracked assemblies to turns is given by the Merritt equation (Merritt (1946) or Ray (1970)) in terms of the width to length ratio

$$\alpha_i = t_i/l_i$$

where t_i = track of tracked assembly i

l_i = length of track on ground of assembly i

A "Merritt constant" is calculated as

$$M_{ki} = 1.0624 - .6999\alpha_i + .051848\alpha_i^2 + .05488\alpha_i^3$$

and a radius factor as

$$K_{1i} = M_{ki}(1.18 - .0090895\ R_c/12. + .00003779(R_c/12.)^2 + 6.70476*10^{-8}(R_c/12)^3)$$

The turning resistance is then calculated by

$$F_{CT} = TFOR/GCW \sum_i K_{1i}W_i + F_{CC} \quad \text{for trails}$$

$$= \mu_j \sum_i K_{1i}W_i + F_{CC} \quad \text{for roads}$$

where TFOR = maximum tractive effort available from surface

GCW = gross vehicle weight.

b) Rolling Resistance

The rolling resistance for trails (soft surface) is given by

$$F_R = S(RTOWP*GCWP + RTOWNP*GCWNP)\cos\theta_k$$

where GCWNP = gross combination weight on non-powered wheels

S = surface condition factor.

For roads (hard surface) the rolling resistance is calculated separately for each running gear assembly i by

$$F_{Ri} = (.007 + .0939/psi)W_iS \quad \text{for wheeled assemblies}$$

$$= .045W_iS \quad \text{for tracked assemblies}$$

where psi = pressure in the tires on axle i.

The total rolling resistance is $F_R = \sum F_{Ri}$.

c) Grade Resistance

The grade resistance is simply

$$F_{Gk} = GCW\sin\theta_k$$

for each slope angle θ_k.

4. Speed Limited by Resistance

The resistances calculated above are summed $F_T = F_{Gk} + F_R + F_{TC} + F_{CT}$ and Subroutine VELFOR (See description of Subroutine IV5 of the Areal Module) is used to calculate the maximum speed achievable, V_p, overcoming F_T.

5. Speed Limited by Surface Roughness

Each road or trail unit description includes an RMS elevation indicating microroughness of the surface. The VRIDE array, calculated from the results of repeated runs of the Ride Dynamics Module (Vol. III) is interpolated to yield the maximum speed, V_R, achievable over the current road or trail unit keeping the driver absorbed power below the level used in the cross-plots of the results of the Ride Dynamics Module. These crossplots are part of the vehicle data file. (See Section II.D.10.)

6. Speed Limited by Sliding on Curves

For trails and secondary roads it is possible for banking (or reversed camber) to be steep enough so that vehicles could slide on curves at achievable speeds. This section calculates the maximum speed achievable before sliding by

$$V_s = [385.9R_c(\tan e_a + TFOR/GCW)/ (1 - TFOR\tan e_a/GCWP)]^{1/2}$$

for trails,

$$V_s = [385.9R_c(\tan e_a + \mu_j)/(1 - \mu_j\tan e_a)]^{1/2}$$

for roads,

7. Speed Limited by Tipping on Curves

Similarly the maximum speed with which a vehicle could negotiate a curve before tipping over is calculted by

$$V_T = [385.9R_c(t_w + h_{CG}\tan e_a)/(h_{CG} - t_w\tan e_a)]^{1/2}$$

where t_w = maximum tread width

h_{CG} = height of CG.

8. Speed Limited by Visibility

The speed limited by a driver being able to see an obstruction in time to brake to a halt before striking it is calculated by a call to Subroutine IV13 of the Areal Module. That subroutine requires the maximum braking force available from the vehicle and surface, possibly attenuated by the maximum deceleration to which the driver will choose to be subjected. The total braking force, F_{TB}, available is calculated by Subroutine IV11 of the Areal Module in case the terrain unit is a trail (soft surface). For roads (hard surfaces)

$$F_{TBk} = GCW\sin\theta_k + \min\,[X_{BR},\ \mu_j GCWB\cos\theta_k]$$

where X_{BR} = maximum braking effort of the vehicle.

The program may call a NOGO (not completing the processing for this terrain unit) if $F_{TBk} < 0$ due to insuffient braking ability.* When the total braking effort is available, the maximum braking effort the driver will actually use is calculated by

When the total braking effort is available, the maximum braking effort the driver will actually use is calculated by

$$F_{BMAX,k} = \min\,[D_{max}\ GCW,\ F_{TBk}\ S_p/100.]$$

where D_{max} = maximum deceleration which the driver will use

S_p = percent of total braking effort driver will use (simulating the effect that the driver will reserve some percentage of the braking effort "for safety")

These driver limits may be overriden by setting D_c and S_p to higher values. in the Scenario inputs. $F_{BMAX,k}$ is the force used by Subroutine IV13 to calculate speed limited by visibility, V_v.

* If post processors will concern themselves with acceleration/deceleration between terrain units in traverse mode, users may wish to change the code to allow all calculations in spite of the lack of braking ability (this requires the elimination of the two RETURN statements in Section 5B - Total Braking Hard Surface of the Computer Program).

9. Curvature Speed Limit

Since physical stability limits on vehicles are often far greater than the self imposed ones exhibited by a driver, a set of empirically derived curvature vs. speed limits are included in the Road Module Of the INRMM. They are based on a curve published by AASHO (1965) supplemented by observations made at the USAEWES. The relationships used are presented in Table I.E.1. This table is currently an integral part of the terrain input routine MPRD74. If user terrain input routines are to be added which will read road or trail data, this table (or others like it) will have to be included in these new routines.

Table II.E.1

Curvature Speed Limits

Radius of Curvature (feet)	Speed Limits (MPH) Super-highways	Primary Roads	Secondary Roads	Trails
5730	100	100	70	55
1910	70	70	60	49
1146	60	60	58	44
819	54	54	50	42
637	48	48	43	39
458	41	41	36	34
327	34	34	31	29
229	29	29	26	23
164	25	25	23	19
115	19	19	19	14
82	13	13	13	10

The curvature speed limit, V_c, is calculated by a linear interpolation of this table. If the coefficient of friction, μ_j, or the surface limited pull ratio, TFOR/GCWP, is less than .7, the curvature speed limit, V_c, is further attenuated by the square root of the ratio of μ_j or TFOR/GCWP to .7.

10. Speed Selection

The Maximum roadway speed is now chosen for each slope angle θ_K to be the minimum of the following:

$VTIRE_j$ = speed limited by tire at inflation pressure j

V_p = speed limited by resistance

V_L = posted speed limits (scenario variable)

V_T = speed limited by tipping

V_S = speed limited by sliding on curve

V_{RID} = speed limited by ride roughness

V_v = speed limited by visibility/braking

V_c = speed limited by curvature.

In traverse mode, NTRAV = 1, only one speed for θ_k is reported. For bidirectional travel, NTRAV = 3, a speed for θ_k and $-\theta_k$ is calculated and a harmonic average is taken as the selected overall speed for the road or trail unit.

F. Linear Feature Module

No code or description of the Linear Feature Module of the INRMM is included at this stage of development.

III INPUTS AND OUTPUTS

A. Introduction

When the time came to produce a FORTRAN implementation of the Operational Modules of the INRMM, NRMM, following the AMC-74 Report (which is essentially a coding specification) the decision was made that the Vehicle and Terrain Preprocessors, Areal, Road and Linear Feature Modules would be coded following the ANSI FORTRAN -66 standard, to allow portability. However input and output would be part of the Control and I/O Module which need not conform to the Standard. The programmers have attempted, however, to use only those extensions to the FORTRAN language which are commonly available. In particular in NRMM, NAMELIST directed READ and WRITE are used for vehicle, scenario and control input and most output.

To use the model, the sequence of operations may be organized as follows:

1. Vehicle data are collected (or measured, estimated or assigned by analogy) and organized into computer input files in formats specified by Sections III.B, Vol. III, Vehicle Data for the Ride Dynamics Module, and III.a, Vol. II, Vehicle Data for the Obstacle Module, described below.

2. Terrain data are gathered (or measured, estimated, or assigned by analogy) and organized into computer input files in formats specified by Section III.C, Terrain Data for the Operational Modules, described below.

3. The range of obstacle and surface roughness data present in the terrain data files is used to specify the base values for input to the preprocessor modules. These base values are organized into input files in formats specified in Volumes II and III.

4. The Ride Dynamics Module, VEHDYN, is run using the vehicle data file and a terrain profile exhibiting a value of the surface roughness (RMS), obstacle height (OBH) of a single obstacle or obstacle spacing (OBS) for sequence of equally spaced, identical obstacles. Successive runs are made varying forward speed (V). Crossplots are made for RMS vs. V for fixed value of absorbed power, OBH vs. V for fixed value of maximum vertical acceleration, and OBS vs. V for fixed values of obstacle height and maximum vehicle acceleration. These three tables are made part of the vehicle input data for the Operational Modules.

5. The Obstacle Module, OBS78B, is run using the vehicle data file organized as in Section III.A, Vol. II and a table of obstacles described by height (OBH), approach angle (OBAA), and width (OBW). For each obstacle, OBS78B will calculate minimum clearance (CLRMIN), maximum force (FOOMAX), and average force (FOO) during a simulated override of the obstacle. These sets of six numbers are organized into a single table which is made part of the vehicle input data for the Operational Modules.

6. The vehicle data, along with the results of Steps 4 and 5 or equivalent data, above, is organized into a computer input file in format as specified by Section III.B below.

7. The scenario/control file is constructed for each operational module run as specified by Section III.D below.

8. The Operational Modules, organized into a source file called NRMM, is run using the vehicle file from Step 6, the terrain file from Step 2, and the scenario/control file from Step 7, above.

The result of these eight steps will be a computer file giving vehicle speed for each terrain unit which was included in the terrain file of the Operational Modules. Other results beside speed are also given. Usually, further computer programs which use this terrain unit vs. speed file as input will be required to further evaluate the vehicle performance. These other programs, called post-processor as a group, will be developed by the user for his or her own purpose.

Experienced programmers will notice that the coded version of the INRMM is highly modular and coded so that a person reading the code will have a good idea of what is being calculated. This approach has resulted in a rather inefficient program, both globally and locally. Experienced programmers will undoubtedly want to enhance program efficiency after the current code is understood.

B. Vehicle Data

The data used to describe the vehicle for NRMM consist of a large number of vehicle descriptors together with outputs of the ride dynamics and obstacle modules. All these data are organized into a single computer file and read from unit LUN3. The vehicle descriptors are read using a NAMELIST directed read statement (NAMELIST VEHICL). The descriptors which are included are listed and described in Table III.B.1.

The data from the Obstacle Module is read using a formatted READ statement which accepts the results as produced by the Obstacle Module program and described in Volume II. Sample input files are contained in Appendix B.

Table III.B.1

Vehicle Input Data - NAMELIST VEHICL

Variable Name	Description
ACD	Aerodynamic drag coefficient
ASHOE(I)	Area of one track shoe on track assembly I, (in^2.)
AVGC	Average cornering stiffness of tires (lb./deg.)
AXLSP(I)	Distance from running gear assembly I to next assembly (inter-axle distance) (in.)
CD	Hydrodynamic drag coefficient
CGH	Height of CG of loaded vehicle above ground (in.)
CGLAT	Lateral distance of CG measured from centerline of combination (in.)
CGR	Loaded horizontal distance from CG to centerline of rearmost suspension assembly of prime mover (in.)
CID	Displacement of each engine (in^3.)
CL	Minimum ground clearance of combination (in.)
CLRMIN(I)	Minimum ground clearance of assembly I (in.)
CONV1(1,J)	Input speed component of the torque converter speed ratio versus torque converter input speed curve, (rpm)
CONV1(2,J)	Speed ratio component of the torque converter speed ratio versus torque converter input speed curve at constant input torque, TQIND
CONV2(1,J)	Torque ratio component of the torque converter speed ratio versus torque converter torque ratio curve
CONV2(2,J)	Speed ratio component of the torque converter speed ratio versus torque converter torque ratio curve
DFLCT(I,J)	Deflection of each tire on axle assembly I under load WGHT(I)/NWHL(I), in., at the pressure specified for J=1 fine grained, =2 coarse grained, =3 highway
DIAW(I)	Outside wheel diameter of unloaded tires on running gear assembly I (in.)
DRAFT	Combination draft when fully floating (in.) (0 if combination cannot float)

TABLE III.B.1 (Continued)

Variable Name	Description
ENGINE(1,J)	Engine speed component of engine speed versus engine torque curve, (rpm)
ENGINE(2,J)	Engine torque component of engine speed versus engine torque curve, (lb.-ft.)
EYEHGT	Height of driver's eyes above ground (in.)
FD(1)	Final drive gear ratio
FD(2)	Final drive efficiency
FORDD	Maximum water depth combination can ford (in.) (Note: FORDD= DRAFT if DRAFT ≠0.)
GROUSH(I)	Track grouser height of track assembly I (in.)
HPNET	Net engine power (HP.)
HVALS(N)	Nth obstacle height in driver limited speed vs obstacle height table for single obstacle crossing
IAPG	0 if power train data available only, 1 if both measured tractive effort and power train data given, 2 if measured tractive effort given only
IB(I)	1 if running gear assembly I is braked, 0 otherwise
ICONST(I)	0 if radial tires are on wheel assembly I, 1 if bias tires
ICONV1	Number of point pairs in the array CONV1(I,J)
ICONV2	Number of point pairs in the array CONV2(I,J)
ID(I)	0 if wheels on wheeled assembly I are singles, 1 if duals
IDIESL	2 if the engine is a two cycle diesel, 1 otherwise
IENGIN	Number of point pairs in the array ENGINE(I,J),
IP(I)	1 if running gear assembly I is powered, 0 otherwise
IPOWER	Number of point pairs in the array POWER(I,J)

TABLE III.B.1 (Continued)

Variable Name	Description
IT(I)	0 if assembly I is not part of a tandem axle, J if assembly I is the Jth of a tandem axle
ITCASE	1 if vehicle has engine to transmission transfer gear box 0 otherwise
ITRAN	0 if transmission is manual with clutch, 1 if automatic transmission with torque converter
ITVAR	1 if transmission is mechanical, 0 if transmission is hydraulic
LOCOIF	1 if all powered running gear assemblies have locking differentials, 0 otherwise
LOCKUP	0 if torque converter does not lockup, 1 if torque converter has lockup
MAXIPR	Number of surface roughness values per tolerance level
MAXL	Number of roughness tolerance levels specified
NAMBLY	Total number of running gear assemblies
NBOGIE(I)	Number of road wheels on track assembly I
NCHAIN(I)	1 if chains are present on tire on assembly I 0 otherwise
NCYL	Number of cylinders per engine
NENG	Number of engines
NFL(I)	0 if track on assembly I is rigid, 1 otherwise
NGR	Number of transmission gear ratios
NHVALS	Number of height values used in arrays VOOB and HVALS

TABLE III.B.1 (Continued)

Variable Name	Description
NPAD(I)	1 if track on assembly I has pads, 0 otherwise
NSVALS	Number of obstacle spacing values used in arrays VOOBS and SVALS
NVEH(I)	0 if running gear assembly I is tracked, 1 if wheeled
NWHL(I)	Number of tires on wheeled assembly I
NWR	Number of water depths between 0 and FORDD for which weight ratios are given
PBF	Maximum force pushbar can tolerate (lb.)
PBHT	Unit pushbar height (in.)
PFA	Vehicle projected frontal area (in^2.)
POWER(1,J)	Vehicle velocity component of the tractive effort versus speed curve (mi/hr. on input)
POWER(2,J)	Tractive force component of the tractive force versus speed curve (lb.)
QMAX	Maximum torque of each engine (ft.-lb.)
RDIAM(I)	Rim diameter of wheel for tires on axle assembly I (in.)
REVM(I)	Revolutions/mile of tire element on assembly I, (rev/mi)
RIMW(I)	Wheel rim width of assembly I, (in.)
RMS(N)	Nth surface roughness value (in.)
RW(I)	Track thickness + bogie rolling radius for tracked assembly I (in.)
SAE	Swamp angle (egress) (deg.)
SAI	Swamp angle (ingress) (deg.)

TABLE III.B.1 (Continued)

Variable Name	Description
SECTH(I)	Section height of tires on running gear assembly I (in.)
SECTW(I)	Section width of tires on running gear assembly I
(in.)	
SVALS(N)	Nth obstacle spacing in driver limited speed versus obstacle spacing table for successive obstacle crossing
TCASE(1)	Gear ratio for gear between engine and transmission (1. if no such gear)
TCASE(2)	Efficiency of gear between engine and transmission (1. if no such gear)
TL	Distance from front of first running gear assembly to rear of last (in.)
TPLY(I)	Tire ply rating of tires on axle I
TPSI(I,J)	Tire inflation pressure of tires on axle I (psi), specified for j=1 fine grained, =2 coarse grained, =3 highway
TQIND	Constant torque converter input torque at which torque converter performance curves are measured, (lb.-ft.)
TRAKLN(I)	Track length of track assembly I (in.)
TRAKWD(I)	Track width of track assembly I (in.)
TRANS(1,J)	Transmission gear ratio of gear NG
TRANS(2,J)	Transmission efficiency of gear NG
VAA	Vehicle approach angle (deg.)
VDA	Vehicle departure angle (deg.)
VFS	Vehicle fording speed
VOOB(I)	Maximum driver limited speed at which vehicle can impact an obstacle of height HVALS(I) if obstacles are spaced farther than two vehicle lengths apart (mph)

TABLE III.B.1 (Continued)

Variable Name	Description
VOOBS(I)	Maximum driver limited speed at which vehicle can impact successive obstacles spaced SVALS(I) apart (mph)
VRIDE(I,J)	Maximum speed over ground for surface roughness class I at roughness tolerance level J (mph)
VSS	Maximum combination still water speed without auxiliary propulsion (mph)
VSSAXP	Maximum combination still water speed with auxiliary propulsion (mph)
WC	Winch capacity (lb.)
WDAXP	Water depth at which auxiliary power can be used (in.)
WDPTH(N)	Nth water depth (in.)
WDTH	Maximum combination width (in.)
WGHT(I)	Weight on running gear assembly I (lb.)
WI	Minimum width between running gear elements (in.)
WRAT(N)	Ratio of vehicle weight on ground to total vehicle vehicle weight at Nth water depth, WDPTH(N)
WRFORD	Proportion of combination weight supported by ground when combination is operating at maximum fording depth
WT(I)	Tread width of running gear assembly I (in.) (Center to center plane if duals)
WTE(I)	Minimum width between left-right suspension elements (tires or tracks) on assembly I (in.)
XBRCOF	Maximum combination braking coefficient per assembly in lb./lb. of load carried

C. Terrain Input - Operational Modules

The user has several options in the organization of his terrain input data. First, the data required to describe each terrain unit varies depending on its nature, areal, road or linear feature. Secondly, over the years areal terrain data has been amassed in computer readable form in two different structures and the current computer program contains terrain data input/translation routines which accept data in each of these arrangements and perform the computations necessary to pass the expected data to the Terrain Preprocessor and the Areal Module. Only a single organization of road data is presently supported. Finally the user can, of course, prepare a customized subroutine to accept data in another format. The terrain data organization is signaled by the control variable MAP. The three organizations of data now implemented are described below.

It should be noted that the terrain input routines and the output subroutine, BUFFO, are now coded to deal with terrain data files which contain only areal units or road units, not a mixture of the two types.

1. Areal Terrain Input File - Class Interval Values

The first option for representation of areal terrain data is that used in the AMC-71 Mobility Model. For each of the 22 primary descriptors of the terrain unit, the data file contains the [integer] value of the factor class in which the descriptor lies.

Table III.C.1

Terrain Input File Structure

Variable Name	Input Format	Description
NPAT	I4	Terrain unit number (index)
IST	I2	Soil type - 1 fine grained 2 coarse grained 3 muskeg 6 fine grained soil CH relatively impervious to water
IRCI1	I2	Soil strength class - dry season
IRCI2	I2	Soil strength class - normal season
IRCI3	I2	Soil strength class - wet season
IGRADE	I1	Topographic slope class
IOA	I2	Obstacle approach angle class
IOH	I1	Obstacle height class
IOW	I1	Obstacle width class
IOL	I1	Obstacle length class
IOS	I1	Obstacle spacing class
IOST	I1	Ostacle spacing type 1 - potentially avoidable (random) 2 - nonavoidable (linear)
IRMS	I1	Surface roughness class
ISTEM1	I1	Spacing class of vegetation in stem diameter class 1 and greater
ISTEM2	I1	Spacing class of vegetation in stem diameter class 2 and greater
ISTEM3	I1	Spacing class of vegetation in stem diameter class 3 and greater

Table III.C.1 (Continued)

Variable Name	Input Format	Description
ISTEM4	I1	Spacing class of vegetation in stem diameter class 4 and greater
ISTEM5	I1	Spacing class of vegetation in stem diameter class 5 and greater
ISTEM6	I1	Spacing class of vegetation in stem diameter class 6 and greater
ISTEM7	I1	Spacing class of vegetation in stem diameter class 7 and greater
ISTEM8	I1	Spacing class of vegetation in stem diameter class 8 and greater
IRD	I1	Recognition distance class
AREA	F10.3	Area

Note: Area is not used in the Mobility Model. It is included in many of the existing terrain files and is included in the input and reported in the output for convenience in further processing of the output.

2. Areal Terrain Input File - Real Values

The second option supported by the current program for input of terrain data is a format used for a variety of studies performed in the United States during the middle of the 1970 decade. This format provides for input of actual values for various primary terrain descriptors. The organization of the data for each terrain unit is presented in Table III.C.2.

A few changes in the content of the terrain data should be noted. First, a fourth soil strength value is included to extend to an extremely wet seasonal condition, presently taken as the wet season condition during a year when daily rainfall is 150% of the long term average values for the area. Secondly, four recognition distances are included to provide for the variation in this quantity with season (primarily due to changes in vegetation). The value used is determined from the scenario variable MONTH.

The organization of the data should be carefully noted. The data for each terrain unit occupy two lines of the input file and some of the positions on each line are not used. Most of the data are read as four digit whole numbers with the decimal point omitted. To allow the expected range of the values to fit reasonably with the implied scaling a mixture of units is used, as noted in the table. In particular the surface roughness value is multiplied by 10.

Table III.C.2

Terrain Input File Structure - Real Values

Variable Name	Input Format	Description
NTU	I5	Terrain unit number
IST	I2	Soil type - 1 fine grained 2 course grained 3 muskeg 6 fine grained soil CH relatively impervious to water
	20X	Filler
RCIC(1)	I4	Soil strength - dry (RCI)
RCIC(2)	I4	Soil strength - average (RCI)
RCIC(3)	I4	Soil strength - wet (RCI)
RCIC(4)	I4	Soil strength - wet,wet (RCI)
GRADE	I4	Topographic slope (%)
AA	I4	Obstacle approach angle (deg.)
OBH	I4	Obstacle height (in.)
OBW	I4	Obstacle width (ft.)
OBL	I4	Obstacle length (ft.)
OBS	I4	Obstacle spacing (ft.)
IOST	I4	Obstacle spacing type 1 - potentially avoidable (random) 2 - nonavoidable (linear)
ACTRMS	I4	10*Surface roughness (RMS - 10*in.)
NOTE: The remaining data form the second line		
	5X	Filler
S(1)	I4	Mean spacing (in ft.) of vegetation in stem diameter class 1 and greater
S(2)	I4	Mean spacing (in ft.) of vegetation in stem diameter class 2 and greater

Table III.C.2 (Continued)

Variable Name	Input Format	Description
S(3)	I4	Mean spacing (in ft.) of vegetation in stem diameter class 3 and greater
S(4)	I4	Mean spacing (in ft.) of vegetation in stem diameter class 4 and greater
S(5)	I4	Mean spacing (in ft.) of vegetation in stem diameter class 5 and greater
S(6)	I4	Mean spacing (in ft.) of vegetation in stem diameter class 6 and greater
S(7)	I4	Mean spacing (in ft.) of vegetation in stem diameter class 7 and greater
S(8)	I4	Mean spacing (in ft.) of vegetation in stem diameter class 8 and greater
RDA1	I4	Recognition distance - Winter
RDA2	I4	Recognition distance - Spring
RDA3	I4	Recognition distance - Summer
RDA4	I4	Recognition distance - Autumn
AREA	F8.4	Area of the terrain unit

NOTE: Area is not used in the mobility model. It is included in many of the existing terrain files and is included in the input and reported in the output for convenience in further processing of the output.

3. Road Terrain Input File

The terrain data input routine contained in the program is that requried to accept the terrain data files which were prepared for studies conducted in the mid 1970's using programs other than the NRMM Road Module. These simple programs do not use the full collection of road descriptors required for the NRMM Road Module.

Missing values are provided by the input/translation routine, MPRD74. Furthermore, for the earlier studies, the road segment curvature was preprocessed into a speed limit so that the curvature must be computed from this speed limit (and the speed limit recomputed in TPP!). The organization of data is much like that described for areal terrain in the previous section. Soil strength is provided (for trails) for our seasonal conditions, visibility recognition distances for four seasons and most of the data is read as four digit whole numbers like that described in the previous section.

Table III.C.3

Terrain Input File Structure - Roads

Variable Name	Input Format	Description
NTU	I5	Terrain unit number
IROAD	I2	Road type index 1 - super highway 2 - primary road 3 - secondary road 4 - trail
IST	I2	Soil type (used for trails) 1 - fine grained 2 - coarse grained 6 - fine grained, CH, relatively impervious to water
IURB	I2	Urban code
RCIC(1)	I4	Soil strength - dry season (RCI)
RCIC(2)	I4	Soil strength - average season (RCI)
RCIC(3)	I4	Soil strength - wet season (RCI)
RCIC(4)	I4	Soil strength - wet, wet season (RCI)
GRADE	I4	Topographic slope (%)
RDA1	I4	Recognition distance - Winter (ft.)
RDA2	I4	Recognition distance - Spring (ft.)
RDA3	I4	Recognition distance - Summer (ft.)
RDA4	I4	Recognition distance - Autumn (ft.)
ACTRMS	I4	10*Surface roughness (RMS - in.)
CURVV	I4	AASHO Curvature speed limit (mph)
DIST	F8.4	Road segment length (mi.)

NOTE: Road segment length is not used in the mobility model. It is included in many of the existing terrain files and is included in the input and reported in the output for convenience in further processing of the output.

D. Scenario and Control Input Data

To provide flexibility to the user in the use of the NRMM many options are provided through the scenario and control variables. The control variables SEARCH and NTUX allow restriction of the operational modules to a single terrain unit anywhere in the terrain file. The variable MAP indicates the format of the terrain input file. The remainder of the control variables allow the user to generate output at any of the 41 breakpoints in the flow of data through NRMM.

The scenario variables provide an option to model some of the operations which may be possible for a given vehicle on a fixed terrain. Presently available terrain files contain soil strength and visibility limits under different conditions which are specified by ISEASN and MONTH. Shallow snow can be introduced using the indicator ISNOW with the snow described by COHES, GAMMA, PHI and ZSNOW. For roadways, an overall speed limit can be set by VLIM and recognition distance variation due to weather (e.g. fog) can introduced by RDFOG. Variables NSLIP and ISURF indicate surface moisture content for areal and road terrain units respectively.

The variable NTRAV allows the model to be run in "traverse" or "omnidirectional" modes. In the former mode the speed is calculated only for the topographic slope value (which may be negative) in the terrain data. In the latter mode, speeds are calculated with the input slope, its negative and [for areal units] zero slope and then the harmonic average of the speeds is computed and reported.

Several of the scenario variables describe the man-vehicle-terrain relations. These include the roughness acceptance level, LAC, speeds at which a driver will proceed regardless of lack of visibility (which may be different on and off road), VBRAKE and VISMN, and the driver's reaction time, REACT. Braking is controlled for comfort and safety through DCLMAX and SFTYPC while vegetation override is set through IOVER and VWALK.

The final scenario variables allow selection of the tire pressure (including the possibility of using the optimal tire pressure for each terrain unit as is possible on a vehicle equipped with a proper central tire inflation system) and selecting between an input (perhaps measured) tractive effort vs speed curve and one produced by the program from the power train characteristics.

The control and scenario variables are assembled into a single input data file and input to the program through two NAMELIST directed READ statements located in subroutine SCN. In SCN, default values are first set. Next most of the control variables are read using NAMELIST "CONTRL". The output print controls are then set as determined by the value of DETAIL. Finally, the remainder of the control variables and the scenario variables are read using NAMELIST "SCENAR". It is assumed that the computer self initializes variables to zero. The control and scenario variables are described in Tables III.D.1 and III.D.2. The variables are grouped in the tables as they are to be grouped in the input data file. Thus the second table contains both control and scenario variables. A sample data file is contained in the Appendix.

TABLE III.D.1

Control Input Data - Operational Modules
NAMELIST CONTRL

Variable Name	Default Value	Description
DETAIL	1	Level of detail of output 1-Summary output together with any detailed outputs specified through the KXXXX variables described below. 2-Echoes of vehicle, control and scenario inputs together with output for each terrain unit of a) terrain unit number and type b) selected speed made good c) average speed made good d) speeds upgrade, on level and down grade contributing to the selected speed made good e) speeds upgrade, on level and down grade contributing to the average speed made good f) grade g) terrain unit area 3,4-Presently, the same as 2 5-All output possible from the vehicle preprocessor together with a printer plot of the tractive effort vs speed curve - terminates execution after the vehicle preprocessor 10-All outputs - in general, inputs and outputs are reported at each breakpoint in the flow of computation through the vehicle pre-processor and areal modules - can only be run on a single terrain unit (the effect is the same as settinng all the KXXXX variables to 1)
NTUX		The number of the terrain unit when output is to be produced for a single terrain unit from a terrain input data file with more than one terrain unit (see SEARCH)
SEARCH		Search option flag: 0-Output desired for all terrain units in terrain input data file 1-Output desired only for a single terrain unit specified by NTUX

TABLE III.D.1 (Continued)

Variable Name	Default Value	Description

Note: The remaining variables are the output controls. For each the output described is produced if the variable = 1 and is not produced if the variable = 0. If DETAIL = 5 or 10, all values read are over written and set to 1.

Variable Name	Default Value	Description
KSCEN		Control and scenario variables are echoed
KVEH		Vehicle input data is echoed
KII1		Input and output of subroutine II1
KII2		Input and output of subroutine II2
KII3		Input and output of subroutine II3
KII4		Input and output of subroutine II4
KII5		Input and output of subroutine II5
KII6		Input and output of subroutine II6
KII7		Input and output of subroutine II7
KII8		Input and output of subroutine II8
KII9		Input and output of subroutine II9
KII10		Input and output of subroutine II10
KII11		Input and output of subroutine II11
KII12		Input and output of subroutine II12
KII13		Input and output of subroutine II13
KII14		Input and output of subroutine II14
KII15		Input and output of subroutine II15
KII16		Input and output of subroutine II16
KII17		Input and output of subroutine II17
KMAP		Terrain input data are echoed. The values reported are those passed to the computational modules after any translation required for the input file format
KTPP		Input and output of the terrain preprocessor
KIV1		Input and output of subroutine IV1
KIV2		Input and output of subroutine IV2
KIV3		Input and output of subroutine IV3
KIV4		Input and output of subroutine IV4
KIV5		Input and output of subroutine IV5
KIV6		Input and output of subroutine IV6
KIV7		Input and output of subroutine IV7
KIV8		Input and output of subroutine IV8
KIV9		Input and output of subroutine IV9
KIV10		Input and output of subroutine IV10
KIV11		Input and output of subroutine IV11
KIV12		Input and output of subroutine IV12

TABLE III.D.1 (Continued)

Variable Name	Default Value	Description
KIV13		Input and output of subroutine IV13
KIV14		Input and output of subroutine IV14
KIV15		Input and output of subroutine IV15
KIV16		Input and output of subroutine IV16
KIV17		Input and output of subroutine IV17
KIV18		Input and output of subroutine IV18
KIV19		Input and output of subroutine IV19
KIV20		Input and output of subroutine IV20
KIV21		Input and output of subroutine IV21

Note: This table is organized in a (hopefully) logical rather than alphabetical order - In particular, the KXXXX which provide output control are grouped after the other variables and are in order of use in the program.

TABLE III.D.2

Control and Scenario Variables - Operational Modules
NAMELIST SCENAR

Variable Name	Default Value	Description
COHES	.05	Cohesion of snow (lb./in^2.)
DCLMAX	.50	Maximum deceleration the driver will actually accept (g's)
GAMMA	.20	Specific gravity of snow
IOVER	9	The index of the maximum stem diameter class to be overriden if the speed to do so is greater than walking speed
ISEASN	1	Seasonal soil strength (moisture) indicator for areal terrain - 1 dry 2 normal 3 wet 4 wet,wet
ISURF	1	Seasonal surface traction condition indicator for roads - 1 dry 2 wet 3 ice covered
ISNOW		Shallow snow cover indicator 0 no snow 1 snow cover
LAC	1	Indicator of surface roughness absorbed power acceptance level
MAP	71	Terrain input file format indicator 71 - class interval values, areal terrain 74 - Real values, areal terrain 11 - Real values, road terrain
MAPG	1	Powertrain computation method 1-Tractive effort vs speed data calculated from engine & transmission characteristics 2-Input tractive effort vs speed data used
MONTH		Month of year (1=Jan.,2=Feb.,etc. required if MAP= 11 or 74)
NOPP	0	Operating tire pressure indicator 0-Tire pressure used for each terrain unit determined by surface type in that unit 1-Tire pressure for cross-country used for all terrain units 2-Tire pressure for sand used for all terrain units 3-Tire pressure for highways used for all terrain units
NSLIP	0	Surface moisture due to rain fall indicator (for areal terrain) 0-No moisture to make surface slippery 1-Less than 1 in. of rain with no free surface water 2-Less than 6 hours flooding with no free surface water

TABLE III.D.2 (Continued)

Variable Name	Default Value	Description
NSLIP		3-More than 6 hours flooding with no free surface water 4-Less than 1 in. of rain with free surface water 5-Less than 6 hours flooding with free surface water 6-More than 6 hours flooding with free surface water
NTRAV	3	Operational mode 1-Traverse 3-Omnidirectional
NTUX		Number of terrain unit to be examined if SEARCH = 1
PHI	21.0	Internal friction angle of snow (deg.)
RDFOG	1000.0	Recognition distance on road, influenced by weather (in.)
REACT	.5	Driver reaction time (time from recognition to initiation of deceleration) (sec.)
SFTYPC	90.0	Percent of maximum deceleration available that the driver will actually use (%)
VBRAKE	5.0	Speed at which vehicle will proceed on a road if visibility is entirely obscured (mph)
VISMNV	2.0	Speed at which vehicle will proceed in areal terrain if visibility is entirely obscured (mph)
VLIM	55.0	Speed limit on road (mph)
ZSNOW	3.0	Snow depth (in.)

E. Output

NRMM has been designed to provide to its users a quantity of output data which can vary from a single number, speed-made-good on one terrain unit, to hundreds of pages covered with numbers (to a printer and/or file). The selection of the data to be written during the program execution from that which is possible is determined by the input values of the control variables described in the preceding section. Four choices of outputs are implemented in the NRMM computer code at present through the control variable DETAIL.

If DETAIL = 1 only the basic output of the NRMM is written for each terrain unit in the terrain input file. This consists of the following for areal terrain units:

1. Terrain unit number, NTU
2. Terrain type, ITUT
   - 1 - normally dry patch
   - 2 - marsh or other water covered patch
   - 11 - superhighway
   - 12 - primary road
   - 13 - secondary road
   - 14 - trail
3. The omnidirectional speed-made-good attainable by the vehicle in the terrain unit, VMAX
4. The attainable speed-made-good going with the topographic slope (up grade)
5. The attainable speed-made-good on level
6. The attainable speed-made-good going against the topographic slope (down grade)
7. The selected omni-directional speed-made-good which considers both the vehicle capabilities and human factors, VSEL
8. The selected speed-made-good upgrade

9. The selected speed-made-good on level
10. The selected speed-made-good down grade
11. Grade (topographic slope)
12. Area of the terrain unit

All speeds are in miles per hour, grade (topographic slope) is in percent and area is repeated from the input file. On roads, since the selected speed-made-good is the same as the attainable speed-made-good, items 3.-6. are not included in the output, also, on a road one can only proceed with or against the given grade so item 9 is omitted.

If DETAIL = 2, the basic output table described above is preceded by listings of the values of the control and scenario variables and the vehicle input data. This additional data is written using NAMELIST, so that the name of each variable precedes its value.

Setting DETAIL = 5 provides complete information about the vehicle pre-processor. After each major subroutine of the vehicle pre-processor (II1 - II17) a NAMELIST is written which contains the input and output of the subroutine. In addition, a line printer-plot of the tractive effort vs. speed curve is output. This printer-plot is also produced when errors are detected in the automatic curve-fitting section of the vehicle pre-processor. The program execution is terminated after the vehicle pre-processor (before any terrain data is read) at this output level.

If DETAIL = 10, all available data is written to the output file. In addition to all the data described above (except the printer plot), after each major subroutine of the Areal Module (IV1-IV21) the input and output of the subroutine is written. Again, this is done through a NAMELIST so that each variable is identified. Detail level 10 output is not presently implemented for the Road Module. Since this output is voluminous, the program terminates after a single terrain unit.

The user of the NRMM who requires a different selection of data from that written under one of the above options has two choices. The 42 control variables for writing of intermediate output KII1-KII17, KIV1-KIV21, KMAP, KSCEN, KTPP and KVEH provide great flexibility for production of desired output. Alternately, the output section of the Control & I/O Module, Subroutine BUFFO, can be modified to yield the desired results. Samples of output are included in the appendices.

IV REFERENCES

Abramowitz, Milton and Stegan, Irene A., Handbook of Mathematical Functions, National Bureau of Standards, Applied Mathematics Series, December 1965.

American Association of State Highway Officials (AASHO), A Policy on Geometric Design of Rural Highways, 1965, The American Association of State Highway Officials, Washington, D. C., pp 152 to 159.

Eklund, K. F., "The Influence of Load and Inflation on the Selection of Pneumatic Tires for Military Vehicles", Report 922, April 1945, U. S. Army Engineers Research and Development Laboratories, Fort Belvoir, Va.

Jurkat, M. P., Nuttall, C. J., and Haley, P. W., "The AMC '74 Mobility Model", Technical Report No. 11921 (LL-149), May 1975, U. S. Army Tank-Automotive Command, Warren, Mich.

Lessem, A. S., "Dynamics of Wheeled Vehicles: A Mathematical Model for the Traversal of Rigid Obstacles by a Pneumatic Tire", Technical Report M-68-1, U. S. Army Waterways Experiment Station, CE, Vicksburg, Miss., May 1968.

Merritt, H. E., "The Evolution of a Tank Transmission", The Institution of Mechanical Engineers, Vol. 154, p.412, April 1946.

Nuttall, C. J., Jr., and Dugoff, H. J., "A Hierarchial Structure of Models for the Analysis of Land Mobility Systems", presented at Twelfth Annual U. S. Army Operations Research Symposium, Durham, N. C., Oct 1973.

Nuttall, C. J., Jr., Rula, A. A., and Dugoff, H. J., "Computer Model for Comprehensive Evaluation of Cross-Country Vehicle Mobility", Earthmoving Industry Conference, SAE Paper No. 740426, presented at Society of Automotive Engineers, Apr 1974.

Pradko, F., Lee, R. A., and Kaluza, V., "Theory of Human Vibration Response", Paper 66-WA-BHF-15, presented at ASME Meeting, Nov 1966.

Ray, J. R., "Investigation of the Factors Involved in Steering Tracklaying Vehicles", Technical Report No. 10969 (AD 871157), May 1970, U. S. Army Tank-Automotive Command, Warren, Mich.

Rula, A. A., and Nuttall, C. J., Jr., "An Analysis of Ground Mobility Models (ANAMOB)", Technical Report M-71-4, Jul 1971, U. S. Army Engineer Waterways Experiment Station, CE, Vicksburg, Miss.

Rula, A. A. and Nuttall, C. J., Jr., "Terrain Modeling to Support Mobility Evaluation", Proceedings of Fifth International Conference, The International Society for Terrain-Vehicle Systems, Inc., Detroit, Mich., Vol 4, Jun 1975.

Schreiner, B. G., "A Technique for Estimating the Performance of Tracked Vehicles in Muskeg", Journal of Terramechanics, Vol. 4, 1967, No. 3, pp 23 to 29.

Smith, G., "Commercial Vehicles Performance and Fuel Economy", Society of Automotive Engineers, Special Publication 355, 1970.

Taborek, J. J., Mechanics of Vehicles, Penton Publications, Cleveland, Ohio, 1957.

Turnage, G. W., "Performance of Soils Under Tire Loads, Application of Test Results to Tire Selection for Off-Road Vehicles", Technical Report 3-666, Report 8, Sept 1972, U. S. Army Waterways Experiment Station, CE, Vicksburg, Miss.

U. S. ATAC, "The AMC-71 Mobility Model", Technical Report No. 11789 (LL 143), Vols. I and II, Jul 1973, U. S. Army Tank-Automotive Command, Warren, Mich.

```
      PROGRAM NRMM (
     +    INPUT   = 81,
     +    OUTPUT  = 137,
     +    TAPE1   = OUTPUT,
     +    TAPE2   = 81,
     +    TAPE3   = 81,
     +    TAPE4   = 81,
     +    TAPE5   = 81,
     +    TAPE10  = 512 )
C
C     ------------------------------------------------------------------
C     TITLE:
C        NATO REFERENCE MOBILITY MODEL
C     ORGANIZATION:
C        U.S. ARMY TANK AUTOMOTIVE RESEARCH AND DEVELOPMENT COMMAND
C     ------------------------------------------------------------------
C
C     1. LABLED COMMON ASSIGNMENTS
         COMMON /IO/        IEOF
         COMMON /IO/        KBUFF
         COMMON /IO/        LUN1
         COMMON /IO/        LUN2
         COMMON /IO/        LUN3
         COMMON /IO/        LUN4
         COMMON /IO/        LUN5
         COMMON /IO/        LUN6
         COMMON /IO/        LUN7
         COMMON /IO/        LUN8
         COMMON /IO/        LUN9
         COMMON /IO/        LUN10
         INTEGER            MB
         COMMON /INDEX/     MB
         INTEGER            DOWN
         COMMON /INDEX/     DOWN
         INTEGER            EGF
         COMMON /INDEX/     EGF
         INTEGER            FORCE
         COMMON /INDEX/     FORCE
         INTEGER            GR
         COMMON /INDEX/     GR
         COMMON /INDEX/     LEVEL
         INTEGER            MN
         COMMON /INDEX/     MN
         INTEGER            RPM
         COMMON /INDEX/     RPM
         INTEGER            SPEED
         COMMON /INDEX/     SPEED
         INTEGER            SR
         COMMON /INDEX/     SR
         INTEGER            TR
         COMMON /INDEX/     TR
         INTEGER            TORQUE
```

```
COMMON /INDEX/     TORQUE
INTEGER            UG
COMMON /INDEX/     UR
INTEGER            MX
COMMON /INDEX/     MX
COMMON /VEHICL/    ACD
COMMON /VEHICL/    ASHOE     (20)
COMMON /VEHICL/    AVGC
COMMON /VEHICL/    AXLSP     (20)
COMMON /VEHICL/    CB
COMMON /VEHICL/    CGH
COMMON /VEHICL/    CGLAT
COMMON /VEHICL/    CGR
COMMON /VEHICL/    CID
COMMON /VEHICL/    CL
COMMON /VEHICL/    CLRMIN    (20)
COMMON /VEHICL/    CONV1     (2,25)
COMMON /VEHICL/    CONV2     (2,25)
COMMON /VEHICL/    CELCT     (20,3)
COMMON /VEHICL/    DIAW      (20)
COMMON /VEHICL/    DRAFT
COMMON /VEHICL/    ENGINE    (2,50)
COMMON /VEHICL/    EYEHGT
COMMON /VEHICL/    FB        (2)
COMMON /VEHICL/    FORDD
COMMON /VEHICL/    GROUSH    (20)
COMMON /VEHICL/    HWALS     (25)
COMMON /VEHICL/    IMPG
COMMON /VEHICL/    IB        (20)
COMMON /VEHICL/    ID        (20)
REAL               IDIESL
COMMON /VEHICL/    IDIESL
COMMON /VEHICL/    IENGIN
COMMON /VEHICL/    IR        (20)
COMMON /VEHICL/    ICONST    (20)
COMMON /VEHICL/    ICONV1
COMMON /VEHICL/    ICONV2
COMMON /VEHICL/    IPOWER
COMMON /VEHICL/    IT        (20)
COMMON /VEHICL/    ITCASE
COMMON /VEHICL/    ITRAN
COMMON /VEHICL/    ITVAR
COMMON /VEHICL/    LOCKUP
COMMON /VEHICL/    MAXIPR
COMMON /VEHICL/    MAXL
COMMON /VEHICL/    NAMBLY
COMMON /VEHICL/    NBOGIE    (20)
COMMON /VEHICL/    NCHAIN    (20)
REAL               NCYL
COMMON /VEHICL/    NCYL
REAL               NENG
COMMON /VEHICL/    NENG
```

```
      COMMON /VEHICL/   HPNET
      COMMON /VEHICL/   NFL        (20)
      COMMON /VEHICL/   NGR
      COMMON /VEHICL/   NHVALS
      COMMON /VEHICL/   NRAD       (20)
      COMMON /VEHICL/   NSVALS
      COMMON /VEHICL/   NVEH       (20)
      COMMON /VEHICL/   NWHL       (20)
      COMMON /VEHICL/   NWR
      COMMON /VEHICL/   PBF
      COMMON /VEHICL/   PBHT
      COMMON /VEHICL/   PEA
      COMMON /VEHICL/   POWER      (2,20)
      COMMON /VEHICL/   CMAX
      COMMON /VEHICL/   RDIAM      (20)
      COMMON /VEHICL/   REVM       (20)
      COMMON /VEHICL/   RIMW       (20)
      COMMON /VEHICL/   RNS        (20)
      COMMON /VEHICL/   RW         (20)
      COMMON /VEHICL/   SAE
      COMMON /VEHICL/   SAI
      COMMON /VEHICL/   SECTH      (20)
      COMMON /VEHICL/   SECTW      (20)
      COMMON /VEHICL/   SVALS      (25)
      COMMON /VEHICL/   TCASE      (2)
      COMMON /VEHICL/   TE
      COMMON /VEHICL/   TPLY       (20)
      COMMON /VEHICL/   TPSI       (20,3)
      COMMON /VEHICL/   TQIND
      COMMON /VEHICL/   TRAKLN     (20)
      COMMON /VEHICL/   TRAKWD     (20)
      COMMON /VEHICL/   TRANS      (2,20)
      COMMON /VEHICL/   VAA
      COMMON /VEHICL/   VDA
      COMMON /VEHICL/   VES
      COMMON /VEHICL/   VOOB       (25)
      COMMON /VEHICL/   VGOBS      (25)
      COMMON /VEHICL/   VRIDE      (20,3)
      COMMON /VEHICL/   VSS
      COMMON /VEHICL/   VSSAXP
      COMMON /VEHICL/   WC
      COMMON /VEHICL/   WDAXP
      COMMON /VEHICL/   WDPTH      (20)
      COMMON /VEHICL/   WDTH
      COMMON /VEHICL/   WGHT       (20)
      COMMON /VEHICL/   WRAT       (20)
      COMMON /VEHICL/   WRFORD
      COMMON /VEHICL/   WT         (20)
      COMMON /VEHICL/   WTE        (20)
      COMMON /VEHICL/   WWAXP
      COMMON /VEHICL/   XBRCOF
      COMMON /VEHICL/   WI
```

```
      COMMON /VEHICL/    LOCDIF
      COMMON /VEHICL/    SHF
      COMMON /PREP/      A          (3,4)
      COMMON /PREP/      ATF        (2Ø)
      COMMON /PREP/      EIF        (2Ø)
      COMMON /PREP/      CWARLN     (2Ø,3)
      COMMON /PREP/      COFCFG     (3)
      COMMON /PREP/      COFCCG     (3)
      COMMON /PREP/      COFCG      (2Ø,3)
      COMMON /PREP/      COFFG      (2Ø,3)
      COMMON /PREP/      CTF        (2Ø)
      COMMON /PREP/      CRAT       (2Ø,3)
      COMMON /PREP/      GCA        (2Ø,3)
      COMMON /PREP/      GCW
      COMMON /PREP/      GCWB
      COMMON /PREP/      GCWNB
      COMMON /PREP/      GCWNP
      COMMON /PREP/      GCWP
      COMMON /PREP/      FRT
      COMMON /PREP/      NDF        (38)
      COMMON /PREP/      NVEHC
      COMMON /PREP/      PWTE
      COMMON /PREP/      R          (3)
      COMMON /PREP/      RMK        (2Ø)
      COMMON /PREP/      RR
      COMMON /PREP/      TRACTF     (2Ø,5)
      COMMON /PREP/      TRAPSI     (3)
      COMMON /PREP/      VCICG      (2Ø,3)
      COMMON /PREP/      VCIFG      (2Ø,3)
      COMMON /PREP/      VCIMUK     (2Ø)
      COMMON /PREP/      VGV        (2Ø,5)
      COMMON /PREP/      VI         (2Ø,3)
      COMMON /PREP/      VTIRE      (3)
      COMMON /PREP/      XIMAX
      COMMON /PREP/      X          (3)
      COMMON /PREP/      XBR
      COMMON /OBS/       AVALS      (14)
      COMMON /OBS/       CLEAR      (7,14,5)
      COMMON /OBS/       FOO        (7,14,5)
      COMMON /OBS/       FOOMAX     (7,14,5)
      COMMON /OBS/       HOVALS     (7)
      COMMON /OBS/       NANG
      COMMON /OBS/       NOHGT
      COMMON /OBS/       NWDTH
      COMMON /OBS/       WVALS      (5)
      COMMON /DERIVE/    ADT        (9)
      COMMON /DERIVE/    NOGOBF
      COMMON /DERIVE/    CAREA
      COMMON /DERIVE/    COWB
      COMMON /DERIVE/    COWP
      COMMON /DERIVE/    COWPB      (2Ø)
      COMMON /DERIVE/    FA         (2Ø,3)
```

```
      COMMON /DERIVE/    FAT        (9)
      COMMON /DERIVE/    FAT1       (9)
      COMMON /DERIVE/    FB         (2Ø,3)
      COMMON /DERIVE/    FC         (2Ø,3)
      COMMON /DERIVE/    FNT        (9)
      COMMON /DERIVE/    FØM
      COMMON /DERIVE/    FØMMAX
      COMMON /DERIVE/    FØRMX      (3)
      COMMON /DERIVE/    IFLOAT
      COMMON /DERIVE/    IMAX       (3)
      COMMON /DERIVE/    ISAFE      (3)
      COMMON /DERIVE/    J
      COMMON /DERIVE/    MAXI
      COMMON /DERIVE/    BEMX       (3)
      COMMON /DERIVE/    NEVERO
      COMMON /DERIVE/    CBSE
      COMMON /DERIVE/    PAV        (9)
      COMMON /DERIVE/    RTOWB
      COMMON /DERIVE/    RTOWNB
      COMMON /DERIVE/    RTOWNP
      COMMON /DERIVE/    RTOWP
      COMMON /DERIVE/    FTOWPB     (2Ø)
      COMMON /DERIVE/    RTOWT      (2Ø)
      COMMON /DERIVE/    STRACT     (2Ø,3,3)
      COMMON /DERIVE/    SRFO       (9)
      COMMON /DERIVE/    SRFV       (9)
      COMMON /DERIVE/    STR        (3,9)
      COMMON /DERIVE/    TBF        (3)
      COMMON /DERIVE/    TBEN       (9)
      COMMON /DERIVE/    TRESIS     (3,9)
      COMMON /DERIVE/    VA         (3,9)
      COMMON /DERIVE/    NOGOVA     (3,9)
      COMMON /DERIVE/    VAVOID     (3,9)
      COMMON /DERIVE/    VBO        (3,9)
      COMMON /DERIVE/    IMPACT     (9)
      COMMON /DERIVE/    VELV       (3)
      COMMON /DERIVE/    VFMAX      (3)
      COMMON /DERIVE/    VG         (2Ø,3,3)
      COMMON /DERIVE/    NOGOVO     (3,9)
      COMMON /DERIVE/    VMAX
      COMMON /DERIVE/    VMAX1      (3)
      COMMON /DERIVE/    VMAX2      (3,9)
      COMMON /DERIVE/    VOLA
      COMMON /DERIVE/    VOVER      (3,9)
      COMMON /DERIVE/    VRID
      COMMON /DERIVE/    VSEL
      COMMON /DERIVE/    VSEL1      (3)
      COMMON /DERIVE/    VSEL2      (3,9)
      COMMON /DERIVE/    VSOIL      (3,9)
      COMMON /DERIVE/    VTT        (3,9)
      COMMON /DERIVE/    VXT        (3,9)
      COMMON /DERIVE/    NOGONO
```



```
      COMMON /DERIVE/    WRATIO
      COMMON /TERRAN/    AA
      COMMON /TERRAN/    ACTRMS
      COMMON /TERRAN/    AREA
      COMMON /TERRAN/    AREAC
      COMMON /TERRAN/    CI
      COMMON /TERRAN/    DIST
      COMMON /TERRAN/    EANG
      COMMON /TERRAN/    ECF
      COMMON /TERRAN/    ELEV
      COMMON /TERRAN/    FMU       (3)
      COMMON /TERRAN/    GRADE
      COMMON /TERRAN/    IOBS
      COMMON /TERRAN/    IOST
      COMMON /TERRAN/    IROAD
      COMMON /TERRAN/    IS        (9)
      COMMON /TERRAN/    IST
      COMMON /TERRAN/    ITUT
      COMMON /TERRAN/    NI
      COMMON /TERRAN/    NTU
      COMMON /TERRAN/    OAW
      COMMON /TERRAN/    OBAA
      COMMON /TERRAN/    OBH
      COMMON /TERRAN/    OBL
      COMMON /TERRAN/    OBS
      COMMON /TERRAN/    OBW
      COMMON /TERRAN/    OBMINW
      COMMON /TERRAN/    OBIA
      COMMON /TERRAN/    NWASHO
      COMMON /TERRAN/    RADC
      COMMON /TERRAN/    RCI
      COMMON /TERRAN/    RCIC      (4)
      COMMON /TERRAN/    RCURV     (11)
      COMMON /TERRAN/    RD
      COMMON /TERRAN/    RDA       (12)
      COMMON /TERRAN/    S         (9)
      COMMON /TERRAN/    SC        (9)
      COMMON /TERRAN/    SDL       (9)
      COMMON /TERRAN/    SURFF
      COMMON /TERRAN/    TANPHI
      COMMON /TERRAN/    THETA     (3)
      COMMON /TERRAN/    VCURV     (4,11)
      COMMON /TERRAN/    WA
      COMMON /TERRAN/    WC
      COMMON /SCEN/      CCHES
      COMMON /SCEN/      VWALK
      COMMON /SCEN/      CCLMAX
      COMMON /SCEN/      GAMMA
      COMMON /SCEN/      IGVER
      COMMON /SCEN/      ISEASN
      COMMON /SCEN/      ISURF
      COMMON /SCEN/      ISNOW
```

```
COMMON /SCEN/      KII1
COMMON /SCEN/      KII2
COMMON /SCEN/      KII3
COMMON /SCEN/      KII4
COMMON /SCEN/      KII5
COMMON /SCEN/      KII6
COMMON /SCEN/      KII7
COMMON /SCEN/      KII8
COMMON /SCEN/      KII9
COMMON /SCEN/      KII1Ø
COMMON /SCEN/      KII11
COMMON /SCEN/      KII12
COMMON /SCEN/      KII13
COMMON /SCEN/      KII14
COMMON /SCEN/      KII15
COMMON /SCEN/      KII16
COMMON /SCEN/      KII17
COMMON /SCEN/      KMAP
COMMON /SCEN/      KSCEN
COMMON /SCEN/      KIPP
COMMON /SCEN/      KVEH
COMMON /SCEN/      KIV1
COMMON /SCEN/      KIV2
COMMON /SCEN/      KIV3
COMMON /SCEN/      KIV4
COMMON /SCEN/      KIV5
COMMON /SCEN/      KIV6
COMMON /SCEN/      KIV7
COMMON /SCEN/      KIV8
COMMON /SCEN/      KIV9
COMMON /SCEN/      KIV1Ø
COMMON /SCEN/      KIV11
COMMON /SCEN/      KIV12
COMMON /SCEN/      KIV13
COMMON /SCEN/      KIV14
COMMON /SCEN/      KIV15
COMMON /SCEN/      KIV16
COMMON /SCEN/      KIV17
COMMON /SCEN/      KIV18
COMMON /SCEN/      KIV19
COMMON /SCEN/      KIV2Ø
COMMON /SCEN/      KIV21
COMMON /SCEN/      LAC
INTEGER            DETAIL
COMMON /SCEN/      DETAIL
COMMON /SCEN/      MAP
COMMON /SCEN/      MAPG
COMMON /SCEN/      MONTH
COMMON /SCEN/      NOPP
COMMON /SCEN/      NSLIP
COMMON /SCEN/      NTRAV
COMMON /SCEN/      NAUX
```

```
      COMMON /SCEN/      PHI
      COMMON /SCEN/      REACT
      COMMON /SCEN/      RBFOG
      INTEGER            SEARCH
      COMMON /SCEN/      SEARCH
      COMMON /SCEN/      SETYPC
      COMMON /SCEN/      VBRAKE
      COMMON /SCEN/      VISMNV
      COMMON /SCEN/      VLIM
      COMMON /SCEN/      ZSNOW
C     2. INITILIZE PROGRAM VARIABLES
      KBUFF  =0
      LUN1   =  1
      LUN2   =  2
      LUN3   =  3
      LUN4   =  4
      LUN5   =  5
      LUN10  = 10
      MD     =  2
      DOWN   =  3
      EFF    =  2
      FORCE  =  2
      GR     =  1
      LEVEL  =  2
      MN     =  1
      RPM    =  1
      SPEED  =  1
      SR     =  2
      TORCUE =  2
      TR     =  1
      UP     =  1
      MX     =  3
C     3. INITIALIZE I/O CHANNELS
      REWIND LUN2
      REWIND LUN3
      REWIND LUN4
      REWIND LUN5
      REWIND LUN10
C     4. REAC SCENARIO PARAMETERS
      CALL SCN
C     5. REAC VEHICLE PARAMETERS
      CALL VEH(KVEH)
C     6. EXECUTE VEHICLE PREPROCESSOR
      CALL VPP
C     7. EXECUTE TERRAIN TRANSLATOR
2000  CONTINUE
      CALL TERTL
C     8. EXECUTE AREAL OR ROAD MODULE
      IF( ITUT .LE. 2) CALL AREAL
      IF( ITUT .GE. 11) CALL ROAD
C     9. OUTPUT REQUESTED INFORMATION
      IF( IEOF .EQ. 1) GO TO 4000
```

```
            CALL BUFF0
            IF((SEARCH .EQ.1 ) .OR. (DETAIL .EQ. 10)) GO TO 4000
                GO TO 2000
4000      CONTINUE
          END
      SUBROUTINE SCN
C     ---------------------
C     SCENARIO INPUT ROUTINE
C     ---------------------
C
C     1. LABLED COMMON ASSIGNMENTS
          COMMON /IO/          IEOF
          COMMON /IO/          KBUFF
          COMMON /IO/          LUN1
          COMMON /IO/          LUN2
          COMMON /IO/          LUN3
          COMMON /IO/          LUN4
          COMMON /IO/          LUN5
          COMMON /IO/          LUN6
          COMMON /IO/          LUN7
          COMMON /IO/          LUN8
          COMMON /IO/          LUN9
          COMMON /IO/          LUN10
          COMMON /SCEN/        COHES
          COMMON /SCEN/        VWALK
          COMMON /SCEN/        DCLMAX
          COMMON /SCEN/        GAMMA
          COMMON /SCEN/        IOVER
          COMMON /SCEN/        ISEASN
          COMMON /SCEN/        ISURF
          COMMON /SCEN/        ISNOW
          COMMON /SCEN/        KII1
          COMMON /SCEN/        KII2
          COMMON /SCEN/        KII3
          COMMON /SCEN/        KII4
          COMMON /SCEN/        KII5
          COMMON /SCEN/        KII6
          COMMON /SCEN/        KII7
          COMMON /SCEN/        KII8
          COMMON /SCEN/        KII9
          COMMON /SCEN/        KII10
          COMMON /SCEN/        KII11
          COMMON /SCEN/        KII12
          COMMON /SCEN/        KII13
          COMMON /SCEN/        KII14
          COMMON /SCEN/        KII15
          COMMON /SCEN/        KII16
          COMMON /SCEN/        KII17
          COMMON /SCEN/        KMAP
          COMMON /SCEN/        KSCEN
          COMMON /SCEN/        KTPP
          COMMON /SCEN/        KVEH
```

```
      COMMON /SCEN/       KIV1
      COMMON /SCEN/       KIV2
      COMMON /SCEN/       KIV3
      COMMON /SCEN/       KIV4
      COMMON /SCEN/       KIV5
      COMMON /SCEN/       KIV6
      COMMON /SCEN/       KIV7
      COMMON /SCEN/       KIV8
      COMMON /SCEN/       KIV9
      COMMON /SCEN/       KIV10
      COMMON /SCEN/       KIV11
      COMMON /SCEN/       KIV12
      COMMON /SCEN/       KIV13
      COMMON /SCEN/       KIV14
      COMMON /SCEN/       KIV15
      COMMON /SCEN/       KIV16
      COMMON /SCEN/       KIV17
      COMMON /SCEN/       KIV18
      COMMON /SCEN/       KIV19
      COMMON /SCEN/       KIV20
      COMMON /SCEN/       KIV21
      COMMON /SCEN/       LAC
      INTEGER             DETAIL
      COMMON /SCEN/       DETAIL
      COMMON /SCEN/       MAP
      COMMON /SCEN/       MAPG
      COMMON /SCEN/       MONTH
      COMMON /SCEN/       NOPP
      COMMON /SCEN/       NSLIP
      COMMON /SCEN/       NTRAV
      COMMON /SCEN/       NTUX
      COMMON /SCEN/       PHI
      COMMON /SCEN/       REACT
      COMMON /SCEN/       RDFOG
      INTEGER             SEARCH
      COMMON /SCEN/       SEARCH
      COMMON /SCEN/       SFTYPC
      COMMON /SCEN/       VBRAKE
      COMMON /SCEN/       VISMNV
      COMMON /SCEN/       VLIM
      COMMON /SCEN/       ZSNOW
C     2. SCENARIO INPUT PARAMETERS
      NAMELIST /SCENAR/
     +    COHES  ,DCLMAX ,GAMMA  ,IOVER  ,ISEASN ,ISURF
     +   ,ISNOW  ,LAC    ,MAP    ,MAPG   ,MONTH  ,NOPP
     +   ,NSLIP  ,NTRAV  ,NTUX   ,PHI    ,REACT  ,RDFOG
     +   ,SFTYPC ,VBRAKE ,VISMNV ,VLIM   ,ZSNOW
      NAMELIST /CONTRL/
     +     DETAIL   ,KSCEN    ,KVEH     ,KII1     ,KII2
     +    ,KII3     ,KII4     ,KII5     ,KII6     ,KII7
     +    ,KII8     ,KII9     ,KII10    ,KII11    ,KII12
     +    ,KII13    ,KII14    ,KII15    ,KII16    ,KII17
```

```
     +      ,KMAP     ,KTPP     ,KIV1     ,KIV2     ,KIV3
     +      ,KIV4     ,KIV5     ,KIV6     ,KIV7     ,KIV8
     +      ,KIV9     ,KIV10    ,KIV11    ,KIV12    ,KIV13
     +      ,KIV14    ,KIV15    ,KIV16    ,KIV17    ,KIV18
     +      ,KIV19    ,KIV20    ,KIV21    ,NTUX     ,SEARCH
C        3. INITIALIZE SCENARIO VARIABLES
         COHES  =     .05
         VWALK  =    4.00
         DCLMAX =     .50
         GAMMA  =     .20
         IOVER  =       9
         ISEASN =       1
         ISNOW  =       0
         ISURF  =       1
         LAC    =       1
         DETAIL =       1
         MAP    =      71
         MAPG   =       1
         NOPP   =       0
         NSLIP  =       0
         NTRAV  =       3
         PHI    =   21.00
         REACT  =     .50
         RDFCG  =1000.00
         SEARCH =       0
         SFTYPC =   90.00
         VBRAKE =    5.00
         VISMNV =    2.00
         VLIM   =   55.00
         ZSNOW  =    3.00
C      4. READ CONTROL VARIABLES
         READ(LUN4,CONTRL)
         IF( DETAIL .EQ.  1)  GO TO 330
         IF( DETAIL .EQ.  2)  GO TO 310
         IF( DETAIL .EQ.  3)  GO TO 310
         IF( DETAIL .EQ.  4)  GO TO 310
         IF( DETAIL .EQ.  5)  GO TO 320
         IF( DETAIL .EQ. 10)  GO TO 320
         WRITE(LUN1,3000) DETAIL
         STOP 1
         GO TO 320
310         CONTINUE
            KSCEN  = 1
            KVEH   = 1
            GO TO 330
320      CONTINUE
            KSCEN  = 1
            KVEH   = 1
            KII1   = 1
            KII2   = 1
            KII3   = 1
            KII4   = 1
```

```
            KII5    = 1
            KII6    = 1
            KII7    = 1
            KII8    = 1
            KII9    = 1
            KII10   = 1
            KII11   = 1
            KII12   = 1
            KII13   = 1
            KII14   = 1
            KII15   = 1
            KII16   = 1
            KII17   = 1
            KMAP    = 1
            KTPP    = 1
            KIV1    = 1
            KIV2    = 1
            KIV3    = 1
            KIV4    = 1
            KIV5    = 1
            KIV6    = 1
            KIV7    = 1
            KIV8    = 1
            KIV9    = 1
            KIV10   = 1
            KIV11   = 1
            KIV12   = 1
            KIV13   = 1
            KIV14   = 1
            KIV15   = 1
            KIV16   = 1
            KIV17   = 1
            KIV18   = 1
            KIV19   = 1
            KIV20   = 1
            KIV21   = 1
330     CONTINUE
        IF(KSCEN .EQ. 1) WRITE(LUN1,CONTRL)
C     5. READ SCENARIO VARIABLES
        READ(LUN4,SCENAR)
        IF(KSCEN .EQ. 1) WRITE(LUN1,SCENAR)
C     6. UNITS CONVERSION
        PHI    = PHI*3.14159265/180.
        VBRAKE = VBRAKE*17.6
        VISMNV = VISMNV*17.6
        VLIM   = VLIM*17.6
        VWALK  = VWALK*17.6
3000    FORMAT(1X,31HINVALID SPECIFICATION OF DETAIL,I2)
        RETURN
        END
      SUBROUTINE VEH(KVEH)
C     ------------------------------
```

```
C       VEHICLE PARAMETER INPUT ROUTINE
C       --------------------------------
C
C       1. LABLED COMMON ASSIGNMENTS
           COMMON /IO/          IEOF
           COMMON /IO/          KBUFF
           COMMON /IO/          LUN1
           COMMON /IO/          LUN2
           COMMON /IO/          LUN3
           COMMON /IO/          LUN4
           COMMON /IO/          LUN5
           COMMON /IO/          LUN6
           COMMON /IO/          LUN7
           COMMON /IO/          LUN8
           COMMON /IO/          LUN9
           COMMON /IO/          LUN10
           COMMON /VEHICL/      ACD
           COMMON /VEHICL/      ASHOE      (20)
           COMMON /VEHICL/      AWGC
           COMMON /VEHICL/      AXLSP      (20)
           COMMON /VEHICL/      CD
           COMMON /VEHICL/      CGH
           COMMON /VEHICL/      CGLAT
           COMMON /VEHICL/      CGR
           COMMON /VEHICL/      CID
           COMMON /VEHICL/      CL
           COMMON /VEHICL/      CLRMIN     (20)
           COMMON /VEHICL/      CONV1      (2,25)
           COMMON /VEHICL/      CONV2      (2,25)
           COMMON /VEHICL/      CSLCT      (20,3)
           COMMON /VEHICL/      DBAW       (20)
           COMMON /VEHICL/      DRAFT
           COMMON /VEHICL/      ENGINE     (2,50)
           COMMON /VEHICL/      EXEHGT
           COMMON /VEHICL/      FD         (2)
           COMMON /VEHICL/      FORDD
           COMMON /VEHICL/      GROUSH     (20)
           COMMON /VEHICL/      HWALLS     (25)
           COMMON /VEHICL/      IAPG
           COMMON /VEHICL/      IB         (20)
           COMMON /VEHICL/      IG         (20)
           REAL                 IDIESL
           COMMON /VEHICL/      IDIESL
           COMMON /VEHICL/      IENGIN
           COMMON /VEHICL/      IR         (20)
           COMMON /VEHICL/      ICONST     (20)
           COMMON /VEHICL/      ICONV1
           COMMON /VEHICL/      ICONV2
           COMMON /VEHICL/      IPOWER
           COMMON /VEHICL/      IT         (20)
           COMMON /VEHICL/      ITCASE
           COMMON /VEHICL/      ITRAN
```

```
      COMMON /VEHICL/    ITVAR
      COMMON /VEHICL/    LOCKUP
      COMMON /VEHICL/    MAXIPR
      COMMON /VEHICL/    MAXL
      COMMON /VEHICL/    NAMBLY
      COMMON /VEHICL/    NBOGIE     (20)
      COMMON /VEHICL/    NCHAIN     (20)
      REAL               NCYL
      COMMON /VEHICL/    NCYL
      REAL               NENG
      COMMON /VEHICL/    NENG
      COMMON /VEHICL/    HPNET
      COMMON /VEHICL/    NEL        (20)
      COMMON /VEHICL/    NOR
      COMMON /VEHICL/    NWVALS
      COMMON /VEHICL/    NRAD       (20)
      COMMON /VEHICL/    NSVALS
      COMMON /VEHICL/    NVEH       (20)
      COMMON /VEHICL/    NWHL       (20)
      COMMON /VEHICL/    NWR
      COMMON /VEHICL/    PBF
      COMMON /VEHICL/    PBHT
      COMMON /VEHICL/    PEA
      COMMON /VEHICL/    POWER      (2,20)
      COMMON /VEHICL/    QMAX
      COMMON /VEHICL/    ROGAM      (20)
      COMMON /VEHICL/    REVM       (20)
      COMMON /VEHICL/    ROMW       (20)
      COMMON /VEHICL/    RNS        (20)
      COMMON /VEHICL/    RW         (20)
      COMMON /VEHICL/    SAE
      COMMON /VEHICL/    SAI
      COMMON /VEHICL/    SECTH      (20)
      COMMON /VEHICL/    SECTW      (20)
      COMMON /VEHICL/    SVALS      (25)
      COMMON /VEHICL/    TCASE      (2)
      COMMON /VEHICL/    TE
      COMMON /VEHICL/    TOLY       (20)
      COMMON /VEHICL/    TRSI       (20,3)
      COMMON /VEHICL/    TGIND
      COMMON /VEHICL/    TRAKLN     (20)
      COMMON /VEHICL/    TRAKWD     (20)
      COMMON /VEHICL/    TRANS      (2,20)
      COMMON /VEHICL/    VAA
      COMMON /VEHICL/    VDA
      COMMON /VEHICL/    VGS
      COMMON /VEHICL/    VOCB       (25)
      COMMON /VEHICL/    VOOBS      (25)
      COMMON /VEHICL/    VRIDE      (20,3)
      COMMON /VEHICL/    VSS
      COMMON /VEHICL/    VSSAXP
      COMMON /VEHICL/    WC
```

```
      COMMON /VEHICL/    WDAXP
      COMMON /VEHICL/    WDPTH     (20)
      COMMON /VEHICL/    WDTH
      COMMON /VEHICL/    WGHT      (20)
      COMMON /VEHICL/    WRAT      (20)
      COMMON /VEHICL/    WRFORD
      COMMON /VEHICL/    WT        (20)
      COMMON /VEHICL/    WTE       (20)
      COMMON /VEHICL/    WWAXP
      COMMON /VEHICL/    XBRCOF
      COMMON /VEHICL/    WB
      COMMON /VEHICL/    LOCDIF
      COMMON /VEHICL/    SHF
      COMMON /OBS/       AVALS     (14)
      COMMON /OBS/       CLEAR     (7,14,5)
      COMMON /OBS/       FOO       (7,14,5)
      COMMON /OBS/       FOOMAX    (7,14,5)
      COMMON /OBS/       HOVALS    (7)
      COMMON /OBS/       NANG
      COMMON /OBS/       NOHGT
      COMMON /OBS/       NWDTH
      COMMON /OBS/       WVALS     (5)
C     2. VEHICLE INPUT PARAMETERS
      NAMELIST /VEHICLE/
     +    ACD     ,ASHOE  ,AVGC   ,AXLSP  ,CD     ,CGH    ,CGLAT
     +   ,CGR     ,CID    ,CL     ,CLRMIN ,CONV1  ,CONV2  ,DFLCT
     +   ,CIAW    ,DRAFT  ,ENGINE ,EYEHGT ,FD     ,FORDD  ,GROUSH
     +   ,HPNET   ,HVALS  ,IAPG   ,IB     ,ID     ,IDIESL ,IENGIN
     +   ,ICONST  ,ICONV1 ,ICONV2 ,IP     ,IPOWER ,IT     ,ITCASE
     +   ,ITRAN   ,ITVAR  ,LOCDIF ,LOCKUP ,MAXIPR ,MAXL   ,NAMBLY
     +   ,NBOGIE  ,NCHAIN ,NCYL   ,NENG   ,NFL    ,NGR    ,NHVALS
     +   ,NPAD    ,NSVALS ,NVEH   ,NWHL   ,NWR    ,PBF    ,PBHT
     +   ,PFA     ,POWER  ,QMAX   ,RDIAM  ,REVM   ,RIMW   ,RMS
     +   ,RW      ,SAE    ,SAI    ,SECTH  ,SECTW  ,SVALS  ,TCASE
     +   ,TL      ,TPLY   ,TPSI   ,TQIND  ,TRAKLN ,TRAKWD ,TRANS
     +   ,VAA     ,VDA    ,VFS    ,VOOB   ,VOOBS  ,VRIDE  ,VSS
     +   ,VSSAXP  ,WC     ,WDAXP  ,WDPTH  ,WDTH   ,WGHT   ,WI
     +   ,WRAT    ,WRFORD ,WT     ,WTE    ,WWAXP  ,XBRCOF
C     NAMELIST /BUMP/
C    +    AVALS     ,CLEAR     ,FOO       ,FOOMAX    ,HOVALS
C    +   ,NANG      ,NOHGT     ,NWDTH     ,WVALS
C     3. READ VEHICLE PARAMETERS
      READ(LUN3,VEHICLE)
      IF(KVEH .EQ. 1) WRITE(LUN1,VEHICLE)
C     ------------------------------
C     OBSTACLE INTERFERENCE HISTORY
C     ------------------------------
      READ(LUN3,1000) DUMMY
      READ(LUN3,2000) NOHGT
      READ(LUN3,1000) DUMMY
      READ(LUN3,2000) NANG
      READ(LUN3,1000) DUMMY
```

```
      READ(LUN3,2000) NWDTH
      READ(LUN3,1000) DUMMY
      READ(LUN3,1000) DUMMY
1000  FORMAT(A1)
2000  FORMAT(5X,I2)
      DO 3000 NW=1,NWDTH
         DO 3010 NA=1,NANG
            DO 3020 NH=1,NOHGT
               READ(LUN3,4000)
     +            CLEAR(NH,NA,NW),
     +            FOOMAX(NH,NA,NW),
     +            FOO(NH,NA,NW),
     +            HOVALS(NH),
     +            AVALS(NA),
     +            WVALS(NW)
4000           FORMAT(F7.2,F10.1,F10.1,F10.2,F10.2,F10.2)
3020        CONTINUE
3010     CONTINUE
3000  CONTINUE
C     READ(LUN3,BUMP)
C     IF(KVEH .EQ. 1) WRITE(LUN1,BUMP)
      RETURN
      END
      SUBROUTINE VPP
C     --------------------
C     VEHICLE PREPROCESSOR
C     --------------------
C
C     1. LABLED COMMON ASSIGNMENTS
      COMMON /IO/          IEOF
      COMMON /IO/          KBUFF
      COMMON /IO/          LUN1
      COMMON /IO/          LUN2
      COMMON /IO/          LUN3
      COMMON /IO/          LUN4
      COMMON /IO/          LUN5
      COMMON /IO/          LUN6
      COMMON /IO/          LUN7
      COMMON /IO/          LUN8
      COMMON /IO/          LUN9
      COMMON /IO/          LUN10
      INTEGER              MO
      COMMON /INDEX/       MO
      INTEGER              DOWN
      COMMON /INDEX/       DOWN
      INTEGER              EFF
      COMMON /INDEX/       EFF
      INTEGER              FORCE
      COMMON /INDEX/       FORCE
      INTEGER              GR
      COMMON /INDEX/       GR
      COMMON /INDEX/       LEVEL
```

```
      INTEGER            MN
      COMMON /INDEX/     MN
      INTEGER            RPM
      COMMON /INDEX/     RPM
      INTEGER            SPEED
      COMMON /INDEX/     SPEED
      INTEGER            SR
      COMMON /INDEX/     SR
      INTEGER            TR
      COMMON /INDEX/     TR
      INTEGER            TORQUE
      COMMON /INDEX/     TORQUE
      INTEGER            UR
      COMMON /INDEX/     UR
      INTEGER            MX
      COMMON /INDEX/     MX
      COMMON /VEHICL/    AOD
      COMMON /VEHICL/    ASHOE     (20)
      COMMON /VEHICL/    AVGC
      COMMON /VEHICL/    AXLSP     (20)
      COMMON /VEHICL/    CD
      COMMON /VEHICL/    CGH
      COMMON /VEHICL/    CGLAT
      COMMON /VEHICL/    CGR
      COMMON /VEHICL/    CID
      COMMON /VEHICL/    CM
      COMMON /VEHICL/    CLRMIN    (20)
      COMMON /VEHICL/    CONV1     (2,25)
      COMMON /VEHICL/    CONV2     (2,25)
      COMMON /VEHICL/    DELCT     (20,3)
      COMMON /VEHICL/    DRAW      (20)
      COMMON /VEHICL/    DRAFT
      COMMON /VEHICL/    ENGINE    (2,50)
      COMMON /VEHICL/    EXBHGT
      COMMON /VEHICL/    FD        (2)
      COMMON /VEHICL/    FORDD
      COMMON /VEHICL/    GROUSH    (20)
      COMMON /VEHICL/    HVALS     (25)
      COMMON /VEHICL/    IAPG
      COMMON /VEHICL/    IB        (20)
      COMMON /VEHICL/    ID        (20)
      REAL               IDIESL
      COMMON /VEHICL/    IDIESL
      COMMON /VEHICL/    IENGIN
      COMMON /VEHICL/    IP        (20)
      COMMON /VEHICL/    ICONST    (20)
      COMMON /VEHICL/    ICONV1
      COMMON /VEHICL/    ICONV2
      COMMON /VEHICL/    IPOWER
      COMMON /VEHICL/    IS        (20)
      COMMON /VEHICL/    ITCASE
      COMMON /VEHICL/    ITRAN
```

```
      COMMON /VEHICL/    IDVAR
      COMMON /VEHICL/    LOCKUP
      COMMON /VEHICL/    MAXIPR
      COMMON /VEHICL/    MAXL
      COMMON /VEHICL/    NAMBLY
      COMMON /VEHICL/    NBOGIE     (20)
      COMMON /VEHICL/    NCHAIN     (20)
      REAL               NCYL
      COMMON /VEHICL/    NCYL
      REAL               NENG
      COMMON /VEHICL/    NENG
      COMMON /VEHICL/    HPNET
      COMMON /VEHICL/    NGL        (20)
      COMMON /VEHICL/    NGR
      COMMON /VEHICL/    NHVALS
      COMMON /VEHICL/    NRAD       (20)
      COMMON /VEHICL/    NSVALS
      COMMON /VEHICL/    NWEH       (20)
      COMMON /VEHICL/    NWHL       (20)
      COMMON /VEHICL/    NWR
      COMMON /VEHICL/    PBF
      COMMON /VEHICL/    PBHT
      COMMON /VEHICL/    PFA
      COMMON /VEHICL/    POWER      (2,201)
      COMMON /VEHICL/    QMAX
      COMMON /VEHICL/    RDIAM      (20)
      COMMON /VEHICL/    REVM       (20)
      COMMON /VEHICL/    RIMW       (20)
      COMMON /VEHICL/    RMS        (20)
      COMMON /VEHICL/    RW         (20)
      COMMON /VEHICL/    SAE
      COMMON /VEHICL/    SAI
      COMMON /VEHICL/    SECTH      (20)
      COMMON /VEHICL/    SECTW      (20)
      COMMON /VEHICL/    SVALS      (25)
      COMMON /VEHICL/    TCASE      (2)
      COMMON /VEHICL/    TE
      COMMON /VEHICL/    TRLY       (20)
      COMMON /VEHICL/    TRSI       (20,3)
      COMMON /VEHICL/    TGIND
      COMMON /VEHICL/    TRAKLN     (20)
      COMMON /VEHICL/    TRAKWD     (20)
      COMMON /VEHICL/    TRANS      (2,20)
      COMMON /VEHICL/    VAA
      COMMON /VEHICL/    VDA
      COMMON /VEHICL/    VES
      COMMON /VEHICL/    VGCB       (25)
      COMMON /VEHICL/    VOOBS      (25)
      COMMON /VEHICL/    VRIDE      (20,3)
      COMMON /VEHICL/    VSS
      COMMON /VEHICL/    VSMAXP
      COMMON /VEHICL/    WC
```

```
      COMMON /VEHICL/    WDAXP
      COMMON /VEHICL/    WDPTH     (2Ø)
      COMMON /VEHICL/    WDTH
      COMMON /VEHICL/    WGHT      (2Ø)
      COMMON /VEHICL/    WRAT      (2Ø)
      COMMON /VEHICL/    WRFORD
      COMMON /VEHICL/    WT        (2Ø)
      COMMON /VEHICL/    WTE       (2Ø)
      COMMON /VEHICL/    WWAXP
      COMMON /VEHICL/    XBRCOF
      COMMON /VEHICL/    WJ
      COMMON /VEHICL/    LOCDIF
      COMMON /VEHICL/    SHF
      COMMON /PREP/      A         (3,4)
      COMMON /PREP/      ATF       (2Ø)
      COMMON /PREP/      BTF       (2Ø)
      COMMON /PREP/      CHARLN    (2Ø,3)
      COMMON /PREP/      CRGCFG    (3)
      COMMON /PREP/      CRFCCG    (3)
      COMMON /PREP/      CRFCG     (2Ø,3)
      COMMON /PREP/      CRFFG     (2Ø,3)
      COMMON /PREP/      CTF       (2Ø)
      COMMON /PREP/      CRAT      (2Ø,3)
      COMMON /PREP/      GCA       (2Ø,3)
      COMMON /PREP/      GCW
      COMMON /PREP/      GCWB
      COMMON /PREP/      GCWNB
      COMMON /PREP/      GCWNP
      COMMON /PREP/      GCWP
      COMMON /PREP/      HRT
      COMMON /PREP/      NDF       (38)
      COMMON /PREP/      NVEHC
      COMMON /PREP/      PNTE
      COMMON /PREP/      R         (3)
      COMMON /PREP/      RNX       (2Ø)
      COMMON /PREP/      RR
      COMMON /PREP/      TRACTF    (2Ø,5)
      COMMON /PREP/      TRAPSI    (3)
      COMMON /PREP/      VCICG     (2Ø,3)
      COMMON /PREP/      VCIFG     (2Ø,3)
      COMMON /PREP/      VCIMUK    (2Ø)
      COMMON /PREP/      VGV       (2Ø,5)
      COMMON /PREP/      VT        (2Ø,3)
      COMMON /PREP/      VTIRE     (3)
      COMMON /PREP/      WTMAX
      COMMON /PREP/      X         (3)
      COMMON /PREP/      WBR
      COMMON /SCEN/      COHES
      COMMON /SCEN/      VWALK
      COMMON /SCEN/      CCLMAX
      COMMON /SCEN/      GAMMA
      COMMON /SCEN/      IOVER
```

```
      COMMON /SCEN/      ISEASN
      COMMON /SCEN/      ISURF
      COMMON /SCEN/      ISNOW
      COMMON /SCEN/      KII1
      COMMON /SCEN/      KII2
      COMMON /SCEN/      KII3
      COMMON /SCEN/      KII4
      COMMON /SCEN/      KII5
      COMMON /SCEN/      KII6
      COMMON /SCEN/      KII7
      COMMON /SCEN/      KII8
      COMMON /SCEN/      KII9
      COMMON /SCEN/      KII10
      COMMON /SCEN/      KII11
      COMMON /SCEN/      KII12
      COMMON /SCEN/      KII13
      COMMON /SCEN/      KII14
      COMMON /SCEN/      KII15
      COMMON /SCEN/      KII16
      COMMON /SCEN/      KII17
      COMMON /SCEN/      KMAP
      COMMON /SCEN/      KSCEN
      COMMON /SCEN/      KTOP
      COMMON /SCEN/      KVEH
      COMMON /SCEN/      KIV1
      COMMON /SCEN/      KIV2
      COMMON /SCEN/      KIV3
      COMMON /SCEN/      KIV4
      COMMON /SCEN/      KIV5
      COMMON /SCEN/      KIV6
      COMMON /SCEN/      KIV7
      COMMON /SCEN/      KIV8
      COMMON /SCEN/      KIV9
      COMMON /SCEN/      KIV10
      COMMON /SCEN/      KIV11
      COMMON /SCEN/      KIV12
      COMMON /SCEN/      KIV13
      COMMON /SCEN/      KIV14
      COMMON /SCEN/      KIV15
      COMMON /SCEN/      KIV16
      COMMON /SCEN/      KIV17
      COMMON /SCEN/      KIV18
      COMMON /SCEN/      KIV19
      COMMON /SCEN/      KIV20
      COMMON /SCEN/      KIV21
      COMMON /SCEN/      LAC
      INTEGER            CETAIL
      COMMON /SCEN/      CETAIL
      COMMON /SCEN/      MAP
      COMMON /SCEN/      MAPG
      COMMON /SCEN/      MONTH
      COMMON /SCEN/      NOPP
```

```
      COMMON /SCEN/      NSLIP
      COMMON /SCEN/      NTRAV
      COMMON /SCEN/      NTUX
      COMMON /SCEN/      PHI
      COMMON /SCEN/      REACT
      COMMON /SCEN/      RDFOG
      INTEGER            SEARCH
      COMMON /SCEN/      SEARCH
      COMMON /SCEN/      SETYPC
      COMMON /SCEN/      VBRAKE
      COMMON /SCEN/      VISMNV
      COMMON /SCEN/      VLIM
      COMMON /SCEN/      ZSNCW
C     2. ALGORITHM
      CALL II1(
     + CONV1  ,ENGINE ,IAPG   ,ICONV1 ,IENGIN ,IPOWER ,MAXIPR ,MAXL
     + ,NHVALS ,NSVALS ,PFA    ,POWER  ,QMAX   ,RPM    ,SPEED  ,TORQUE
     + ,TQINC  ,VAA    ,VDA    ,VOCB   ,VOOBS  ,VRIDE  ,VSS )
      NAMELIST /XII1/
     + CONV1  ,ENGINE ,IAPG   ,ICONV1 ,IENGIN ,IPOWER ,MAXIPR ,MAXL
     + ,NHVALS ,NSVALS ,PFA    ,POWER  ,QMAX   ,RPM    ,SPEED  ,TORQUE
     + ,TQIND  ,VAA    ,VDA    ,VOCB   ,VOOBS  ,VRIDE  ,VSS
      IF(KII1 .EQ. 1) WRITE(LUN1,XII1)
      CALL II2(
     + GCW    ,GCWB   ,GCWNB  ,GCWNP  ,GCWP   ,IB     ,IP     ,NAMBLY
     + ,WGHT )
      NAMELIST /XII2/
     + GCW    ,GCWB   ,GCWNB  ,GCWNP  ,GCWP   ,IB     ,IP     ,NAMBLY
     + ,WGHT
      IF(KII2 .EQ. 1)
     +   WRITE(LUN1,XII2)
      CALL II3(
     + ICONST ,NAMBLY ,NVEH   ,NWHL   ,RDIAM  ,RIMW   ,SECTW  ,TPSI
     + ,VTIRE  ,WGHT )
      NAMELIST /XII3/
     + ICONST ,NAMBLY ,NVEH   ,NWHL   ,RDIAM  ,RIMW   ,SECTW  ,TPSI
     + ,VTIRE  ,WGHT
      IF(KII3 .EQ. 1)
     +   WRITE(LUN1,XII3)
      CALL II4(
     + NAMBLY ,PWTE   ,WT     ,WTE )
      NAMELIST /XII4/
     + NAMBLY ,PWTE   ,WT     ,WTE
      IF(KII4 .EQ. 1)
     +   WRITE(LUN1,XII4)
      CALL II5(
     + DFLCT  ,DRAT   ,NAMBLY ,NVEH   ,SECTH )
      NAMELIST /XII5/
     + DFLCT  ,DRAT   ,NAMBLY ,NVEH   ,SECTH
      IF(KII5 .EQ. 1)
     +   WRITE(LUN1,XII5)
      CALL II6(
```

```
     + CHARLN ,DFLCT  ,DIAW   ,NAMBLY ,NVEH   ,TRAKLN )
      NAMELIST /XII6/
     + CHARLN ,DFLCT  ,DIAW   ,NAMBLY ,NVEH   ,TRAKLN
      IF(KII6 .EQ. 1)
     +   WRITE(LUN1,XII6)
         CALL II7(
     + CHARLN ,GCA    ,NAMBLY ,NVEH   ,SECTW  ,TRAKWD )
      NAMELIST /XII7/
     + CHARLN ,GCA    ,NAMBLY ,NVEH   ,SECTW  ,TRAKWD
      IF(KII7 .EQ. 1)
     +   WRITE(LUN1,XII7)
         CALL II8(
     + CGLAT  ,ID     ,NAMBLY ,NVEH   ,SECTW  ,WT     ,WTMAX )
      NAMELIST /XII8/
     + CGLAT  ,ID     ,NAMBLY ,NVEH   ,SECTW  ,WT     ,WTMAX
      IF(KII8 .EQ. 1)
     +   WRITE(LUN1,XII8)
         CALL II9(
     + IP     ,NAMBLY ,NVEH   ,REVM   ,RR )
      NAMELIST /XII9/
     + IP     ,NAMBLY ,NVEH   ,REVM   ,RR
      IF( KII9 .EQ. 1 ) WRITE(LUN1,XII9)
         CALL II10(
     + IB     ,NAMBLY ,WGHT   ,XBRCOF ,XBR )
      NAMELIST /XII10/
     + IB     ,NAMBLY ,WGHT   ,XBRCOF ,XBR
      IF( KII10 .EQ. 1 ) WRITE(LUN1,XII10)
         CALL II11(
     + GCWP   ,HPNET  ,HPT )
      NAMELIST /XII11/
     + GCWP   ,HPNET  ,HPT
      IF( KII11 .EQ. 1 ) WRITE(LUN1,XII11)
         CALL II12(
     + ASHOE  ,CLRMIN ,CPFFG  ,DFLCT  ,DIAW   ,GROUSH ,HPT    ,IB
     + ,IP    ,ITVAR  ,NAMBLY ,NBCGIE ,NCHAIN ,NVEH   ,NWHL   ,SECTH
     + ,SECTW ,TRAKLN ,TRAKWD ,VCIFG  ,WGHT )
      NAMELIST /XII12/
     + ASHOE  ,CLRMIN ,CPFFG  ,DFLCT  ,DIAW   ,GROUSH ,HPT    ,IB
     + ,IP    ,ITVAR  ,NAMBLY ,NBCGIE ,NCHAIN ,NVEH   ,NWHL   ,SECTH
     + ,SECTW ,TRAKLN ,TRAKWD ,VCIFG  ,WGHT
      IF( KII12 .EQ. 1 ) WRITE(LUN1,XII12)
         CALL II13(
     + CPFCG  ,IB     ,IP     ,NAMBLY ,NVEH   ,NWHL
     + ,RDIAM ,SECTW  ,TPLY   ,TPSI   ,VCICG  ,WGHT )
      NAMELIST /XII13/
     + CPFCG  ,IB     ,IP     ,NAMBLY ,NVEH   ,NWHL
     + ,RDIAM ,SECTW  ,TPLY   ,TPSI   ,VCICG  ,WGHT
      IF( KII13 .EQ. 1 ) WRITE(LUN1,XII13)
         CALL II14(
     + DIAW   ,IB     ,IP     ,NAMBLY ,NVEH
     + ,NWHL  ,SECTW  ,TRAKLN ,TRAKWD ,VCIMUK ,WGHT )
      NAMELIST /XII14/
```

```
     + DIAW    ,IB      ,IP      ,NAMBLY ,NVEH
     + ,NWHL   ,SECTW   ,TRAKWN ,TRAKWD ,VCIMUK ,WGHT
      IF( KII14 .EQ. 1 ) WRITE(LUN1,XII14)
         CALL II15(
     + CPFCCG ,CPFCFG ,CPFCE   ,CPFFG   ,IB
     + ,IP     ,NAMBLY ,NVEH    ,NVEHC )
      NAMELIST /XII15/
     + CPFCCG ,CPFCFG ,CPFCE   ,CPFFG   ,IB
     + ,IP     ,NAMBLY ,NVEH    ,NVEHC
      IF( KII15 .EQ. 1 ) WRITE(LUN1,XII15)
         CALL II16(
     + ATF     ,BTF     ,CONV1   ,CONV2   ,CTF     ,EFF     ,ENGINE ,FD
     + ,FORCE  ,GR      ,IAPG    ,ICONV1 ,ICONV2 ,IENGIN ,IPOWER ,ITCASE
     + ,ITRAN  ,KII16   ,LOCKUP ,LUN1    ,MAPG
     + ,NGR    ,PE      ,POWER   ,RR      ,RPM     ,SPEED   ,SR      ,TCASE
     + ,TORQUE ,TQIND   ,TR      ,TRACTF ,TRANS   ,VGV     ,TOPSPD ,NPTS
     + ,IERROR )
         NAMELIST /XII16/
     + ATF     ,BTF     ,CONV1   ,CONV2   ,CTF     ,EFF     ,ENGINE ,FD
     + ,FORCE  ,GR      ,IAPG    ,ICONV1 ,ICONV2 ,IENGIN ,IERROR ,ITCASE
     + ,ITRAN  ,KII16   ,LOCKUP ,LUN1    ,MAPG
     + ,NGR    ,PE      ,POWER   ,RR      ,RPM     ,SPEED   ,SR      ,TCASE
     + ,TORQUE ,TQIND   ,TR      ,TRACTF ,TRANS   ,VGV     ,TOPSPD ,NPTS
     + ,IERROR
         IF( KII16 .EQ. 1 ) WRITE(LUN1,XII16)
         CALL II17(
     + CID     ,IDIESL ,GCW     ,NAMBLY ,NCYL    ,NENG
     + ,NGR    ,NVEH    ,QMAX    ,RMX     ,RR      ,TRACTF )
         NAMELIST /XII17/
     + CID     ,IDIESL ,NCYL    ,NENG    ,NVEH    ,QMAX    ,RMX
         IF(KII17 .EQ. 1) WRITE(LUN1,XII17)
           IF(( DETAIL .NE. 5 ) .AND. ( IERROR .NE. 1 )) GOTO 100
           CALL PLTSET(
     +     NPTS    ,VGV     ,NGR     ,ATF     ,BTF     ,CTF
     +     ,IPOWER ,POWER ,TOPSPD,LUN1)
           STOP 3
100        CONTINUE
         RETURN
         END
      SUBROUTINE II1(
     + CONV1   ,ENGINE ,IAPG    ,ICONV1 ,IENGIN ,IPOWER ,MAXIPR ,MAXL
     + ,NHVALS ,NSVALS ,PFA     ,POWER   ,QMAX    ,RPM     ,SPEED   ,TORQUE
     + ,TQIND  ,VAA     ,VDA     ,VOCB    ,VOOBS   ,VRIDE   ,VSS )
C
C     -----------------------------
C     UNITS CONVERSION ROUTINE
C     -----------------------------
C
C     1. VARIABLE DECLARATION
         REAL        CONV1   (2,25)
         REAL        ENGINE (2,25)
         INTEGER     IAPG
```

```
      INTEGER   ICONV1
      INTEGER   IENGIN
      INTEGER   IPOWER
      INTEGER   L
      INTEGER   MAXIPR
      INTEGER   MAXL
      INTEGER   RPM
      INTEGER   SPEED
      INTEGER   TORQUE
      INTEGER   N
      INTEGER   NH
      INTEGER   NHVALS
      INTEGER   NR
      INTEGER   NS
      INTEGER   NSVALS
      REAL      POWER  (2,201)
      REAL      QMAX
      REAL      TQINC
      REAL      VAA
      REAL      VDA
      REAL      VOOB   (50)
      REAL      VOOBS  (50)
      REAL      VRIDE  (50,10)
      REAL      VSS
C     3. ALGORITHM
      VSS=VSS*5280.*12./60./60.
      DO 110 NH=1,NHVALS
         VOOB(NH)=VOOB(NH)*5280.*12./60./60.
110   CONTINUE
      DO 120 NS=1,NSVALS
         VOOBS(NS)=VOOBS(NS)*5280.*12./60./60.
120   CONTINUE
      DO 135 L=1,MAXL
         DO 130 NR=1,MAXIPR
            VRIDE(NR,L)=VRIDE(NR,L)*5280.*12./60./60.
130      CONTINUE
135   CONTINUE
      PFA=PFA*144.
      VAA=VAA*3.14159265/180.
      VDA=VDA*3.14159265/180.
      IF( IAPG .EQ. 1 ) GO TO 145
         DO 140 N=1,IPOWER
            POWER(SPEED,N)=POWER(SPEED,N)*(88./60.)*12.
140      CONTINUE
145   CONTINUE
      IF( IAPG .EQ. 2 ) GO TO 199
         DO 150 N=1,IENGIN
            ENGINE(RPM,N)=ENGINE(RPM,N)/60.0
            ENGINE(TORQUE,N)=ENGINE(TORQUE,N)*12.0
150      CONTINUE
         DO 170 N=1,ICONV1
            CONV1(RPM,N)=CONV1(RPM,N)/60.
```

```
170         CONTINUE
            CMAX =QMAX*12.0
            TQIND=TQIND*12.0
199      CONTINUE
         RETURN
         END
      SUBROUTINE II2(
     +  GCW     ,GCWB    ,GCWNB   ,GCWNP   ,GCWP
     + ,IB      ,IP      ,NAMBLY  ,WGHT )
C
      DIMENSION
     +   IB(20)               ,IP(20)                ,WGHT(20)
C  GROSS COMBINED WEIGHT ROUTINE
      GCW=0.0
      GCWP=0.0
      GCWB=0.0
      DO 210 I=1,NAMBLY
         GCW=GCW+WGHT(I)
         GCWB=GCWB+WGHT(I)*FLOAT( IB(I) )
         GCWP=GCWP+WGHT(I)*FLOAT( IP(I) )
210   CONTINUE
      GCWNP=GCW-GCWP
      GCWNB=GCW-GCWB
      RETURN
         END
      SUBROUTINE II3(
     +  ICONST ,NAMBLY ,NVEH    ,NWHL    ,RDIAM
     + ,RIMW    ,SECTW   ,TPSI    ,VTIRE   ,WGHT )
C
      DIMENSION
     +   ICONST(20)           ,NVEH(20)              ,NWHL(20)
     +  ,RDIAM(20)            ,RIMW(20)              ,SECTW(20)
     +  ,TPSI(20,3)           ,VT(20,3)
     +  ,VTIRE(3)             ,WGHT(20)
C  MAXIMUM TIRE SPEED ROUTINE
      DO 325 J=1,3
         DO 320 I=1,NAMBLY
            VT(I,J)=0.
            IF( NVEH(I) .EQ. 0 ) GO TO 315
               S1=( SECTW(I)-0.4*RIMW(I) )/0.75
               HWY=( 4.32/(S1**2.38))
     +            *(((WGHT( I)/FLOAT(NWHL(I)))/(RDIAM(I)*S1))**1.71)
               IF( ICONST(I) .EQ. 1 ) GO TO 310
                  VT(I,J)=(100.*5280.*12./3600.)
     +               *( (TPSI(I,J)/HWY)**2 )
                  GO TO 320
310            CONTINUE
               VT(I,J)=(70.*5280.*12./3600.)
     +            *( (TPSI(I,J)/HWY)**2.25 )
315         CONTINUE
320      CONTINUE
325   CONTINUE
```

```
      DO 340 J=1,3
         VTIRE(J)=VT(1,J)
         DO 330 I=2,NAMBLY
            IF(VT(I,J) .LT. VTIRE(J))
     +         VTIRE(J)=VT(I,J)
  330    CONTINUE
340   CONTINUE
      RETURN
         END
      SUBROUTINE II4(
     +   NAMBLY ,PWTE   ,WT      ,WTE )
C
      DIMENSION
     +   WT( 20)              ,WTE(20)
C  MAXIMUM PATH WIDTH OF COMBINATION'S TRACTION ELEMENTS
      PWTE=WT(1)-WTE(1)
      DO 410 I=2,NAMBLY
         IF((WT(I)-WTE(I)) .GT. PWTE)
     +      PWTE=WT(I)-WTE(I)
410   CONTINUE
      RETURN
         END
      SUBROUTINE II5(
     +   DFLCT  ,DRAT   ,NAMBLY ,NVEH   ,SECTH )
C
      DIMENSION
     +   DFLCT(20,3)          ,DRAT(20,3)          ,NVEH(20)
     +   ,SECTH( 20)
C  TIRE DEFLECTION RATIOS
      DO 520 J=1,3
         DO 510 I=1,NAMBLY
            IF(NVEH(I) .EQ. 0) GO TO 500
               DRAT(I,J)=DFLCT(I,J)/SECTH(I)
500         CONTINUE
510      CONTINUE
520   CONTINUE
      RETURN
         END
      SUBROUTINE II6(
     +   CHARLN ,DFLCT  ,DIAW   ,NAMBLY ,NVEH   ,TRAKLN )
C
      DIMENSION
     +   CHARLN( 20,3)        ,DFLCT(20,3)         ,DIAW(20)
     +   ,NVEH( 20)           ,TRAKLN(20)
C  CHARACTERISTIC LENGHT OF ELEMENTS
      DO 630 J=1,3
         DO 620 I=1,NAMBLY
            IF( NVEH(I) .EQ. 1 ) GO TO 610
C              TRACKED ELEMENT
               CHARLN(I,J)=TRAKLN(I)
               GO TO 620
610         CONTINUE
```

```
C            WHEELED ELEMENT
             CHARLN(I,J)=2.Ø
     +          *SQRT( (DFLCT(I,J)*DIAW(I))
     +          -(DFLCT(I,J)*DFLCT(I,J)) )
62Ø      CONTINUE
63Ø    CONTINUE
       RETURN
          END
       SUBROUTINE II7(
     +   CHARLN ,GCA     ,NAMBLY ,NVEH    ,SECTW   ,TRAKWD )
       DIMENSION
     +    CHARLN(2Ø,3)        ,GCA(2Ø,3)             ,NVEH(2Ø)
     +   ,SECTW(2Ø)           ,TRAKWD(2Ø)
C  GROUND CONTACT AREA OF ELEMENTS
       DO 73Ø J=1,3
          DO 72Ø I=1,NAMBLY
             IF(NVEH(I) .EQ. 1) GO TO 71Ø
C               TRACKED ELEMENT
                GCA(I,J)=CHARLN(I,J)*TRAKWD(I)*2.
71Ø          CONTINUE
C            WHEELED ELEMENT
             GCA(I,J)=CHARLN(I,J)*SECTW(I)
72Ø       CONTINUE
73Ø    CONTINUE
       RETURN
          END
       SUBROUTINE II8(
     +   CGLAT   ,ID      ,NAMBLY ,NVEH    ,SECTW   ,WT      ,WTMAX )
C
       DIMENSION
     +    ID(2Ø)              ,NVEH(2Ø)              ,SECTW(2Ø)
     +   ,WT(2Ø)
C  CONTROLLING LATERAL DISTANCE TO C.G.
       WTMAX=5ØØ.
       DO 83Ø I=1,NAMBLY
          IF(NVEH(I) .EQ. Ø) GO TO 81Ø
C            WHEELED ELEMENT
             TEMP=( WT(I)/2. )-CGLAT
     +          +( SECTW(I)/2. )*FLOAT( ID(I) )
             GO TO 82Ø
81Ø       CONTINUE
C         TRACKED ELEMENT
          TEMP=( WT(I)/2. )-CGLAT
82Ø       CONTINUE
          IF(TEMP .GE. WTMAX) GO TO 825
             WTMAX=TEMP
825       CONTINUE
83Ø    CONTINUE
       RETURN
          END
       SUBROUTINE II9(
     +   IP      ,NAMBLY ,NVEH    ,REVM    ,RR )
```

```
C
      DIMENSION
     +    IP(20)                ,NVEH(20)                ,REVM(20)
C
C  -----------------------------------------------------
C  ROLLING RADIUS OF LARGEST POWERED TIRE ELEMENT
C  -----------------------------------------------------
C
      RR=0.0
      DO 910 I=1,NAMBLY
         IF( IP(I) .EQ. 0 ) GO TO 910
            RX=( 5280.0*12.0 )/( 2.0*3.14159265*REVM(I) )
            IF( RR .LT. RX ) RR=RX
910      CONTINUE
      RETURN
         END
      SUBROUTINE II10(
     +  IB       ,NAMBLY ,WGHT    ,XBRCOF ,XBR )
C
      DIMENSION
     +    IB(20)                ,WGHT(20)
C
C  ------------------------------------------------------------
C  MAXIMUM BRAKING FORCE DEVELOPED BY BRAKED ASSEMBLIES
C  ------------------------------------------------------------
C
      XBR=0.0
      DO 1010 I=1,NAMBLY
         XBR=XBR+XBRCOF*WGHT(I)*FLOAT( IB(I) )
1010  CONTINUE
      RETURN
         END
      SUBROUTINE II11 (
     +  GCWP     ,HPNET   ,HPT )
C
C
C  --------------
C  HORSEPOWER/TON
C  --------------
C
      HPT=HPNET/(GCWP/2000.)
      RETURN
         END
      SUBROUTINE II12(
     +  ASHOE    ,CLRMIN ,CPFFG  ,DFLCT  ,DIAW   ,GROUSH ,HPT     ,IB
     + ,IP       ,ITVAR  ,NAMBLY ,NBOGIE ,NCHAIN ,NVEH    ,NWHL    ,SECTH
     + ,SECTW    ,TRAKLN ,TRAKWD ,VCIFG  ,WGHT )
C
      DIMENSION
     +    ASHOE(20)             ,CLRMIN(20)              ,CPFFG(20,3)
     +  ,DFLCT(20,3)            ,DIAW(20)                ,GROUSH(20)
     +  ,IB(20)                 ,IP(20)                  ,NBOGIE(20)
```

```
     +   ,NCHAIN(20)          ,NVEH(20)            ,NWHL(20)
     +   ,SECTH(20)           ,SECTW(20)           ,TDF(3)
     +   ,TRAKLN(20)          ,TRAKWD(20)          ,VCIFG(20,3)
     +   ,WGHT(20)
C
C  -----------------------------------
C  VEHICLE CONE INDEX IN FINE GRAINED SOIL
C  -----------------------------------
C
      DO 1556 I=1,NAMBLY
         IF( IP(I) .EQ. 0  .AND.  IB(I) .EQ. 0 ) GO TO 1555
            IF( NVEH(I) .EQ. 0 ) GO TO 1534
C              -----------------------
C              WHEELED ASSEMBLY ROUTINE
C              -----------------------
C              CONTACT PRESSURE FACTOR
                 CPFFG(I,1)=WGHT(I)
     +           /( SECTW(I)*FLOAT(NWHL(I))*DIAW(I)/2. )
                 CPFFG(I,2)=CPFFG(I,1)
                 CPFFG(I,3)=CPFFG(I,1)
C              WEIGHT FACTOR
                 IF( WGHT(I) .GE. 2000.0 ) GO TO 1510
                    WF=0.553*WGHT(I)/1000.
                    GO TO 1516
1510             CONTINUE
                 IF( WGHT(I) .GE. 13500. ) GO TO 1512
                    WF=0.033*WGHT(I)/1000.0+1.0
                    GO TO 1516
1512             CONTINUE
                 IF( WGHT(I) .GE. 20000.0 ) GO TO 1514
                    WF=0.142*WGHT(I)/1000.0-0.42
                    GO TO 1516
1514             CONTINUE
                 WF=0.278*WGHT(I)/1000.0-3.115
1516           CONTINUE
C              TIRE FACTOR
                 TF=(10.0+SECTW(I))/100.0
C              GROUSER FACTOR
                 IF( NCHAIN(I) .EQ. 0 ) GO TO 1518
                    GF=1.05
                    GO TO 1520
1518             CONTINUE
                 GF=1.0
1520           CONTINUE
C              WHEEL LOAD FACTOR
                 WLORF=WGHT(I)/1000.0/FLOAT(NWHL(I))/2.
C              CLEARANCE FACTOR
                 CLF=CLRMIN(I)/10.0
C              ENGINE FACTOR
                 IF( HPT .LT. 10.0 ) GO TO 1522
                    EF=1.0
                    GO TO 1524
```

```
1522          CONTINUE
              EF=1.05
1524        CONTINUE
C           TRANSMISSION FACTOR
              IF( ITVAR .EQ. 0 ) GO TO 1526
                TFX=1.05
                GO TO 1528
1526          CONTINUE
              TFX=1.0
1528        CONTINUE
C           TIRE DEFLECTION FACTOR
              DO 1530 J=1,3
                TDF(J)=((1.0-DFLCT(I,J)/SECTH(I))/0.85)**1.5
1530          CONTINUE
C           MOBILITY INDEX
              XMI=(CPFFG(I,1)*WF/TF/GF*WLORF-CLF)*EF*TFX
C           VEHICLE CONE INDEX
              DO 1532 J=1,3
                VCIFG(I,J)=(11.48+0.2*XMI-39.2/(XMI+3.74))
     +            *TDF(J)
1532          CONTINUE
            GO TO 1556
1534      CONTINUE
C         -------------------------
C         TRACKED ASSEMBLY ROUTINE
C         -------------------------
C         CONTACT PRESSURE FACTOR
            CPFFG(I,1)=WGHT(I)/(2.*TRAKLN(I)*TRAKWD(I))
            CPFFG(I,2)=CPFFG(I,1)
            CPFFG(I,3)=CPFFG(I,1)
C         WEIGHT FACTOR
            IF( WGHT(I) .GE. 50000.0 ) GO TO 1536
              WF=1.0
              GO TO 1542
1536        CONTINUE
            IF( WGHT(I) .GE. 70000.0 ) GO TO 1538
              WF=1.2
              GO TO 1542
1538        CONTINUE
            IF( WGHT(I) .GE. 100000.0 ) GO TO 1540
              WF=1.4
              GO TO 1542
1540        CONTINUE
            WF=1.8
1542      CONTINUE
C         TRACK FACTOR
            TF=TRAKWD(I)/100.0
C         GROUSER FACTOR
            IF( GROUSH(I) .LT. 1.5 ) GO TO 1544
              GF=1.1
              GO TO 1546
1544        CONTINUE
```

```
               GF=1.0
1546        CONTINUE
C           BOGIE LOAD RANGE FACTOR
               WLORF=WGHT(I)*10.0/FLOAT(NBOGIE(I))/ASHOE(I)
C           CLEARANCE FACTOR
               CLF=CLRMIN(I)/10.0
C           ENGINE FACTOR
               IF( HPT .LT. 10.0 ) GO TO 1548
                  EF=1.0
                  GO TO 1550
1548           CONTINUE
               EF=1.05
1550        CONTINUE
C           TRANSMISSION FACTOR
               IF( ITVAR .EQ. 0 ) GO TO 1552
                  TFX=1.05
                  GO TO 1554
1552           CONTINUE
               TFX=1.0
1554        CONTINUE
C           MOBILITY INDEX
               XMI=(CPFFG(I,1)*WF/TF/GF+WLORF-CLF)*EF*TFX
C           VEHICLE CONE INDEX
               VCIFG(I,1)=7.0+0.2*XMI-39.2/(XMI+5.6)
               VCIFG(I,2)=VCIFG(I,1)
               VCIFG(I,3)=VCIFG(I,1)
1555      CONTINUE
1556   CONTINUE
       RETURN
          END
       SUBROUTINE II13(
     +   CPFCG   ,IB      ,IP      ,NAMBLY  ,NVEH    ,NWHL
     +  ,RDIAM   ,SECTW   ,TPLY    ,TPSI    ,VCICG   ,WGHT )
       DIMENSION
     +    CPFCG(20,3)         ,IB(20)              ,IP(20)
     +   ,NVEH(20)            ,NWHL(20)            ,RDIAM(20)
     +   ,SECTW(20)           ,TPLY(20)            ,TPSI(20,3)
     +   ,VCICG(20,3)         ,WGHT(20)
C
C  ---------------------------------------------
C  VEHICLE CONE INDEX IN COARSE GRAINED SOIL
C  ---------------------------------------------
C
       DO 1670 I=1,NAMBLY
          IF( IP(I) .EQ. 0  .AND. IB(I) .EQ. 0 ) GO TO 1660
             IF( NVEH(I) .EQ. 0 ) GO TO 1640
C                -------------------------
C                WHEELED ASSEMBLY ROUTINE
C                -------------------------
C                WHEEL DIAMETER FACTOR
                    IF( SECTW(I)/RDIAM(I) .LT. 2.4 ) GO TO 1610
                       WDF=2.0
```

```
                         GO TO 1620
1610                  CONTINUE
                      WDF=5.0
1620               CONTINUE
                   DO 1630 J=1,3
C                     CONTACT PRESSURE FACTOR
                         CPFCG( I,J)=0.607*TPSI(I,J)+
     +                      1.35*(117.0*TPLY(I)/
     +                      (WDF*SECTW(I)+RDIAW( I)))-4.93
C                     CONTACT AREA FACTOR
                         CAF=ALOG10(WGHT(I)/CPFCG(I,J))
C                     STRENGH FACTOR
                         STF=0.0526*FLOAT(NWHL(I))+0.0211*TPSI(I,J)
     +                      -0.35*CAF+1.587
C                     VEHICLE CONE INDEX
                         VCICG( I,J)=10.0**STF
1630               CONTINUE
                   GO TO 1660
1640            CONTINUE
C               ---------------------------
C               TRACKED ASSEMBLY FACTOR
C               ---------------------------
                CO 1650 J=1,3
                   VCICG(I,J)=0.0
1650            CONTINUE
1660         CONTINUE
1670      CONTINUE
             RETURN
             END
          SUBROUTINE LI14 (
     +    DIAW    ,IB      ,IP      ,NAMBLY ,NVEH
     +  ,NWHL    ,SECTW   ,TRAKLN ,TRAKWD ,VCIMUK ,WGHT )
          DIMENSION
     +    DIAW(20)               ,IB(20)              ,IP(20)
     +   ,NVEH(20)               ,NWHL(20)            ,SECTW(20)
     +   ,TRAKLN(20)             ,TRAKWD(20)          ,WGHT(20)
     +   ,VCIMUK(20)
C
C    ---------------------------
C    VEHICLE CONE INDEX IN MUSKEG
C    ---------------------------
C
          CO 1730 I=1,NAMBLY
             IF( IP(I) .EQ. 0  .AND.  IB(I) .EQ. 0 ) GO TO 1720
                IF( NVEH(I) .EQ. 0 ) GC TO 1710
                   VCIMUK(I)=13.0+0.535*WGHT(I)
     +                /((SECTW( I)+DIAW(I))*FLOAT(NWHL(I)))
                GO TO 1730
1710            CONTINUE
                VCIMUK(I)=13.0+0.0625*WGHT( I)
     +             /(TRAKWD( I)+TRAKLN(I))
1720         CONTINUE
```

```
1730  CONTINUE
      RETURN
         END
      SUBROUTINE II15 (
     +  CPFCCG ,CPFCFG ,CPFCG  ,CPFFG  ,IB
     + ,IP     ,NAMBLY ,NVEH   ,NVEHC )
C
      DIMENSION
     +    CPFCCG(3)            ,CPFCG(20,3)          ,CPFCFG(3)
     +  ,CPFFG(20,3)           ,IB(20)               ,IP(20)
     +  ,NVEH(20)
C
C  ----------------------------------------------
C  COMBINED CONTACT PRESSURE FACTOR ROUTINE
C  ----------------------------------------------
C
      N1=0
      N2=0
      DO 1800 I=1,NAMBLY
         IF( NVEH(I) .EQ. 0 ) N1=-1
         IF( NVEH(I) .EQ. 1 ) N2=1
1800  CONTINUE
      NVEHC=N1+N2
      IF( NVEHC .NE. 1 ) GO TO 1840
         DO 1830 J=1,3
            CPFCFG(J)=0.0
            CPFCCG(J)=0.0
            DO 1820 I=1,NAMBLY
               IF( IB(I) .EQ. 0  .AND.  IP(I) .EQ. 0 ) GO TO 1820
                  IF( CPFFG(I,J) .GT. CPFCFG(J) ) CPFCFG(J)=CPFFG(I,J)
                  IF( CPFCG(I,J) .GT. CPFCCG(J) ) CPFCCG(J)=CPFCG(I,J)
1810           CONTINUE
1820        CONTINUE
1830     CONTINUE
         GO TO 1890
1840  CONTINUE
      DO 1880 J=1,3
         CPFCFG(J)=0.0
         CPFCCG(J)=0.0
         DO 1870 I=1,NAMBLY
            IF( NVEH(I) .NE. 0 ) GO TO 1860
               IF( IB(I) .EQ. 0  .AND.  IP(I) .EQ. 0 ) GO TO 1850
                  IF( CPFFG(I,J) .GT. CPFCFG(J) ) CPFCFG(J)=CPFFG(I,J)
                  IF( CPFCG(I,J) .GT. CPFCCG(J) ) CPFCCG(J)=CPFCG(I,J)
1850           CONTINUE
1860        CONTINUE
1870     CONTINUE
1880  CONTINUE
1890  CONTINUE
      RETURN
         END
      SUBROUTINE II16 (
```

```
     +  ATF     ,BTF     ,CONV1   ,CONV2   ,CTF     ,EFF     ,ENGINE ,FD
     + ,FORCE   ,GR      ,IAPG    ,ICONV1  ,ICONV2  ,IENGIN  ,IPOWER ,ITCASE
     + ,ITRAN   ,KII16   ,LOCKUP  ,LUN1    ,MAPG
     + ,NGR     ,PE      ,POWER   ,RR      ,RPM     ,SPEED   ,SR     ,TCASE
     + ,TORQUE  ,TQIND   ,TR      ,TRACTF  ,TRANS   ,VGV     ,TOPSPD ,NPTS
     + ,IERROR )
C
C     ---------------------------
C     POWER TRAIN SCENARIO LOGIC
C     ---------------------------
C
C     1. VARIABLE DECLARATION
         REAL         ATF      (20)
         REAL         BTF      (20)
         REAL         CONV1    (2,25)
         REAL         CONV2    (2,25)
         REAL         CTF      (20)
         REAL         ENGINE   (2,25)
         REAL         FD       (2)
         INTEGER      IAPG
         INTEGER      ICONV1
         INTEGER      ICONV2
         INTEGER      IENGIN
         INTEGER      IPOWER
         INTEGER      ITCASE
         INTEGER      ITRAN
         INTEGER      KII16
         INTEGER      LOCKUP
         INTEGER      LUN1
         INTEGER      MAPG
         INTEGER      MD
         INTEGER      EFF
         INTEGER      FORCE
         INTEGER      GR
         INTEGER      RPM
         INTEGER      SPEED
         INTEGER      SR
         INTEGER      TORQUE
         INTEGER      TR
         INTEGER      NGR
         REAL         PE
         REAL         POWER    (2,201)
         REAL         RR
         REAL         TCASE    (2)
         REAL         TQIND
         REAL         TRACTF   (20,5)
         REAL         TRANS    (2,25)
         REAL         VGV      (20,5)
         PE=2.0
C     3. ALGORITHM
         IF(MAPG .NE. 1) GO TO 200
            IF(IAPG .EQ. 2) GO TO 100
```

```
            CALL TRAIN (
     +          CONV1      ,CONV2      ,ENGINE    ,FD        ,ICONV1
     +         ,ICONV2     ,IENGIN     ,ITCASE    ,ITRAN     ,LOCKUP
     +         ,EFF        ,FORCE      ,GR        ,RPM       ,SPEED
     +         ,SR         ,TORQUE     ,TR        ,NGR       ,PE
     +         ,RR         ,TCASE      ,TQIND     ,TRANS     ,IPOWER
     +         ,POWER      ,LUN1       ,KII16)
            CALL FIT   (
     +          IPOWER     ,FORCE      ,SPEED
     +         ,POWER      ,ATF        ,BTF       ,CTF
     +         ,NGR        ,TRACTF     ,VGV       ,IERROR    ,LUN1     ,KII16)
            GO TO 400
100       CONTINUE
          MAPG=3
          CALL FIT (
     +       IPOWER    ,FORCE     ,SPEED
     +           ,POWER      ,ATF        ,BTF       ,CTF
     +      ,NGR         ,TRACTF    ,VGV        ,IERROR    ,LUN1     ,KII16)
          GO TC 400
200     CONTINUE
        IF( IAPG .EQ. 1) GC TO 300
          CALL FIT (
     +       IPOWER    ,FCRCE     ,SPEED
     +           ,POWER      ,ATF        ,BTF       ,CTF
     +    ,NGR         ,TRACTF    ,VGV        ,IERROR    ,LUN1     ,KII16)
          GO TC 400
300     CONTINUE
        MAPG=4
        CALL TRAIN (
     +      CONV1     ,CONV2      ,ENGINE    ,FD        ,ICONV1
     +    ,ICONV2     ,IENGIN     ,ITCASE    ,ITRAN     ,LOCKUP
     +    ,EFF        ,FORCE      ,GR        ,RPM       ,SPEED
     +    ,SR         ,TORQUE     ,TR        ,NGR       ,PE
     +    ,RR         ,TCASE      ,TQIND     ,TRANS     ,IPOWER
     +    ,POWER      ,LUN1       ,KII16)
        CALL FIT (
     +       IPOWER    ,FORCE     ,SPEED
     +           ,POWER      ,ATF        ,BTF       ,CTF
     +    ,NGR         ,TRACTF    ,VGV        ,IERROR    ,LUN1     ,KII16)
400     TOPSPD = POWER(SPEED,IPOWER)*(60./88.)/12.
        NPTS = INT((TOPSPD+.5)*4.)
        CONTINUE
        RETURN
        END
      SUBROUTINE AUTOM (
     +    ENGINE    ,IENGIN     ,CONV1      ,ICONV1    ,CONV2
     +   ,ICONV2    ,TQIND      ,TRANS      ,NGR       ,FD
     +   ,RR        ,PE         ,POWER      ,IPOWER    ,RPM
     +   ,TORCUE    ,SR         ,TR         ,GR        ,EFF
     +   ,SPEED     ,FORCE      ,LUN1       ,KII16)
C
C     ---------------------------------------------
```

```
C      AUTOMATIC TRANSMISSION WITH TORQUE CONVERTER
C      -------------------------------------------
C
C      1. VARIABLE DECLARATION
          REAL        CONV1  (2,25)
          REAL        CONV2  (2,25)
          REAL        ENGINE (2,25)
          REAL        ESMAX
          REAL        ESMIN
          REAL        FD     (2)
          INTEGER     ICONV1
          INTEGER     ICONV2
          INTEGER     IENGIN
          INTEGER     IPOWER
          INTEGER     EFF
          INTEGER     FORCE
          INTEGER     GR
          INTEGER     RPM
          INTEGER     SPEED
          INTEGER     SR
          INTEGER     TORQUE
          INTEGER     TR
          INTEGER     N
          INTEGER     NG
          INTEGER     NGR
          REAL        PE
          REAL        POWER  (2,201)
          REAL        P1
          REAL        RPMIN
          REAL        RPMOUT
          REAL        RR
          REAL        SPDINC
          REAL        SRATIO
          REAL        TF
          REAL        TORQEN
          REAL        TORQIN
          REAL        TQINC
          REAL        TRANS  (2,25)
          REAL        TRATIO
C      2. ALGORITHM
          DO 160 NG=1,NGR
             DO 150 N=1,201
                ESMIN=ENGINE(RPM,1)
                ESMAX=ENGINE(RPM,IENGIN)
                RPMOUT=( POWER(SPEED,N)/2.0/3.14159265/RR)
     +             *FD(GR)*TRANS(GR,NG)
110             CONTINUE
                   RPMIN=(ESMIN+ESMAX)/2.0
                   SRATIO=RPMOUT/RPMIN
                   IF(SRATIO .LE. CONV1(SR,ICONV1)) GO TO 120
                      SRATIO=CONV1(SR,ICONV1)
120                CONTINUE
```

```
                    CALL LINEAR (
     +              CONV1     ,ICONV1    ,SR        ,RPM       ,SRATIO
     +             ,SPDIND )
                    TORQIN=TQIND*(RPMIN/SPDIND)**2.0
                    CALL LINEAR (
     +              ENGINE    ,IENGIN    ,RPM       ,TORQUE    ,RPMIN
     +             ,TORQEN )
                    IF( (ESMAX-ESMIN) .LE. 1.0/60.0 ) GO TO 130
C                      EXIT LOOP
                    IF(TORQEN .EQ. TORQIN) GO TO 130
C                      EXIT LOOP
                    IF(TORQEN .LT. TORQIN) ESMAX=RPMIN
                    IF(TORQEN .GT. TORQIN) ESMIN=RPMIN
                    GO TO 110
130              CONTINUE
                 P1=PE*TORCIN/100.0
                 IF(   ABS(TORCIN-TORCEN)  .GT.  ABS(P1)   ) GO TO 140
                    CALL LINEAR (
     +              CONV2     ,ICONV2    ,SR        ,TR        ,SRATIO
     +             ,TRATIO )
                    TF=TORCIN*TRATIO*TRANS(GR,NG)
     +              *TRANS(EFF,NG)*FD(GR)*FD(EFF)/RR
                    IF(POWER(FORCE,N) .LT. TF) POWER(FORCE,N)=TF
                    IF(IPOWER .LT. N) IPOWER=N
140              CONTINUE
150           CONTINUE
160        CONTINUE
C       3.DIAGNOSTIC OUTPUT
        IF( KII16 .NE. 1 ) GOTO 300
        WRITE(LUN1,190)
 190    FORMAT(1H1,6H$AUTOM,/)
        WRITE(LUN1,200)
200     FORMAT(1H0,8HCONV1    =)
        WRITE(LUN1,210) ((CONV1(I,J),I=RPM,SR),J=1,ICONV1)
210     FORMAT(10X,E14.8,2X,E14.8,2X,E14.8,2X,E14.8,2X,E14.8,2X,E14.8)
        WRITE(LUN1,220)
220     FORMAT(1H0,8HCONV2    =)
        WRITE(LUN1,210) ((CONV2(I,J),I=TR,SR),J=1,ICONV2)
        WRITE(LUN1,230) EFF
230     FORMAT(1H0,8HEFF      =,I4)
        WRITE(LUN1,240)
240     FORMAT(1H0,8HENGINE   =)
        WRITE(LUN1,210) ((ENGINE(I,J),I=RPM,TORQUE),J=1,IENGIN)
        WRITE(LUN1,245) FD(EFF),FD(GR)
245     FORMAT(1H0,8HFD       =,E14.8,2X,E14.8)
        WRITE(LUN1,250) FORCE,GR,ICONV1,ICONV2,IENGIN,IPOWER,NGR
250     FORMAT(1H0,8HFORCE    =,I4,/,
     +          1H0,8HGR       =,I4,/,1H0,8HICONV1   =,I4,/,
     +          1H0,8HICONV2   =,I4,/,1H0,8HIENGIN   =,I4,/,
     +          1H0,8HIPOWER   =,I4,/,1H0,8HNGR      =,I4)
        WRITE(LUN1,260)
260     FORMAT(1H0,8HPOWER    =)
```

```
      WRITE(LUN1,210) ((POWER(I,J),I=SPEED,FORCE),J=1,IPOWER)
      WRITE(LUN1,270) RR
270   FORMAT(1H0,8HRR      =,E14.8)
      WRITE(LUN1,280) RPM,SPEED,SR,TORQUE,TR
280   FORMAT(1H0,8HRPM     =,I4,/,1H0,8HSPEED   =,I4,/,
     +        1H0,8HSR      =,I4,/,1H0,8HTORQUE =,I4,/,
     +        1H0,8HTR      =,I4)
      WRITE(LUN1,290)
290   FORMAT(1H0,8HTRANS   =)
      WRITE(LUN1,210) ((TRANS(I,J),I=GR,EFF),J=1,NGR)
      WRITE(LUN1,310) TQINC
310   FORMAT(1H0,8HTQINC   =,E14.8)
 300  CONTINUE
         RETURN
         END
      SUBROUTINE FIT(
     +  IPOWER ,FORCE  ,SPEED
     + ,POWER  ,ATF    ,BTF    ,CTF    ,NGR    ,TRACTF
     + ,VGV    ,IERROR ,LUN1   ,KII16)
C
C     --------------------
C     VARIABLE DECLARATIONS
C     --------------------
C
      DIMENSION A      (3,4)
      DIMENSION ATF    (20)
      DIMENSION BTF    (20)
      DIMENSION CTF    (20)
      DIMENSION POWER  (2,20)
      DIMENSION TRACTF(20,5)
      DIMENSION VGV    (20,5)
      DIMENSION X      (3)
      INTEGER   BEGIN
      INTEGER   FORCE
      INTEGER   END
      INTEGER   RIGHT
      INTEGER   SPEED
      LOGICAL   MEMBER
      REAL      LOWER
      REAL      MEDIAN
C
C
C     -------------------------
C     INITIALIZE PROGRAM INDICES
C     -------------------------
C
      PCT = 2.0
      NGR=1
      IERROR=0
      FMAX=0.
      VMAX=0.
      DO 2000 I=1,IPOWER
```

```
         IF( POWER(FORCE,I) .GT. FMAX) FMAX=POWER(FORCE,I)
         IF( POWER(SPEED,I) .GT. VMAX) VMAX=POWER(SPEED,I)
2ØØØ  CONTINUE
C
C     ----------------------------------------------------------
C     BEGIN BY FITTING A QUADRATIC TO THE FIRST THREE POINTS
C     ----------------------------------------------------------
C
      DO 21Ø5 N=1,IPOWER
         IF( POWER(FORCE,N) .EQ. POWER(FORCE,N+1)) GO TO 21ØØ
            LEFT=N
            RIGHT=LEFT+2
            NEXT=RIGHT+1
            BEGIN=LEFT
            END=RIGHT
            GO TO 211Ø
21ØØ     ATF(NGR)=POWER(FORCE,N+1)
         BTF(NGR)=Ø.
         CTF(NGR)=Ø.
         LEFT=1
         RIGHT=N+1
         IF( POWER(FORCE,N+1) .NE. POWER(FORCE,N+2)) GO TO 2172
21Ø5  CONTINUE
211Ø  CONTINUE
         DO 2121 IR=1,3
            DO 212Ø IC=1,4
               A(IR,IC)=Ø.
212Ø        CONTINUE
2121     CONTINUE
2122     CONTINUE
C
C     ------------------------------------------------------------
C     BUILD THE "A" MATRIX FOR THE LEAST SQUARES FIT PROCEDURE
C     ------------------------------------------------------------
C
            DO 214Ø N=BEGIN,END
               IF(N-1 .LT. 1) GO TO 213Ø
C
C     ----------------------------------------------------------------
C     CHECK WHETHER TWO ADJACENT POINTS HAVE THE SAME VALUE OF
C     SPEED AS MIGHT OCCUR AT A GEAR SHIFT POINT.
C     ----------------------------------------------------------------
C
                  IF(POWER(SPEED,N) .EQ. POWER(SPEED,N-1)
     +               .AND. LEFT .LE. N-3) GO TO 2168
                     IF( POWER(SPEED,N) .EQ. POWER(SPEED,N-1)
     +                  .AND. LEFT .EQ. N-2) GO TO 2165
                        IF( POWER(SPEED,N) .EQ. POWER(SPEED,N-1)
     +                     .AND. LEFT .EQ. N-1) GO TO 2174
213Ø           A(3,4)=A(3,4)+POWER(FORCE,N)
               A(2,4)=A(2,4)+POWER(FORCE,N)*(POWER(SPEED,N)/17.6)
               A(1,4)=A(1,4)+POWER(FORCE,N)*(POWER(SPEED,N)/17.6)**2
```

```
                A(3,3)=A(3,3)+1.
                A(2,3)=A(2,3)+(POWER(SPEED,N)/17.6)
                A(1,3)=A(1,3)+(POWER(SPEED,N)/17.6)**2
                A(1,2)=A(1,2)+(POWER(SPEED,N)/17.6)**3
                A(1,1)=A(1,1)+(POWER(SPEED,N)/17.6)**4
2140        CONTINUE
            A(2,1)=A(1,2)
            A(2,2)=A(1,3)
            A(3,1)=A(1,3)
            A(3,2)=A(2,3)
C
C     ------------------------------------------------
C     CALL SUBROUTINE TO INVERT "A" MATRIX AND SOLVE FOR
C     THE COEFFICIENTS TO THE FITTED QUADRATIC
C     ------------------------------------------------
C
            CALL SOLVER (A,X)
            X(2)=X(2)/17.6
            X(1)=X(1)/17.6/17.6
C
C     ------------------------------------------------
C     CHECK NEXT POINT AGAINST THE MAXIMUM NUMBER IN ARRAY
C     ------------------------------------------------
C
         IF(RIGHT+1 .GT. IPOWER) GO TO 2170
            MEDIAN=X(3)+X(2)*POWER(SPEED,NEXT)+X(1)*POWER(SPEED,NEXT)**2
            DIFFER=PCT*MEDIAN/100.0
            LOWER=MEDIAN-DIFFER
            UPPER=MEDIAN+DIFFER
            MEMBER=.TRUE.
            IF(POWER(FORCE,NEXT) .LT. LOWER) MEMBER=.FALSE.
            IF(POWER(FORCE,NEXT) .GT. UPPER) MEMBER=.FALSE.
         IF(.NOT. MEMBER) GO TO 2170
            RIGHT=RIGHT+1
            NEXT=RIGHT+1
            BEGIN=RIGHT
            END=RIGHT
      GO TO 2122
C
C     -----------------
C     STRAIGHT LINE GEAR
C     -----------------
C
2165            X(1)=0.
                RIGHT=RIGHT-1
                X(2)=(POWER(FORCE,RIGHT)-POWER(FORCE,LEFT))/
     +               (POWER(SPEED,RIGHT)-POWER(SPEED,LEFT))
                X(3)=POWER(FORCE,LEFT)-X(2)*POWER(SPEED,LEFT)
                GO TO 2170
2168            RIGHT=RIGHT-1
2170     CONTINUE
         ATF(NGR)=X(3)
```

```
      BTF(NGR)=X(2)
      CTF(NGR)=X(1)
2172  VGV(NGR,1) =POWER(SPEED,LEFT)
      VGV(NGR,5) =POWER(SPEED,RIGHT)
      DO 2176 L=2,4
         VGV(NGR,L)=VGV(NGR,L-1) + (VGV(NGR,5) - VGV(NGR,1))/4.
2176  CONTINUE
      DO 2177 L=1,5
         TRACTF(NGR,L)=ATF(NGR) + BTF(NGR) * VGV(NGR,L) +
     +                    CTF(NGR) * VGV(NGR,L) * VGV(NGR,L)
2177  CONTINUE
      IF(CTF(NGR) .EQ. 0.) GO TO 2173
      CALL APPROX(
     +  POWER  ,FORCE  ,SPEED  ,NGR    ,LEFT
     + ,RIGHT  ,ATF    ,BTF    ,CTF    ,IERROR
     + ,FMAX   ,VMAX   ,LUN1   )
      IF(CTF(NGR) .EQ. 0.) GO TO 2172
      GO TO 2173
C
C    -----------------------------------------------------------------
C     BACKUP TWO POINTS FOR ARTIFICIAL GEAR AT GAP IN TRACTIVE FORCE.
C    -----------------------------------------------------------------
C
2174  RIGHT=RIGHT-2
2173  IF(N-1 .LT. 1) GO TO 2175
      IF(LEFT .LE. N-2) GO TO 2175
C
C    -------------------------------------------------------------
C     IF A GAP IN THE TRACTIVE FORCE DATA OCCURS SUCH AS AT THE
C     SHIFT POINT OF A MANUAL TRANSMISSION, INSERT AN ARTIFICIAL
C     GEAR WHICH IS A VERTICAL LINE HAVING ZERO COEFFICIENTS AND
C     MN, MD, MX SPEED VALUES EQUAL TO THE VALUE AT THE GAP. SET
C     THE MN, MX TRACTIVE FORCE VALUES EQUAL TO THE END POINTS OF
C     THE DISCONTINUITY, AND THE MD TRACTIVE FORCE EQUAL TO THE
C     AVERAGE OF THE MN, MX VALUES. PROCEED TO THE NEXT REAL GEAR.
C    -------------------------------------------------------------
C
        IF(POWER(SPEED,N) .NE. POWER(SPEED,N-1)) GO TO 2175
          RIGHT=RIGHT+1
          ATF(NGR)=0.
          BTF(NGR)=0.
          CTF(NGR)=0.
          DO 2178 L=1,5
             VGV(NGR,L)=POWER(SPEED,RIGHT)
2178      CONTINUE
          TRACTF(NGR,1)=POWER(FORCE,RIGHT-1)
          TRACTF(NGR,5)=POWER(FORCE,RIGHT)
          DO 2179 L=2,4
             TRACTF(NGR,L)=TRACTF(NGR,L-1) -
     +                  (TRACTF(NGR,1) - TRACTF(NGR,5)) / 4.
2179      CONTINUE
2175  IF(RIGHT+2 .GT. IPOWER) GO TO 2180
```

```
         NGR=NGR+1
         LEFT=RIGHT
         RIGHT=LEFT+2
         NEXT=RIGHT+1
         BEGIN=LEFT
         END=RIGHT
      GO TO 2110
2180  CONTINUE
C
C     ------------------------------------------------
C     TEST FINAL POINT. FIT STRAIGHT LINE IF FINAL
C     POINT LIES OUTSIDE OF PREVIOUS GEAR.
C     ------------------------------------------------
C
      IF(RIGHT .GE. IPOWER) GO TO 2190
         NGR=NGR+1
         LEFT=RIGHT
         RIGHT=LEFT+1
         BEGIN=LEFT
         END=RIGHT
         X(1)=0.
         X(2)=(POWER(FORCE,END)-POWER(FORCE,BEGIN))/
     +         (POWER(SPEED,END)-POWER(SPEED,BEGIN))
         X(3)=POWER(FORCE,END)-X(2)*POWER(SPEED,END)
         GO TO 2170
2190  CONTINUE
C     ------------
C     EXIT ROUTINE
C     ------------
      IF( KII16 .NE. 1 ) GOTO 2300
C     -----------------
C     DIAGNOSTIC OUTPUT
C     -----------------
      WRITE(LUN1,2195)
 2195 FORMAT(1H1,4H$FIT,/)
      WRITE(LUN1,2200)
2200  FORMAT(1H0,8HATF     =)
      WRITE(LUN1,2210) (ATF(NG),NG=1,NGR)
2210  FORMAT(10X,E14.8,2X,E14.8,2X,E14.8,2X,E14.8,2X,E14.8,2X,E14.8)
      WRITE(LUN1,2220)
2220  FORMAT(1H0,8HBTF     =)
      WRITE(LUN1,2210) (BTF(NG),NG=1,NGR)
      WRITE(LUN1,2240)
2240  FORMAT(1H0,8HCTF     =)
      WRITE(LUN1,2210) (CTF(NG),NG=1,NGR)
      WRITE(LUN1,2260) FORCE
      WRITE(LUN1,2250) IERROR
2250  FORMAT(1H0,8HIERROR =,I4)
2260  FORMAT(1H0,8HFORCE  =,I4)
      WRITE(LUN1,2270) IPOWER
2270  FORMAT(1H0,8HIPOWER =,I4)
      WRITE(LUN1,2280) NGR
```

```
 2280  FORMAT(1H0,8HNGR     =,I4)
       WRITE(LUN1,2320)
 2320  FORMAT(1H0,8HPOWER   =)
       WRITE(LUN1,2210) ((POWER(L,N),L=1,2),N=1,IPOWER)
       WRITE(LUN1,2340) SPEED
 2340  FORMAT(1H0,8HSPEED   =,I4)
       WRITE(LUN1,2350)
 2350  FORMAT(1H0,8HTRACTF =)
       WRITE(LUN1,2210) ((TRACTF(NG,L),L=1,5),NG=1,NGR)
       WRITE(LUN1,2370)
 2370  FORMAT(1H0,8HVGV     =)
       WRITE(LUN1,2210) ((VGV(NG,L),L=1,5),NG=1,NGR)
 2300  CONTINUE
       RETURN
          END
       SUBROUTINE SOLVER (A,X)
C
C      ---------------------------
C      MATRIX INVERSION SUBROUTINE
C      ---------------------------
C
C      ---------------------
C      VARIABLE DECLARATIONS
C      ---------------------
C
       DIMENSION A(3,4)
       DIMENSION B(3,4)
       DIMENSION X(3)
       INTEGER   COLUMN
       INTEGER   ROW
C
       DO 20 ROW=1,3
          DO 10 COLUMN=1,4
             B(ROW,COLUMN)=A(ROW,COLUMN)
 10       CONTINUE
 20    CONTINUE
       DO 50 KPIVOT=1,2
C
C         ------------------------------------
C         NORMALIZE W.R.T. PIVOTAL ELEMENT
C         ------------------------------------
C
          NPIVOT=5-KPIVOT
          DO 30 KOUNT=1,NPIVOT
             COLUMN=5-KOUNT
             B(KPIVOT,COLUMN)=B(KPIVOT,COLUMN)
     +                         /B(KPIVOT,KPIVOT)
 30       CONTINUE
          KELIM=KPIVOT+1
C
C         ----------------------------------------
C         PERFORM ELIMINATION ON ROWS OF MATRIX A
```

```
C         ----------------------------------------
C
          DO 45 ROW=KELIM,3
             DO 40 COLUMN=KELIM,4
                B(ROW,COLUMN)=B(ROW,COLUMN)
     +                            -(B(ROW,KPIVOT)/B(KPIVOT,KPIVOT))
     +                            *B(KPIVOT,COLUMN)
40           CONTINUE
45        CONTINUE
50     CONTINUE
C
C      ------------------------------------------
C      PERFORM BACK SUBSTITUTION TO OBTAIN
C      COEFFICIENTS  X(1), X(2), X(3)
C      ------------------------------
C
       X(3)=B(3,4)/B(3,3)
       DO 70 KBACK=2,3
          ICOEFF=4-KBACK
          KTERMS=ICOEFF+1
          Q=0.
          DO 60 COLUMN=KTERMS,3
             Q=Q+B(ICOEFF,COLUMN)*X(COLUMN)
60        CONTINUE
          X(ICOEFF)=(B(ICOEFF,4)-Q)/B(ICOEFF,ICOEFF)
70     CONTINUE
       RETURN
       END
C
       SUBROUTINE APPROX
     + (POWER   ,FORCE   ,SPEED   ,NGR     ,NLEFT   ,NRIGHT
     + ,ATF     ,BTF     ,CTF     ,IERROR  ,FMAX
     + ,VMAX    ,LUN1)
C
C      ------------------------------------------------
C      COMPARISON OF A SECOND ORDER POLYNOMIAL CURVE
C      FITTED TO THE POWERTRAIN DATA AND A STRAIGHT
C      LINE FITTED EXACTLY BETWEEN TWO ADJACENT POINTS.
C      ------------------------------------------------
C
C
C      -------------------------
C      1. VARIABLE DECLARATIONS
C      -------------------------
C
          DIMENSION ATF      (20)
          DIMENSION BTF      (20)
          DIMENSION CTF      (20)
          DIMENSION POWER    (2,201)
          REAL      LINE0    (120)
          REAL      LINE1    (120)
          INTEGER FORCE
```

```
          INTEGER SPEED
          REAL    QUAD0
          REAL    QUAD1
          REAL    QUAD2
          REAL    XCOORD   (201)
          REAL    YCOORD   (201)
C
C     ------------
C     2. ALGORITHM
C     ------------
C
          DO 200 N=NLEFT,NRIGHT
             XCOORD(N)=POWER(SPEED,N)
             YCOORD(N)=POWER(FORCE,N)
200       CONTINUE
          QUAD0=ATF(NGR)
          QUAD1=BTF(NGR)
          QUAD2=CTF(NGR)
             CALL LINES(
     +   NLEFT  ,NRIGHT ,XCOORD ,YCOORD ,LINE0  ,LINE1)
             CALL RESIDU(
     +   NLEFT  ,NRIGHT ,XCOORD ,YCOORD ,LINE0  ,LINE1
     +  ,QUAD0  ,QUAD1  ,QUAD2  ,IERROR ,FMAX
     +  ,VMAX   ,LUN1)
          ATF(NGR)=QUAD0
          BTF(NGR)=QUAD1
          CTF(NGR)=QUAD2
       CONTINUE
          RETURN
       END
C
       SUBROUTINE LINES
     + (NLEFT  ,NRIGHT ,XCOORD ,YCOORD ,LINE0  ,LINE1)
C
C     ------------------------------------------------
C     EXACT LINEAR FIT BETWEEN TWO ADJACENT POINTS
C     ------------------------------------------------
C
C
C     ------------------------
C     1. VARIABLE DECLARATIONS
C     ------------------------
C
          REAL    LINE0   (120)
          REAL    LINE1   (120)
          INTEGER SEG
          INTEGER SEGA
          INTEGER SEGB
          REAL    XCOORD (201)
          REAL    YCOORD (201)
C
C     ------------
```

```
C       2. ALGORITHM
C       ------------
C
           SEGA=NLEFT+1
           SEGB=NRIGHT
           DO 200  SEG=SEGA,SEGB
              LINE1(SEG)=(YCOORD(SEG)-YCOORD(SEG-1))
     +                  /(XCOORD(SEG)-XCOORD(SEG-1))
              LINE0(SEG)=YCOORD(SEG)-LINE1(SEG)*XCOORD(SEG)
200        CONTINUE
           RETURN
        END
C
        SUBROUTINE RESIDU
     + (NLEFT  ,NRIGHT ,XCOORD ,YCOORD ,LINE0  ,LINE1
     + ,QUAD0  ,QUAD1  ,QUAD2  ,IERROR ,FMAX
     + ,VMAX   ,LUN1)
C
C       ---------------------------------------------------------------
C       RESIDUAL BETWEEN FIRST AND SECOND ORDER POLYNOMIAL FIT
C       ---------------------------------------------------------------
C
C
C       -------------------------
C       1. VARIABLE DECLARATIONS
C       -------------------------
C
           REAL     ERROR  (120)
           REAL     LENPPA
           REAL     LENPPB
           REAL     LINE0  (120)
           REAL     LINE1  (120)
           INTEGER SEG
           INTEGER Z
           INTEGER SEGA
           INTEGER SEGB
           INTEGER SEG1
           INTEGER SEG2
           REAL     QUAD0
           REAL     QUAD1
           REAL     QUAD2
           REAL     XCOORD (201)
           REAL     YCOORD (201)
C
C       ------------
C       2. ALGORITHM
C       ------------
C
C          ------------
C          A. FUNCTIONS
C          ------------
C
```

```
C             1. FIRST ORDER POLYNOMIAL
                 ALINE(Z,X)=LINE0(Z)+LINE1(Z)*X
C             2. SECOND ORDER POLYNOMIAL
                 QUAD(X)=QUAD0+QUAD1*X+QUAD2*X**2
C             3. RESIDUAL
                 RESID(Z,X)=ABS(QUAD(X)-ALINE(Z,X))
C
C        ------------------------------------------------------------
C        B. DIFFERENCE CALCULATION BETWEEN QUADRATIC CURVE
C           AND THE STRAIGHT LINE FITTED BETWEEN TWO ADJACENT POINTS.
C        ------------------------------------------------------------
C
           PCT1=0.05
           PCT2=0.01
           SEGA=NLEFT+1
           SEGB=NRIGHT
           DO 250 SEG=SEGA,SEGB
C
C        ---------------------------------------------------
C        FIND THE VALUE OF X WHERE THE DIFFERENCE BETWEEN
C        THE QUADRATIC CURVE AND THE STRAIGHT LINE VALUES
C        OF Y ARE A MAXIMUM.
C        ---------------------------------------------------
C
              XPOINT=(LINE1(SEG)-QUAD1)/(2.*QUAD2)
              IF(XPOINT .LT. XCOORD(SEG-1) .OR.
     +           XPOINT .GT. XCOORD(SEG)) GO TO 230
C
C        ---------------------------------------------------
C        CALCULATE THE SLOPE OF THE QUADRATIC AT XPOINT
C        ---------------------------------------------------
C
                 SLOPE=2.*QUAD2*XPOINT + QUAD1
C
C        ------------------------------------------------------------
C        CALCULATE THE MAXIMUM DISTANCE BETWEEN THE CURVE AND
C        THE STRAIGHT LINE IF THE MAXIMUM OCCURS BETWEEN
C        THE END POINTS OF THE LINE.
C        ------------------------------------------------------------
C
                 ERROR(SEG)=RESID(SEG,XPOINT)
                 GO TO 240
C
230           LENPPA=RESID(SEG,XCOORD(SEG-1))
              LENPPB=RESID(SEG,XCOORD(SEG))
C
C        ------------------------------------------------------------
C        CALCULATE DIFFERENCE BETWEEN CURVE AND STRAIGHT LINE
C        IF THE MAXIMUM OCCURS AT THE END POINTS OF THE LINE.
C        ------------------------------------------------------------
C
              ERROR(SEG)=AMAX1(LENPPA,LENPPB)
```

```
                IF(ERROR(SEG) .EQ. LENPPA) XPOINT=XCOORD(SEG-1)
                IF(ERROR(SEG) .EQ. LENPPB) XPOINT=XCOORD(SEG)
                SLOPE=2.*QUAD2*XPOINT + QUAD1
C
C          ----------------------------
C          CALCULATE THE NORMALIZED SLOPE
C          FOR THE ENTIRE POWERTRAIN CURVE
C          ----------------------------
C
240             SLOPEN=ABS(SLOPE*VMAX/FMAX)
C
C          ------------------------------------------------------------
C          CHECK WHETHER CURVE EXTENDS BEYOND MAXIMUM TRACTIVE FORCE
C          IF SO, FIT STRAIGHT LINE BETWEEN POINTS AT WHICH THIS OCCURS.
C          ------------------------------------------------------------
C
                QMAX=AMAX1(QUAD(XPOINT),QUAD(XCOORD(SEG-1)),
     +                     QUAD(XCOORD(SEG)))
                IF(QMAX .GT. FMAX) GO TO 260
C
C          ---------------------------------------------
C          CALCULATE ALLOWABLE ERROR BETWEEN QUADRATIC
C          CURVE AND STRAIGHT LINE WHICH INCLUDES VARIABLE
C          ERROR BASED ON THE NORMALIZED SLOPE.
C          ---------------------------------------------
C
                ALLOW=((QUAD(XCOORD(SEG-1)) + QUAD(XCOORD(SEG)))/2.)*
     +                 PCT1 + SLOPEN*PCT2*QUAD(XPOINT)
             IF(ALLOW .GT. ERROR(SEG)) GO TO 250
                  SEG1=SEG - 1
                  SEG2=SEG - 2
                  WRITE(LUN1,2200) SEG1,SEG,SEG2,SEG1
                  IERROR=1
250          CONTINUE
             GO TO 270
260             QUAD2=0.
                QUAD1=LINE1(SEG)
                QUAD0=LINE0(SEG)
                NRIGHT=SEG
2200              FORMAT(1X,32HCURVE FIT ERROR EXCEEDED BETWEEN,1X,
     +                   12HDATA POINTS:,2I3,/,
     +               8X,38HINSERT DATA POINT BETWEEN DATA POINTS:,2I3)
270         CONTINUE
            RETURN
          END
        SUBROUTINE LINEAR ( ARRAY,IARRAY,INC,MDEP,X,Y )
C
        DIMENSION  ARRAY(2,50)
C
C       -------------------------------
C       LINEAR INTERPOLATION SUBROUTINE
C       -------------------------------
```

```
      IF( ARRAY(IND,1) .LE. ARRAY( IND,IARRAY) ) GO TO 110
C        THE VALUES IN ARRAY(I,J) ARE NOT ASCENDING
         STOP 4
110   CONTINUE
      IF( X .GE. ARRAY(IND,1) ) GO TO 120
         Y=0.0
         GO TO 150
120   CONTINUE
      IF( X .LE. ARRAY(IND,IARRAY) ) GO TO 130
         Y=0.0
         GO TO 150
130   CONTINUE
      NTEMP1=IARRAY-1
      DO 140 N=1,NTEMP1
         IF(  .NOT.  ( (X .GE. ARRAY(IND,N)) .AND.
     +   (X .LE. ARRAY(IND,N+1)) )   ) GO TO 140
            Y=ARRAY(MDEP,N)+( ARRAY(MDEP,N+1)-ARRAY(MDEP,N) )*
     +        ( X-ARRAY(IND,N) )/( ARRAY(IND,N+1)-ARRAY(IND,N) )
            GO TO 150
140   CONTINUE
C        ARRAY CANNOT BE INTERPLOATED
         STOP 5
150   CONTINUE
      RETURN
         END
      SUBROUTINE STICK (
     +   ENGINE   ,IENGIN   ,TRANS    ,NGR      ,FD
     +  ,RR       ,POWER    ,IPOWER   ,RPM      ,TORQUE
     +  ,GR       ,EFF      ,SPEED    ,FORCE    ,LUN1
     +  ,KII16)
C
C     --------------------------
C     MANUAL TRANSMISSION ROUTINE
C     --------------------------
C
C     1. VARIABLE DECLARATION
         REAL       ENGINE (2,25)
         REAL       ESMAX
         REAL       ESMIN
         REAL       FD      (2)
         INTEGER    IENGIN
         INTEGER    IPOWER
         INTEGER    EFF
         INTEGER    FORCE
         INTEGER    GR
         INTEGER    RPM
         INTEGER    SPEED
         INTEGER    TORQUE
         INTEGER    N
         INTEGER    NG
         INTEGER    NGR
         REAL       POWER   (2,201)
```

```
      REAL      RPMEN
      REAL      RR
      REAL      TF
      REAL      TORQEN
      REAL      TRANS   (2,25)
C
C     2. ALGORITHM
      ESMIN=ENGINE(RPM,1)
      ESMAX=ENGINE(RPM,IENGIN)
      DO 120 NG=1,NGR
         DO 110 N=1,201
            RPMEN=( POWER(SPEED,N)/2.0/3.14159265/RR )*
     +            FD(GR)*TRANS(GR,NG)
            IF(RPMEN .LT. ESMIN) GO TO 110
               IF(RPMEN .GT. ESMAX) GO TO 120
C                 EXIT LOOP 110
               CALL LINEAR(ENGINE,IENGIN,RPM,TORQUE,RPMEN,TORQEN)
               TF=TORQEN*TRANS(GR,NG)*TRANS(EFF,NG)
     +            *FD(GR)*FD(EFF)/RR
               IF(POWER(FORCE,N) .LT. TF) POWER(FORCE,N)=TF
               IF(IPOWER .LT. N) IPOWER=N
110      CONTINUE
120   CONTINUE
      DO 300 N=2,IPOWER
         NN=IPOWER - N + 1
         IF(POWER(FORCE,NN) .EQ. 0.)
     +      POWER(FORCE,NN) = POWER(FORCE,NN+1)
300   CONTINUE
C
C     3. DIAGNOSTIC OUTPUT
      IF (KII16 .NE. 1) GOTO 210
      WRITE(LUN1,100)
100   FORMAT(1H1,6H$STICK,/)
      WRITE(LUN1,115) EFF
115   FORMAT(1H0,8HEFF     =,I4)
      WRITE(LUN1,125)
125   FORMAT(1H0,8HENGINE =)
      WRITE(LUN1,130) ((ENGINE(I,J),I=RPM,TORQUE),J=1,IENGIN)
130   FORMAT(10X,6(E14.8,2X))
      WRITE(LUN1,140) (FD(I),I=1,2)
140   FORMAT(1H0,8HFD      =,2(E14.8,2X))
      WRITE(LUN1,150) FORCE,GR,IENGIN,IPOWER,NGR
150   FORMAT(1H0,8HFORCE   =,I4,/,1H0,8HGR      =,I4,/,
     +       1H0,8HIENGIN =,I4,/,1H0,8HIPOWER =,I4,/,
     +       1H0,8HNGR     =,I4)
      WRITE(LUN1,160)
160   FORMAT(1H0,8HPOWER  =)
      WRITE(LUN1,130) ((POWER(I,J),I=SPEED,FORCE),J=1,IPOWER)
      WRITE(LUN1,170) RPM
170   FORMAT(1H0,8HRPM     =,I4)
      WRITE(LUN1,180) RR
180   FORMAT(1H0,8HRR      =,E14.8)
```

```
      WRITE(LUN1,190) SPEED
190   FORMAT(1H0,8HSPEED   =,34)
      WRITE(LUN1,200)
200   FORMAT(1H0,8HTRANS   =)
      WRITE(LUN1,130) ((TRANS(I,J),I=GR,EFF),J=1,NGR)
210   CONTINUE
         RETURN
         END
      SUBROUTINE TRAIN (
     +   CONV1      ,CONV2     ,ENGINE    ,FC        ,ICONV1
     +  ,ICONV2     ,IENGIN    ,ITCASE    ,ITRAN     ,LOCKUP
     +  ,EFF        ,FORCE     ,GR        ,RPM       ,SPEED
     +  ,SR         ,TORQUE    ,TR        ,NGR       ,PE
     +  ,RR         ,TCASE     ,TQIND     ,TRANS     ,IPOWER
     +  ,POWER      ,LUN1      ,KII16)
C
C     ----------------------------------------
C     CONSTRUCTION OF THE VEHICLE
C     TRACTIVE EFFORT .VERSUS. SPEED CURVE
C     FROM POWER TRAIN DATA
C     ----------------------------------------
C
C     1. VARIABLE DECLARATION
         REAL       CONV1   (2,25)
         REAL       CONV2   (2,25)
         REAL       ENGINE  (2,25)
         REAL       FD      (2)
         INTEGER    ICONV1
         INTEGER    ICONV2
         INTEGER    IENGIN
         INTEGER    IPOWER
         INTEGER    ITCASE
         INTEGER    LOCKUP
         INTEGER    EFF
         INTEGER    FORCE
         INTEGER    GR
         INTEGER    RPM
         INTEGER    SPEED
         INTEGER    SR
         INTEGER    TORQUE
         INTEGER    TR
         INTEGER    N
         INTEGER    NGR
         REAL       PE
         REAL       POWER   (2,201)
         REAL       RR
         REAL       TCASE   (2)
         REAL       TQIND
         REAL       TRANS   (2,25)
C
C     2. ALGORITHM
         IPOWER=0
```

```
      DO 2010 N=1,201
         POWER(SPEED,N)=FLOAT(N-1)*0.5*(88./60.)*12.0
         POWER(FORCE,N)=0.0
2010  CONTINUE
C     -----------------------------------------
C     ADJUST TRANSMISSION INPUT FOR ENGINE
C     TO TRANSMISSION TRANSFER CASE
C     ---------------------------------
      IF( ITCASE .EQ. 0 ) GO TO 2021
         DO 2020 N=1,IENGIN
         ENGINE(RPM,N)=ENGINE(RPM,N)/TCASE(GR)
         ENGINE(TORQUE,N)=ENGINE(TORQUE,N)*TCASE(GR)*TCASE(EFF)
2020     CONTINUE
2021  CONTINUE
C     ---------------------------
C     CHOOSE TRANSMISSION TYPE
C     ---------------------------
      IF( ITRAN .EQ. 0 ) GO TO 2040
         CALL AUTOM (
     +      ENGINE    ,IENGIN    ,CONV1     ,ICONV1    ,CONV2
     +     ,ICONV2    ,TQIND     ,TRANS     ,NGR       ,FD
     +     ,RR        ,PE        ,POWER     ,IPOWER    ,RPM
     +     ,TORQUE    ,SR        ,TR        ,GR        ,EFF
     +     ,SPEED     ,FORCE     ,LUN1      ,KII16)
         IF( LOCKUP .EQ. 0 ) GO TO 2050
2040  CONTINUE
            CALL STICK (
     +         ENGINE    ,IENGIN    ,TRANS     ,NGR       ,FD
     +        ,RR        ,POWER     ,IPOWER    ,RPM       ,TORQUE
     +        ,GR        ,EFF       ,SPEED     ,FORCE     ,LUN1
     +        ,KII16)
2050     CONTINUE
C
C     3. DIAGNOSTIC OUTPUT
      IF( KII16 .NE. 1 ) GOTO 300
      WRITE(LUN1,190)
 190  FORMAT(1H1,6H$TRAIN,/)
      WRITE(LUN1,200)
200   FORMAT(1H0,8HCONV1   =)
      WRITE(LUN1,210) ((CONV1(I,J),I=RPM,SR),J=1,ICONV1)
210   FORMAT(10X,E14.8,2X,E14.8,2X,E14.8,2X,E14.8,2X,E14.8,2X,E14.8)
      WRITE(LUN1,220)
220   FORMAT(1H0,8HCONV2   =)
      WRITE(LUN1,210) ((CONV2(I,J),I=TR,SR),J=1,ICONV2)
      WRITE(LUN1,230) EFF
230   FORMAT(1H0,8HEFF     =,I4)
      WRITE(LUN1,240)
240   FORMAT(1H0,8HENGINE =)
      WRITE(LUN1,210) ((ENGINE(I,J),I=RPM,TORQUE),J=1,IENGIN)
      WRITE(LUN1,245) FD(EFF),FD(GR)
245   FORMAT(1H0,8HFD      =,E14.8,2X,E14.8)
      WRITE(LUN1,250) FORCE,GR,ICONV1,ICONV2,IENGIN,ITCASE,ITRAN,
```

```
     +                    IPOWER,LOCKUP,NGR
250   FORMAT(1H0,8HFORCE  =,I4,/,
     +          1H0,8HGR       =,I4,/,1H0,8HICONV1 =,I4,/,
     +          1H0,8HICONV2 =,I4,/,1H0,8HIENGIN =,I4,/,
     +          1H0,8HITCASE =,I4,/,1H0,8HITRAN  =,I4,/,
     +          1H0,8HIPOWER =,I4,/,1H0,8HLOCKUP =,I4,/,
     +          1H0,8HNGR    =,I4)
      WRITE(LUN1,260)
260   FORMAT(1H0,8HPOWER  =)
      WRITE(LUN1,210) ((POWER(I,J),I=SPEED,FORCE),J=1,IPOWER)
      WRITE(LUN1,270) RR
270   FORMAT(1H0,8HRR      =,E14.8)
      WRITE(LUN1,280) RPM,SPEED,SR,TORQUE,TR
280   FORMAT(1H0,8HRPM     =,I4,/,1H0,8HSPEED  =,I4,/,
     +          1H0,8HSR      =,I4,/,1H0,8HTORQUE =,I4,/,
     +          1H0,8HTR      =,I4)
      WRITE(LUN1,285) (TCASE(I),I=GR,EFF)
285   FORMAT(1H0,8HTCASE  =,2(E14.8,2X))
      WRITE(LUN1,290)
290   FORMAT(1H0,8HTRANS  =)
      WRITE(LUN1,210) ((TRANS(I,J),I=GR,EFF),J=1,NGR)
      WRITE(LUN1,310) TQIND
310   FORMAT(1H0,8HTQIND  =,E14.8)
 300  CONTINUE
         RETURN
         END
      SUBROUTINE II17(
     +  CID     ,IDIESL  ,GCW     ,NAMBLY  ,NCYL     ,NENG
     +  ,NGR    ,NVEH    ,QMAX    ,RMX     ,RR       ,TRACTF  )
C
C     ---------------------
C     ROTATING MASS FACTORS
C     ---------------------
C
C     1. VARIABLE DECLARATION
         REAL        IDIESL
         REAL        MF1
         REAL        MF2
         REAL        NCYL
         REAL        NENG
         INTEGER     NVEH    (20)
         REAL        RMX     (20)
         REAL        TRACTF  (20,5)
C
C     2. ALGORITHMS
         MF1 = 1.03
         ETA = 0.9
         DO 1710 I=1,NAMBLY
            IF(NVEH(I) .NE. 0) GO TO 1710
               MF1 = 1.14
               ETA = 0.7
               GO TO 1720
```

```
1710      CONTINUE
1720      MF2 = (.008*(((CIDIESL*CID)**1.68)/NCYL/GCW)*NENG
          DO 1730 NG=1,NGR
             GR        = TRACTF(NG,3)*RR/ETA/QMAX
             RMX(NG)   = MF1 + MF2*GR*GR
1730      CONTINUE
          RETURN
          END
      SUBROUTINE TERTL
C     ------------------
C     TERRAIN TRANSLATORS
C     ------------------
C
C
C     1. LABLED COMMON ASSIGNMENTS
          COMMON /IO/         IEOF
          COMMON /IO/         KBUFF
          COMMON /IO/         LUN1
          COMMON /IO/         LUN2
          COMMON /IO/         LUN3
          COMMON /IO/         LUN4
          COMMON /IO/         LUN5
          COMMON /IO/         LUN6
          COMMON /IO/         LUN7
          COMMON /IO/         LUN8
          COMMON /IO/         LUN9
          COMMON /IO/         LUN10
          INTEGER             MO
          COMMON /INDEX/      MO
          INTEGER             COWN
          COMMON /INDEX/      COWN
          INTEGER             EFF
          COMMON /INDEX/      EFF
          INTEGER             FORCE
          COMMON /INDEX/      FORCE
          INTEGER             GR
          COMMON /INDEX/      GR
          COMMON /INDEX/      LEVEL
          INTEGER             MN
          COMMON /INDEX/      MN
          INTEGER             RPM
          COMMON /INDEX/      RPM
          INTEGER             SPEED
          COMMON /INDEX/      SPEED
          INTEGER             SR
          COMMON /INDEX/      SR
          INTEGER             TR
          COMMON /INDEX/      TR
          INTEGER             TORQUE
          COMMON /INDEX/      TORQUE
          INTEGER             UR
          COMMON /INDEX/      UR
```

```
      INTEGER              MX
      COMMON /INDEX/       MX
      COMMON /TERRAN/      AA
      COMMON /TERRAN/      ACTRMS
      COMMON /TERRAN/      AREA
      COMMON /TERRAN/      AREAC
      COMMON /TERRAN/      C4
      COMMON /TERRAN/      DIST
      COMMON /TERRAN/      EANG
      COMMON /TERRAN/      ECF
      COMMON /TERRAN/      ELEV
      COMMON /TERRAN/      FNU        (3)
      COMMON /TERRAN/      GRADE
      COMMON /TERRAN/      IOBS
      COMMON /TERRAN/      IOST
      COMMON /TERRAN/      IROAD
      COMMON /TERRAN/      IS         (9)
      COMMON /TERRAN/      IST
      COMMON /TERRAN/      IOUT
      COMMON /TERRAN/      NI
      COMMON /TERRAN/      NIU
      COMMON /TERRAN/      CAW
      COMMON /TERRAN/      CBAA
      COMMON /TERRAN/      CBH
      COMMON /TERRAN/      CBL
      COMMON /TERRAN/      CBS
      COMMON /TERRAN/      CBW
      COMMON /TERRAN/      CBMINW
      COMMON /TERRAN/      CDIA
      COMMON /TERRAN/      NWASFO
      COMMON /TERRAN/      RADC
      COMMON /TERRAN/      RCI
      COMMON /TERRAN/      RCIC       (4)
      COMMON /TERRAN/      RCURV      (11)
      COMMON /TERRAN/      RD
      COMMON /TERRAN/      RDA        (12)
      COMMON /TERRAN/      S          (9)
      COMMON /TERRAN/      SE         (9)
      COMMON /TERRAN/      SDL        (9)
      COMMON /TERRAN/      SURFF
      COMMON /TERRAN/      TANPHI
      COMMON /TERRAN/      THETA      (3)
      COMMON /TERRAN/      VCURV      (4,11)
      COMMON /TERRAN/      WA
      COMMON /TERRAN/      WD
      COMMON /SCEN/        COHES
      COMMON /SCEN/        VWALK
      COMMON /SCEN/        COLMAX
      COMMON /SCEN/        GAMMA
      COMMON /SCEN/        IOVER
      COMMON /SCEN/        ISEASN
      COMMON /SCEN/        ISURF
```

```
COMMON /SCEN/      ISNOW
COMMON /SCEN/      KII1
COMMON /SCEN/      KII2
COMMON /SCEN/      KII3
COMMON /SCEN/      KII4
COMMON /SCEN/      KII5
COMMON /SCEN/      KII6
COMMON /SCEN/      KII7
COMMON /SCEN/      KII8
COMMON /SCEN/      KII9
COMMON /SCEN/      KII1Ø
COMMON /SCEN/      KII11
COMMON /SCEN/      KII12
COMMON /SCEN/      KII13
COMMON /SCEN/      KII14
COMMON /SCEN/      KII15
COMMON /SCEN/      KII16
COMMON /SCEN/      KII17
COMMON /SCEN/      KMAP
COMMON /SCEN/      KSCEN
COMMON /SCEN/      KTPP
COMMON /SCEN/      KVEH
COMMON /SCEN/      KIV1
COMMON /SCEN/      KIV2
COMMON /SCEN/      KIV3
COMMON /SCEN/      KIV4
COMMON /SCEN/      KIV5
COMMON /SCEN/      KIV6
COMMON /SCEN/      KIV7
COMMON /SCEN/      KIV8
COMMON /SCEN/      KIV9
COMMON /SCEN/      KIV1Ø
COMMON /SCEN/      KIV11
COMMON /SCEN/      KIV12
COMMON /SCEN/      KIV13
COMMON /SCEN/      KIV14
COMMON /SCEN/      KIV15
COMMON /SCEN/      KIV16
COMMON /SCEN/      KIV17
COMMON /SCEN/      KIV18
COMMON /SCEN/      KIV19
COMMON /SCEN/      KIV2Ø
COMMON /SCEN/      KIV21
COMMON /SCEN/      LAC
INTEGER            DETAIL
COMMON /SCEN/      DETAIL
COMMON /SCEN/      MAP
COMMON /SCEN/      MAPG
COMMON /SCEN/      MONTH
COMMON /SCEN/      MOPP
COMMON /SCEN/      NSLIP
COMMON /SCEN/      NTRAV
```

```
          COMMON /SCEN/      NTUX
          COMMON /SCEN/      FWI
          COMMON /SCEN/      REACT
          COMMON /SCEN/      RDFOG
          INTEGER            SEARCH
          COMMON /SCEN/      SEARCH
          COMMON /SCEN/      SETYPC
          COMMON /SCEN/      VBRAKE
          COMMON /SCEN/      VISMNV
          COMMON /SCEN/      VLIM
          COMMON /SCEN/      ZSNCW
C      2. ALGCRITHM
          IF( MAP .EQ. 74) GO TO 201Ø
              IF(MAP .EQ. 11) GO TO 2Ø15
C
C         CLASS INTERVAL TERRAIN TRANSLATOR
C
                 CALL MAP71(
     +  AA       ,ACTRMS ,AREA   ,ELEV   ,GRADE  ,IEOF   ,IOST   ,IS
     + ,IST      ,ITUT   ,LUN2   ,NI     ,NTU    ,NTUX   ,OBH    ,OBL
     + ,OBS      ,OBW    ,RCIC   ,RD     ,S      ,SD     ,SDL    ,SEARCH
     + ,WD )
C         DIAGNOSTIC OUTPUT LIST
          NAMELIST /XMAP71/
     +  AA       ,ACTRMS ,AREA   ,ELEV   ,GRADE  ,IEOF   ,IOST   ,IS
     + ,IST      ,ITUT   ,LUN2   ,NI     ,NTU    ,NTUX   ,OBH    ,OBL
     + ,OBS      ,OBW    ,RCIC   ,RD     ,S      ,SD     ,SDL    ,SEARCH
     + ,WD
          IF((KMAP.EQ.1) .AND. (IEOF.EQ.Ø)) WRITE(LUN1,XMAP71)
              GO TO 2Ø2Ø
2Ø1Ø      CONTINUE
C
C         NATURAL TERRAIN UNITS TRANSLATCR
C
          CALL MAP74(
     +  AA       ,ACTRMS ,AREA   ,ELEV   ,GRADE  ,IEOF   ,IOST   ,IS
     + ,IST      ,ITUT   ,LUN2   ,MONTH  ,NI     ,NTU    ,NTUX   ,OBH
     + ,OBL      ,OBS    ,CBW    ,RCIC   ,RD     ,RDA    ,RDA1   ,RDA2
     + ,RDA3     ,RDA4   ,S      ,SD     ,SDL    ,SEARCH ,WD )
C         DIAGNOSTIC OUTPUT LIST
          NAMELIST /XMAP74/
     +  AA       ,ACTRMS ,AREA   ,ELEV   ,GRADE  ,IEOF   ,IOST   ,IS
     + ,IST      ,ITUT   ,LUN2   ,MONTH  ,NI     ,NTU    ,NTUX   ,OBH
     + ,OBL      ,OBS    ,CBW    ,RCIC   ,RD     ,RDA    ,RDA1   ,RDA2
     + ,RDA3     ,RDA4   ,S      ,SD     ,SDL    ,SEARCH ,WD
C         DIAGNOSTIC OUTPUT
          IF((KMAP.EQ.1) .AND. (IEOF.EQ.Ø)) WRITE(LUN1,XMAP74)
              GO TO 2Ø2Ø
C
C         ROAC NET TRANSLATOF
C
2Ø15      CALL MPRD74(
```

```
     +  ACTRMS ,CURVV  ,DIST   ,EANG   ,ELEV   ,FMU    ,GRADE
     + ,IEOF   ,IROAD  ,IST    ,ITUT   ,IURB   ,LUN2   ,NTU    ,NTUX
     + ,NVASHO ,MONTH  ,RADC   ,RCURV  ,RD     ,RDA    ,RDFOG  ,RCIC
     + ,SEARCH ,SURFF  ,VCURV )
C        DIAGNOSTIC OUTPUT LIST
         NAMELIST /XROAD/
     +  ACTRMS ,CURVV  ,DIST   ,EANG   ,ELEV   ,FMU    ,GRADE
     + ,IEOF   ,IROAD  ,IST    ,ITUT   ,IURB   ,LUN2   ,NTU    ,NTUX
     + ,NVASHO ,MONTH  ,RADC   ,RCURV  ,RD     ,RDA    ,RDFOG  ,RCIC
     + ,SEARCH ,SURFF  ,VCURV
         IF((KMAP .EQ. 1) .AND. (IEOF .EQ. 0)) WRITE(LUN1,XROAD)
2020     CONTINUE
         IF( IEOF .EQ. 1) GO TO 4000
C
C        TERRAIN PREPROCESSOR
C
         CALL TPP(
     +  AA     ,ACTRMS ,AREA0  ,CI     ,ECF    ,ELEV   ,GAMMA  ,GRADE
     + ,IOBS   ,ISEASN ,ISNOW  ,IST    ,ITUT   ,NI     ,OAW    ,OBAA
     + ,OBH    ,OBL    ,OBMINW ,OBS    ,OBW    ,ODIA   ,PHI    ,RADC
     + ,RCI    ,RCIC   ,RD     ,S      ,TANPHI ,THETA  ,WA     ,ZSNOW )
C        DIAGNOSTIC OUTPUT LIST
         NAMELIST /XTPP/
     +  AA     ,ACTRMS ,AREA0  ,CI     ,ECF    ,ELEV   ,GAMMA  ,GRADE
     + ,IOBS   ,ISEASN ,ISNOW  ,IST    ,ITUT   ,NI     ,OAW    ,OBAA
     + ,OBH    ,OBL    ,OBMINW ,OBS    ,OBW    ,ODIA   ,PHI    ,RADC
     + ,RCI    ,RCIC   ,RD     ,S      ,TANPHI ,THETA  ,WA     ,ZSNOW
C        DIAGNOSTIC OUTPUT
            IF(KTPP .EQ. 1) WRITE(LUN1,XTPP)
C     4. TERMINUS
4000     CONTINUE
         RETURN
         END
      SUBROUTINE AREAL
C     ------------
C     ------------
C
C
C     1. LABLED COMMON ASSIGNMENTS
         COMMON /IO/         IEOF
         COMMON /IO/         KBUFF
         COMMON /IO/         LUN1
         COMMON /IO/         LUN2
         COMMON /IO/         LUN3
         COMMON /IO/         LUN4
         COMMON /IO/         LUN5
         COMMON /IO/         LUN6
         COMMON /IO/         LUN7
         COMMON /IO/         LUN8
         COMMON /IO/         LUN9
         COMMON /IO/         LUN10
         INTEGER             MD
```

```
      COMMON /INDEX/     MO
      INTEGER            COWN
      COMMON /INDEX/     COWN
      INTEGER            EGF
      COMMON /INDEX/     EGF
      INTEGER            FORCE
      COMMON /INDEX/     FORCE
      INTEGER            GR
      COMMON /INDEX/     GR
      COMMON /INDEX/     LEVEL
      INTEGER            MN
      COMMON /INDEX/     MN
      INTEGER            RPM
      COMMON /INDEX/     RPM
      INTEGER            SPEED
      COMMON /INDEX/     SPEED
      INTEGER            SR
      COMMON /INDEX/     SR
      INTEGER            TR
      COMMON /INDEX/     TR
      INTEGER            TORQUE
      COMMON /INDEX/     TORQUE
      INTEGER            UB
      COMMON /INDEX/     UB
      INTEGER            MX
      COMMON /INDEX/     MX
      COMMON /VEHICL/    AGD
      COMMON /VEHICL/    ASHOE      (20)
      COMMON /VEHICL/    AVEC
      COMMON /VEHICL/    AXLSP      (20)
      COMMON /VEHICL/    CB
      COMMON /VEHICL/    CGH
      COMMON /VEHICL/    CGLAT
      COMMON /VEHICL/    CGR
      COMMON /VEHICL/    CID
      COMMON /VEHICL/    CL
      COMMON /VEHICL/    CLRMIN     (20)
      COMMON /VEHICL/    CONV1      (2,25)
      COMMON /VEHICL/    CONV2      (2,25)
      COMMON /VEHICL/    CSLCT      (20,3)
      COMMON /VEHICL/    CLAW       (20)
      COMMON /VEHICL/    CRAFT
      COMMON /VEHICL/    ENGINE     (2,50)
      COMMON /VEHICL/    EXEHGT
      COMMON /VEHICL/    FD         (2)
      COMMON /VEHICL/    FORDD
      COMMON /VEHICL/    GROUSH     (20)
      COMMON /VEHICL/    HVALS      (25)
      COMMON /VEHICL/    IMPG
      COMMON /VEHICL/    IB         (20)
      COMMON /VEHICL/    ID         (20)
      REAL               IDIESL
```

```
      COMMON /VEHICL/   IDIESL
      COMMON /VEHICL/   IENGIN
      COMMON /VEHICL/   IG        (20)
      COMMON /VEHICL/   ICONST    (20)
      COMMON /VEHICL/   ICONV1
      COMMON /VEHICL/   ICONV2
      COMMON /VEHICL/   IPOWER
      COMMON /VEHICL/   IT        (20)
      COMMON /VEHICL/   ITCASE
      COMMON /VEHICL/   ITRAN
      COMMON /VEHICL/   ITVAR
      COMMON /VEHICL/   LOCKUP
      COMMON /VEHICL/   MAXIPR
      COMMON /VEHICL/   MAXL
      COMMON /VEHICL/   NAMBLY
      COMMON /VEHICL/   NBOGIE    (20)
      COMMON /VEHICL/   NCHAIN    (20)
      REAL              NCYL
      COMMON /VEHICL/   NCYL
      REAL              NENG
      COMMON /VEHICL/   NENG
      COMMON /VEHICL/   NPNET
      COMMON /VEHICL/   NFL       (20)
      COMMON /VEHICL/   NGR
      COMMON /VEHICL/   NHVALS
      COMMON /VEHICL/   NRAD      (20)
      COMMON /VEHICL/   NSVALS
      COMMON /VEHICL/   NVEH      (20)
      COMMON /VEHICL/   NWHL      (20)
      COMMON /VEHICL/   NWR
      COMMON /VEHICL/   PBF
      COMMON /VEHICL/   PBHT
      COMMON /VEHICL/   PCA
      COMMON /VEHICL/   POWER     (2,20)
      COMMON /VEHICL/   QMAX
      COMMON /VEHICL/   RDIAM     (20)
      COMMON /VEHICL/   REVM      (20)
      COMMON /VEHICL/   RIMW      (20)
      COMMON /VEHICL/   RNS       (20)
      COMMON /VEHICL/   RW        (20)
      COMMON /VEHICL/   SAE
      COMMON /VEHICL/   SAI
      COMMON /VEHICL/   SECTH     (20)
      COMMON /VEHICL/   SECTW     (20)
      COMMON /VEHICL/   SVALS     (25)
      COMMON /VEHICL/   TCASE     (2)
      COMMON /VEHICL/   TE
      COMMON /VEHICL/   TPLY      (20)
      COMMON /VEHICL/   TPSI      (20,3)
      COMMON /VEHICL/   TQINC
      COMMON /VEHICL/   TRAKLN    (20)
      COMMON /VEHICL/   TRAKWD    (20)
```

```
COMMON /VEHICL/   TRANS    (2,2Ø)
COMMON /VEHICL/   VAA
COMMON /VEHICL/   VBA
COMMON /VEHICL/   VES
COMMON /VEHICL/   VOOB     (25)
COMMON /VEHICL/   VOOBS    (25)
COMMON /VEHICL/   VRIDE    (2Ø,3)
COMMON /VEHICL/   VSS
COMMON /VEHICL/   VSSAXP
COMMON /VEHICL/   WC
COMMON /VEHICL/   WDAXP
COMMON /VEHICL/   WDPTH    (2Ø)
COMMON /VEHICL/   WOTH
COMMON /VEHICL/   WGHT     (2Ø)
COMMON /VEHICL/   WRAT     (2Ø)
COMMON /VEHICL/   WRFORD
COMMON /VEHICL/   WT       (2Ø)
COMMON /VEHICL/   WTE      (2Ø)
COMMON /VEHICL/   WWAXP
COMMON /VEHICL/   XBRCCF
COMMON /VEHICL/   WJ
COMMON /VEHICL/   LOCDIF
COMMON /VEHICL/   SHF
COMMON /PREP/     A        (3,4)
COMMON /PREP/     ATF      (2Ø)
COMMON /PREP/     BTF      (2Ø)
COMMON /PREP/     CHARLN   (2Ø,3)
COMMON /PREP/     CRFCFG   (3)
COMMON /PREP/     CRFCCG   (3)
COMMON /PREP/     CRFCG    (2Ø,3)
COMMON /PREP/     CRFFG    (2Ø,3)
COMMON /PREP/     CTF      (2Ø)
COMMON /PREP/     DRAT     (2Ø,3)
COMMON /PREP/     GCA      (2Ø,3)
COMMON /PREP/     GCW
COMMON /PREP/     GCWB
COMMON /PREP/     GCWNB
COMMON /PREP/     GCWNP
COMMON /PREP/     GCWP
COMMON /PREP/     FRT
COMMON /PREP/     NDF      (38)
COMMON /PREP/     NVEHC
COMMON /PREP/     FWTE
COMMON /PREP/     F        (3)
COMMON /PREP/     FNX      (2Ø)
COMMON /PREP/     FR
COMMON /PREP/     TRACTF   (2Ø,5)
COMMON /PREP/     TRAPSI   (3)
COMMON /PREP/     VCICG    (2Ø,3)
COMMON /PREP/     VCIFG    (2Ø,3)
COMMON /PREP/     VCIMUK   (2Ø)
COMMON /PREP/     VGV      (2Ø,5)
```

```
      COMMON /PREP/      VT        (20,3)
      COMMON /PREP/      VTIRE     (3)
      COMMON /PREP/      WTMAX
      COMMON /PREP/      X         (3)
      COMMON /PREP/      XBR
      COMMON /OBS/       AVALS     (14)
      COMMON /OBS/       CLEAR     (7,14,5)
      COMMON /OBS/       FGG       (7,14,5)
      COMMON /OBS/       FOOMAK    (7,14,5)
      COMMON /OBS/       HOVALS    (7)
      COMMON /OBS/       NANG
      COMMON /OBS/       NOHGT
      COMMON /OBS/       NACTH
      COMMON /OBS/       WVALS     (5)
      COMMON /DERIVE/    ADT       (9)
      COMMON /DERIVE/    NOGCBF
      COMMON /DERIVE/    CAREA
      COMMON /DERIVE/    DOWB
      COMMON /DERIVE/    COWP
      COMMON /DERIVE/    COWPB     (20)
      COMMON /DERIVE/    FA        (20,3)
      COMMON /DERIVE/    FAT       (9)
      COMMON /DERIVE/    FAT1      (9)
      COMMON /DERIVE/    FB        (20,3)
      COMMON /DERIVE/    FC        (20,3)
      COMMON /DERIVE/    FNT       (9)
      COMMON /DERIVE/    FGM
      COMMON /DERIVE/    FGNMAX
      COMMON /DERIVE/    FORMX     (3)
      COMMON /DERIVE/    IFLOAT
      COMMON /DERIVE/    IMAX      (3)
      COMMON /DERIVE/    ISAFE     (3)
      COMMON /DERIVE/    J
      COMMON /DERIVE/    MAXI
      COMMON /DERIVE/    BGMX      (3)
      COMMON /DERIVE/    NAVERO
      COMMON /DERIVE/    CBSE
      COMMON /DERIVE/    FAV       (9)
      COMMON /DERIVE/    RTOWB
      COMMON /DERIVE/    RTOWNB
      COMMON /DERIVE/    RTOWNP
      COMMON /DERIVE/    RTOWP
      COMMON /DERIVE/    RTOWPB    (20)
      COMMON /DERIVE/    RTOWT     (20)
      COMMON /DERIVE/    STRACT    (20,3,3)
      COMMON /DERIVE/    SRFO      (9)
      COMMON /DERIVE/    SRFV      (9)
      COMMON /DERIVE/    STR       (3,9)
      COMMON /DERIVE/    TBF       (3)
      COMMON /DERIVE/    TBEN      (9)
      COMMON /DERIVE/    TRESIS    (3,9)
      COMMON /DERIVE/    VA        (3,9)
```

```
COMMON /DERIVE/    NGGOVA    (3,9)
COMMON /DERIVE/    VAVOID    (3,9)
COMMON /DERIVE/    VBO       (3,9)
COMMON /DERIVE/    IMPACT    (9)
COMMON /DERIVE/    VELV      (3)
COMMON /DERIVE/    VGMAX     (3)
COMMON /DERIVE/    VG        (2Ø,3,3)
COMMON /DERIVE/    NBGOVO    (3,9)
COMMON /DERIVE/    VMAX
COMMON /DERIVE/    VMAX1     (3)
COMMON /DERIVE/    VMAX2     (3,9)
COMMON /DERIVE/    VBLA
COMMON /DERIVE/    VBVER     (3,9)
COMMON /DERIVE/    VRID
COMMON /DERIVE/    VSEL
COMMON /DERIVE/    VSEL1     (3)
COMMON /DERIVE/    VSEL2     (3,9)
COMMON /DERIVE/    VSOIL     (3,9)
COMMON /DERIVE/    VIT       (3,9)
COMMON /DERIVE/    VXT       (3,9)
COMMON /DERIVE/    WDGONG
COMMON /DERIVE/    WRATIO
COMMON /TERRAN/    AA
COMMON /TERRAN/    AGTRMS
COMMON /TERRAN/    AREA
COMMON /TERRAN/    AREAC
COMMON /TERRAN/    CI
COMMON /TERRAN/    CIST
COMMON /TERRAN/    EANG
COMMON /TERRAN/    ECF
COMMON /TERRAN/    ELEV
COMMON /TERRAN/    FMU       (3)
COMMON /TERRAN/    GRADE
COMMON /TERRAN/    IOBS
COMMON /TERRAN/    IGST
COMMON /TERRAN/    IROAD
COMMON /TERRAN/    IS        (9)
COMMON /TERRAN/    IST
COMMON /TERRAN/    IOUT
COMMON /TERRAN/    NI
COMMON /TERRAN/    NIU
COMMON /TERRAN/    CAW
COMMON /TERRAN/    CBAA
COMMON /TERRAN/    CBH
COMMON /TERRAN/    CBL
COMMON /TERRAN/    CBS
COMMON /TERRAN/    CBW
COMMON /TERRAN/    CBMINW
COMMON /TERRAN/    CDIA
COMMON /TERRAN/    NMASHO
COMMON /TERRAN/    RADC
COMMON /TERRAN/    RCI
```

```
COMMON /TERRAN/   RCIC      (4)
COMMON /TERRAN/   FOURV     (11)
COMMON /TERRAN/   RC
COMMON /TERRAN/   RCA       (12)
COMMON /TERRAN/   S         (9)
COMMON /TERRAN/   SB        (9)
COMMON /TERRAN/   SBL       (9)
COMMON /TERRAN/   SNRFF
COMMON /TERRAN/   TANPHI
COMMON /TERRAN/   THETA     (3)
COMMON /TERRAN/   VCURV     (4, 11)
COMMON /TERRAN/   WA
COMMON /TERRAN/   WB
COMMON /SCEN/     CGHES
COMMON /SCEN/     VWALK
COMMON /SCEN/     CCLMAX
COMMON /SCEN/     GAMMA
COMMON /SCEN/     IGVER
COMMON /SCEN/     ISEASN
COMMON /SCEN/     ISURF
COMMON /SCEN/     ISNOW
COMMON /SCEN/     KII1
COMMON /SCEN/     KII2
COMMON /SCEN/     KII3
COMMON /SCEN/     KII4
COMMON /SCEN/     KII5
COMMON /SCEN/     KII6
COMMON /SCEN/     KII7
COMMON /SCEN/     KII8
COMMON /SCEN/     KII9
COMMON /SCEN/     KII10
COMMON /SCEN/     KII11
COMMON /SCEN/     KII12
COMMON /SCEN/     KII13
COMMON /SCEN/     KII14
COMMON /SCEN/     KII15
COMMON /SCEN/     KII16
COMMON /SCEN/     KII17
COMMON /SCEN/     KMAP
COMMON /SCEN/     KSCEN
COMMON /SCEN/     KIPP
COMMON /SCEN/     KVEH
COMMON /SCEN/     KIV1
COMMON /SCEN/     KIV2
COMMON /SCEN/     KIV3
COMMON /SCEN/     KIV4
COMMON /SCEN/     KIV5
COMMON /SCEN/     KIV6
COMMON /SCEN/     KIV7
COMMON /SCEN/     KIV8
COMMON /SCEN/     KIV9
COMMON /SCEN/     KIV10
```

```
            COMMON /SCEN/      KIV11
            COMMON /SCEN/      KIV12
            COMMON /SCEN/      KIV13
            COMMON /SCEN/      KIV14
            COMMON /SCEN/      KIV15
            COMMON /SCEN/      KIV16
            COMMON /SCEN/      KIV17
            COMMON /SCEN/      KIV18
            COMMON /SCEN/      KIV19
            COMMON /SCEN/      KIV20
            COMMON /SCEN/      KIV21
            COMMON /SCEN/      LAC
            INTEGER            DETAIL
            COMMON /SCEN/      DETAIL
            COMMON /SCEN/      MAP
            COMMON /SCEN/      MAPG
            COMMON /SCEN/      MONTH
            COMMON /SCEN/      NOPP
            COMMON /SCEN/      NSLIP
            COMMON /SCEN/      NTRAV
            COMMON /SCEN/      NTUX
            COMMON /SCEN/      PHI
            COMMON /SCEN/      REACT
            COMMON /SCEN/      RDFOG
            INTEGER            SEARCH
            COMMON /SCEN/      SEARCH
            COMMON /SCEN/      SETYPC
            COMMON /SCEN/      VBRAKE
            COMMON /SCEN/      VISMNV
            COMMON /SCEN/      VLIM
            COMMON /SCEN/      ZSNOW
C     2. ALGORITHM
            CALL IV1(
     +  ADO     ,ADT     ,AREAO   ,CL      ,EWDTH   ,IOBS    ,IOST    ,IS
     +  ,NEVERO ,NI      ,CAW     ,OBAA    ,OBH     ,OBL     ,OBS     ,OBSE
     +  ,ODIA   ,PAV     ,PWTE    ,S       ,SD      ,TDEN    ,WA      ,WDTH
     +  ,WI )
C           DIAGNOSTIC OUTPUT LIST
            NAMELIST /XIV1/
     +  ADO     ,ADT     ,AREAO   ,CL      ,EWDTH   ,IOBS    ,IOST    ,IS
     +  ,NEVERO ,NI      ,CAW     ,OBAA    ,OBH     ,OBL     ,OBS     ,OBSE
     +  ,ODIA   ,PAV     ,PWTE    ,S       ,SD      ,TDEN    ,WA      ,WDTH
     +  ,WI
C           DIAGNOSTIC OUTPUT
               IF(KIV1 .EQ. 1) WRITE(LUN1,XIV1)
            CALL IV2(
     +  CL      ,DAREA   ,CRAFT   ,FORDD   ,GRADE   ,IFLOAT  ,IOBS    ,IST
     +  ,ITUT   ,JPSI    ,NOGOWD  ,NOPP    ,NWR     ,PWTE    ,SRFV    ,VSEW
     +  ,VSS    ,WD      ,WDPTW   ,WDTH    ,WRAT    ,WRATIO  ,WRFORD )
C           DIAGNOSTIC OUTPUT LIST
            NAMELIST /XIV2/
     +  CL      ,DAREA   ,CRAFT   ,FORDD   ,GRADE   ,IFLOAT  ,IOBS    ,IST
```

```
     + ,ITUT    ,JPSI    ,NOGCMD ,NOPP    ,NWR     ,PWTE    ,SRFV    ,VSEL
     + ,VSS     ,WD      ,WDPTH  ,WDTH    ,WRAT    ,WRATIO  ,WRFORD
C        DIAGNOSTIC OUTPUT
           IF(KIV2 .EQ. 1) WRITE(LUN1,XIV2)
         CALL IV3(
     +  CHARLN ,CI      ,CCHES  ,CPFFG   ,DIAW    ,DOWPB   ,DRAT    ,GAMMA
     + ,GCA    ,IB      ,ID     ,IP      ,IST     ,JPSI    ,LUN1    ,NAMBLY
     + ,NPAD   ,NSLIP   ,NVEH   ,NWHL    ,RCI     ,RTOWPB  ,RTOWT   ,SECIW
     + ,TANPHI ,TRAKLN  ,TRAKWD ,VCIFG   ,VCIMUK  ,WGHT    ,WRATIO  ,ZSNOW )
C        DIAGNOSTIC OUTPUT LIST
         NAMELIST /XIV3/
     +  CHARLN ,CI      ,CCHES  ,CPFFG   ,DIAW    ,DOWPB   ,DRAT    ,GAMMA
     + ,GCA    ,IB      ,ID     ,IP      ,IST     ,JPSI    ,LUN1    ,NAMBLY
     + ,NPAD   ,NSLIP   ,NVEH   ,NWHL    ,RCI     ,RTOWPB  ,RTOWT   ,SECIW
     + ,TANPHI ,TRAKLN  ,TRAKWD ,VCIFG   ,VCIMUK  ,WGHT    ,WRATIO  ,ZSNOW
C        DIAGNOSTIC OUTPUT
         IF(KIV3 .EQ. 1) WRITE(LUN1,XIV3)
         CALL IV4(
     +  DOWB   ,DOWP    ,COWPB  ,GCW     ,GCWB    ,GCWP    ,IB      ,IP
     + ,NAMBLY ,RTOWB   ,RTCWNB ,RTCWNP  ,RTOWP   ,RTOWPB  ,RTOWT   ,WGHT )
C        DIAGNOSTIC OUTPUT LIST
         NAMELIST /XIV4/
     +  DOWB   ,DOWP    ,COWPB  ,GCW     ,GCWB    ,GCWP    ,IB      ,IP
     + ,NAMBLY ,RTOWB   ,RTCWNB ,RTCWNP  ,RTOWP   ,RTOWPB  ,RTOWT   ,WGHT
C        DIAGNOSTIC OUTPUT
         IF(KIV4 .EQ. 1) WRITE(LUN1,XIV4)
         CALL IV5(
     +  ATF    ,AVGC    ,BTF    ,CD      ,CPFCCG  ,CPFCFG  ,CTF     ,DAREA
     + ,DOWP   ,EANG    ,ECF    ,FA      ,FB      ,FC      ,FORMX   ,GCWR
     + ,IFLOAT ,IST     ,ITUT   ,JPSI    ,LOCDIF  ,NAMBLY  ,NFL     ,NGR
     + ,NTRAV  ,NVEH    ,NVEHC  ,NWHL    ,RADC    ,RTOWP   ,STRACT  ,TFOR
     + ,THETA  ,TRACTF  ,VFMAX  ,VG      ,VGV     ,WGHT    ,WRATIO  ,LUN1 )
C        DIAGNOSTIC OUTPUT LIST
         NAMELIST /XIV5/
     +  ATF    ,AVGC    ,BTF    ,CD      ,CPFCCG  ,CPFCFG  ,CTF     ,DAREA
     + ,DOWP   ,EANG    ,ECF    ,FA      ,FB      ,FC      ,FORMX   ,GCWR
     + ,IFLOAT ,IST     ,ITUT   ,JPSI    ,LOCDIF  ,NAMBLY  ,NFL     ,NGR
     + ,NTRAV  ,NVEH    ,NVEHC  ,NWHL    ,RADC    ,RTOWP   ,STRACT  ,TFOR
     + ,THETA  ,TRACTF  ,VFMAX  ,VG      ,VGV     ,WGHT    ,WRATIO
C        DIAGNOSTIC OUTPUT
           IF(KIV5 .EQ. 1) WRITE(LUN1,XIV5)
         CALL IV6(
     +  FAT    ,FAT1    ,FMT    ,NI      ,PBHT    ,SD
     + ,SDL    ,TDEN    ,WDTH )
C        DIAGNOSTIC OUTPUT LIST
         NAMELIST /XIV6/
     +  FAT    ,FAT1    ,FMT    ,NI      ,PBHT    ,SD      ,SDL     ,TDEN
     + ,WDTH
C        DIAGNOSTIC OUTPUT
         IF(KIV6 .EQ. 1) WRITE(LUN1,XIV6)
         CALL IV7(
     +  FMT    ,GCW     ,IMPACT ,MAXI    ,NI      ,PBF )
```

```
C        DIAGNOSTIC OUTPUT LIST
         NAMELIST /XIV7/
     +  FMT     ,GCW     ,IMPACT ,MAXI    ,NI      ,PBF
C        DIAGNOSTIC OUTPUT
         IF(KIV7 .EQ. 1)  WRITE(LUN1,XIV7)
         CALL IV8(
     +  FAT     ,FAT1    ,GCW    ,GCWNP   ,GCWP    ,MAXI    ,NTRAV   ,RTOWNP
     + ,RTOWP   ,STR     ,THETA  ,TRESIS  ,WRATIO )
C        DIAGNOSTIC OUTPUT LIST
         NAMELIST /XIV8/
     +  FAT     ,FAT1    ,GCW    ,GCWNP   ,GCWP    ,MAXI    ,NTRAV   ,RTOWNP
     + ,RTOWP   ,STR     ,THETA  ,TRESIS  ,WRATIO
C        DIAGNOSTIC OUTPUT
         IF(KIV8 .EQ. 1) WRITE(LUN1,XIV8)
         CALL IV9(
     +  FA      ,FB      ,FC     ,FORMX   ,MAXI    ,NGR
     + ,NI      ,NTRAV   ,TRESIS ,VFMAX   ,VG      ,VSOIL )
C        DIAGNOSTIC OUTPUT LIST
         NAMELIST /XIV9/
     +  FA      ,FB      ,FC     ,FORMX   ,MAXI    ,NGR     ,NI      ,NTRAV
     + ,TRESIS  ,VFMAX   ,VG     ,VSOIL
C        DIAGNOSTIC OUTPUT
         IF(KIV9 .EQ. 1) WRITE(LUN1,XIV9)
         CALL IV10(
     +  ACTRMS  ,LAC     ,MAXIRR ,RMS     ,VRID    ,VRIDE )
C        DIAGNOSTIC OUTPUT LIST
         NAMELIST /XIV10/
     +  ACTRMS  ,LAC     ,MAXIRR ,RMS     ,VRIDE   ,VRID
C        DIAGNOSTIC OUTPUT
         IF(KIV10 .EQ. 1) WRITE(LUN1,XIV10)
         CALL IV11(
     +  DOWB    ,GCW     ,GCWB   ,GCWNB   ,NOGOBF  ,NTRAV
     + ,RTOWB   ,RTOWNB  ,TBF    ,THETA   ,WRATIO  ,XBR )
C        DIAGNOSTIC OUTPUT LIST
         NAMELIST /XIV11/
     +  DOWB    ,GCW     ,GCWB   ,GCWNB   ,NOGOBF  ,NTRAV   ,RTOWB   ,RTOWNB
     + ,TBF     ,THETA   ,WRATIO ,XBR
C        DIAGNOSTIC OUTPUT
         IF(KIV11 .EQ. 1) WRITE(LUN1,XIV11)
         CALL IV12(
     +  BFMX    ,DCLMAX  ,GCW    ,NTRAV   ,SFTYPC  ,TBF )
C        DIAGNOSTIC OUTPUT LIST
         NAMELIST /XIV12/
     +  BFMX    ,DCLMAX  ,GCW    ,NTRAV   ,SFTYPC  ,TBF
C        DIAGNOSTIC OUTPUT
         IF(KIV12 .EQ. 1) WRITE(LUN1,XIV12)
         CALL IV13(
     +  BFMX    ,EYEHGT  ,GCW    ,NTRAV   ,RD      ,REACT   ,VELV    ,VISMNV )
C        DIAGNOSTIC OUTPUT LIST
         NAMELIST /XIV13/
     +  BFMX    ,EYEHGT  ,GCW    ,NTRAV   ,RD      ,REACT   ,VELV    ,VISMNV
C        DIAGNOSTIC OUTPUT
```

```
      IF( KIV13 .EQ. 1) WRITE(LUN1,XIV13)
      CALL IV14(
     + JPSI    ,NI      ,NTRAV   ,NVEHC   ,VELV    ,VRID
     + ,VSOIL  ,VTIRE   ,VTT )
C      DIAGNOSTIC OUTPUT LIST
       NAMELIST /XIV14/
     + JPSI    ,NI      ,NTRAV   ,NVEHC   ,VELV    ,VRID    ,VSOIL   ,VTIRE
     + ,V-T
C      DIAGNOSTIC OUTPUT
       IF( KIV14 .EQ. 1 ) WRITE(LUN1,XIV14)
       CALL IV15(
     + ADT     ,NI      ,NTRAV   ,NVEHC   ,PAV     ,VAVOID  ,VBO     ,VTT )
C      DIAGNOSTIC OUTPUT LIST
       NAMELIST /XIV15/
     + ADT     ,NI      ,NTRAV   ,NVEHC   ,PAV     ,VAVOID  ,VBO     ,VTT
C      DIAGNOSTIC OUTPUT
       IF( KIV15 .EQ. 1) WRITE(LUN1,XIV15)
      CALL IV16(
     + AVALS   ,CLEAR   ,FOM     ,FOMMAX  ,FOO     ,FOOMAX  ,HOVALS  ,NANG
     + ,NEVERO ,NOHGT   ,NWDTH   ,OBAA    ,OBH     ,OBMINW  ,WVALS )
C      DIAGNOSTIC OUTPUT LIST
       NAMELIST /XIV16/
     + AVALS   ,CLEAR   ,FOM     ,FOMMAX  ,FOO     ,FOOMAX  ,HOVALS  ,NANG
     + ,NEVERO ,NOHGT   ,NWDTH   ,OBAA    ,OBH     ,OBMINW  ,WVALS
C      DIAGNOSTIC OUTPUT
       IF( KIV16 .EQ. 1) WRITE(LUN1,XIV16)
       CALL IV17(
     + HVALS   ,NEVERO  ,NHVALS  ,NSVALS  ,OBH     ,OBSE
     + ,SVALS  ,TL      ,VOLA    ,VOOB    ,VOOBS   ,WA )
C      DIAGNOSTIC OUTPUT LIST
       NAMELIST /XIV17/
     + HVALS   ,NEVERO  ,NHVALS  ,NSVALS  ,OBH     ,OBSE    ,SVALS   ,TL
     + ,VOLA   ,VOOB    ,VOOBS   ,WA
C      DIAGNOSTIC OUTPUT
       IF( KIV17 .EQ. 1) WRITE(LUN1,XIV17)
       CALL IV18(
     + FA      ,FB      ,FC      ,FOM     ,FOMMAX  ,FORMX   ,GCW     ,JPSI
     + ,MAXI   ,NEVERO  ,NGR     ,NI      ,NTRAV   ,NVEHC   ,OBSE    ,SRFW
     + ,TL     ,TRESIS  ,VA      ,VBO     ,VELV    ,VFMAX   ,VG      ,VOLA
     + ,VRID   ,VSOIL   ,VTIRE   ,VXT     ,WA )
C      DIAGNOSTIC OUTPUT LIST
       NAMELIST /XIV18/
     + FA      ,FB      ,FC      ,FOM     ,FOMMAX  ,FORMX   ,GCW     ,JPSI
     + ,MAXI   ,NEVERO  ,NGR     ,NI      ,NTRAV   ,NVEHC   ,OBSE    ,SRFW
     + ,TL     ,TRESIS  ,VA      ,VBO     ,VELV    ,VFMAX   ,VG      ,VOLA
     + ,VRID   ,VSOIL   ,VTIRE   ,VXT     ,WA
C      DIAGNOSTIC OUTPUT
       IF( KIV18 .EQ. 1) WRITE(LUN1,XIV18)
       CALL IV19(
     + BFMX    ,FA      ,FB      ,FC      ,FOMMAX  ,FORMX   ,GCW     ,LUNI
     + ,NEVERO ,NGR     ,NI      ,NTRAV   ,OBSE    ,RMX     ,STRACT  ,TL
     + ,TRESIS ,VA      ,VBO     ,VFMAX   ,VG      ,VOVER   ,VXT     ,WA )
```

```
C          DIAGNOSTIC OUTPUT LIST
           NAMELIST /XIV19/
     +  BFMX     ,FA       ,FB       ,FC       ,FOMMAX ,FORMX    ,GCW      ,LUN1
     + ,NEVERO  ,NGR      ,NI       ,NTRAV    ,OBSE     ,RMX      ,STRACT  ,TL
     + ,TRESIS  ,VA       ,VBC      ,VFMAX    ,VG       ,VOVER    ,VXT      ,WA
C          DIAGNOSTIC OUTPUT
           IF(KIV19 .EQ. 1) WRITE(LUN1,XIV19)
           CALL IV20(
     +  FA       ,FB       ,FC       ,FORMX    ,GCW      ,MAXI     ,NGR      ,NOGOVA
     + ,NOGOVO  ,NTRAV    ,STR      ,VAVOID   ,VG       ,VOVER )
C          DIAGNOSTIC OUTPUT LIST
           NAMELIST /XIV20/
     +  FA       ,FB       ,FC       ,FORMX    ,GCW      ,MAXI     ,NGR      ,NOGOVA
     + ,NOGOVO  ,NTRAV    ,STR      ,VAVOID   ,VG       ,VOVER
C          DIAGNOSTIC OUTPUT
           IF(KIV20 .EQ. 1) WRITE(LUN1,XIV20)
           CALL IV21(
     +  DOWN     ,FMT      ,GCW      ,IMAX     ,IOVER    ,ISAFE    ,LEVEL    ,NI
     + ,NOGOVA  ,NOGOVO   ,NTRAV    ,UP       ,VAVOID   ,VMAX     ,VMAX1    ,VMAX2
     + ,VOVER   ,VSEL     ,VSEL1    ,VSEL2    ,VWALK )
C          DIAGNOSTIC OUTPUT LIST
           NAMELIST /XIV21/
     +  DOWN     ,FMT      ,GCW      ,IMAX     ,IOVER    ,ISAFE    ,LEVEL    ,NI
     + ,NOGOVA  ,NOGOVO   ,NTRAV    ,UP       ,VAVOID   ,VMAX     ,VMAX1    ,VMAX2
     + ,VOVER   ,VSEL     ,VSEL1    ,VSEL2    ,VWALK
C          DIAGNOSTIC OUTPUT
           IF(KIV21 .EQ. 1) WRITE(LUN1,XIV21)
C      3.  TERMINUS
           CONTINUE
           RETURN
           END
       SUBROUTINE MAP71
     + (AA       ,ACTRMS   ,AREA     ,ELEV     ,GRADE    ,IEOF     ,IOST     ,IS
     + ,IST      ,ITUT     ,LUN2     ,NI       ,NTU      ,NTUX     ,OBH      ,OBL
     + ,OBS      ,OBW      ,RCIC     ,RD       ,S        ,SD       ,SDL      ,SEARCH
     + ,WD )
C
C      ------------------------------------
C      MAP LEGEND INPUT ROUTINE (AMC71 FORMAT)
C      ------------------------------------
C
C      1. VARIABLE DEFINITION
          INTEGER    IS      (9)
          REAL       RCIC    (4)
          REAL       S       (9)
          REAL       SD      (9)
          REAL       SDL     (9)
          INTEGER    SEARCH
          REAL       FNAA    (14)
          REAL       FNGRAD  (8)
          REAL       FNOBH   (7)
          REAL       FNOBL   (7)
```

```
      REAL      FNOBS   (8)
      REAL      FNOBW   (5)
      REAL      FNRCI   (11)
      REAL      FNRD    (9)
      REAL      FNRMS   (9)
      REAL      FNSPAC  (8)
C    3. TERRAIN FEATURE CLASSES
C       A. SURFACE STRENGTH CLASSES   LB/SQ0IN
           DATA FNRCI(1)   /300.0/,
     +          FNRCI(2)   /250.0/,
     +          FNRCI(3)   /190.0/,
     +          FNRCI(4)   /130.0/,
     +          FNRCI(5)   / 80.0/,
     +          FNRCI(6)   / 50.0/,
     +          FNRCI(7)   / 36.0/,
     +          FNRCI(8)   / 29.0/,
     +          FNRCI(9)  / 20.0/,
     +          FNRCI(10) / 14.0/,
     +          FNRCI(11) /  5.0/
C       B. GRADE CLASSES   PERCENT
           DATA FNGRAD(1)  / 1.0/,
     +          FNGRAD(2)  / 3.5/,
     +          FNGRAD(3)  / 7.5/,
     +          FNGRAD(4)  /15.0/,
     +          FNGRAD(5)  /30.0/,
     +          FNGRAD(6)  /50.0/,
     +          FNGRAD(7)  /65.0/,
     +          FNGRAD(8)  /72.0/
C       C. RECOGNITION DISTANCE CLASSES   FT
           DATA FNRD(1)   /164.0/,
     +          FNRD(2)   /121.0/,
     +          FNRD(3)   / 59.0/,
     +          FNRD(4)   / 34.8/,
     +          FNRD(5)   / 24.6/,
     +          FNRD(6)   / 17.4/,
     +          FNRD(7)   / 12.5/,
     +          FNRD(8)   /  7.5/,
     +          FNRD(9)   /  2.6/
C       D. OBSTACLE SPACING CLASSES   FT
           DATA FNOBS(1)  /197.0/,
     +          FNOBS(2)  /131.0/,
     +          FNOBS(3)  / 51.2/,
     +          FNOBS(4)  / 31.5/,
     +          FNOBS(5)  / 22.3/,
     +          FNOBS(6)  / 15.7/,
     +          FNOBS(7)  / 10.8/,
     +          FNOBS(8)  /  3.9/
C       E. OBSTACLE APPROACH ANGLE CLASSES   DEG
           DATA FNAA(1)  /179.0/,
     +          FNAA(2)  /181.0/,
     +          FNAA(3)  /127.0/,
     +          FNAA(4)  /183.0/,
```

```
     +            FNAA(5)   /173.0/,
     +            FNAA(6)   /187.0/,
     +            FNAA(7)   /164.0/,
     +            FNAA(8)   /196.0/,
     +            FNAA(9)   /154.0/,
     +            FNAA(10)  /206.0/,
     +            FNAA(11)  /142.0/,
     +            FNAA(12)  /218.0/,
     +            FNAA(13)  /112.0/,
     +            FNAA(14)  /248.0/
C         F. OBSTACLE HEIGHT CLASSES    IN
             DATA FNOBH(1)   / 3.15/,
     +            FNOBH(2)   / 7.87/,
     +            FNOBH(3)   /11.81/,
     +            FNOBH(4)   /15.75/,
     +            FNOBH(5)   /20.87/,
     +            FNOBH(6)   /28.35/,
     +            FNOBH(7)   /43.46/
C         G. OBSTACLE WIDTH CLASSES   FT
             DATA FNOBW(1)   /11.80/,
     +            FNOBW(2)   / 3.48/,
     +            FNOBW(3)   / 2.49/,
     +            FNOBW(4)   / 1.51/,
     +            FNOBW(5)   / 0.49/
C         H. OBSTACLE LENGTH CLASSES    FT
             DATA FNOBL(1)   /  0.66/,
     +            FNOBL(2)   /  2.36/,
     +            FNOBL(3)   /  5.25/,
     +            FNOBL(4)   /  8.53/,
     +            FNOBL(5)   / 15.09/,
     +            FNOBL(6)   /256.00/,
     +            FNOBL(7)   /492.0/
C         I. SURFACE ROUGHNESS CLASSES    IN
             DATA FNRMS(1)   /0.25/,
     +            FNRMS(2)   /1.00/,
     +            FNRMS(3)   /2.00/,
     +            FNRMS(4)   /3.00/,
     +            FNRMS(5)   /4.00/,
     +            FNRMS(6)   /5.00/,
     +            FNRMS(7)   /6.00/,
     +            FNRMS(8)   /7.00/,
     +            FNRMS(9)   /8.00/
C         J. VEGATATION SPACING CLASSES    FT
             DATA FNSPAC(1)  /300.0/,
     +            FNSPAC(2)  / 65.6/,
     +            FNSPAC(3)  / 51.2/,
     +            FNSPAC(4)  / 31.5/,
     +            FNSPAC(5)  / 22.3/,
     +            FNSPAC(6)  / 15.7/,
     +            FNSPAC(7)  / 10.8/,
     +            FNSPAC(8)  /  3.9/
C      4. READ LEGEND
```

```
400       CONTINUE
             IEOF = 0
             READ(LUN2,4000)
     +   NPAT      ,IST        ,IRCI1      ,IRCI2      ,IRCI3
     +  ,IGRADE    ,IOA        ,IOH        ,IOW        ,IOL
     +  ,IOS       ,IOST       ,IRMS       ,ISTEM1     ,ISTEM2
     +  ,ISTEM3    ,ISTEM4     ,ISTEM5     ,ISTEM6     ,ISTEM7
     +  ,ISTEM8    ,IRD        ,AREA
4000         FORMAT(I4,4I2,I1,I2,15I1,F10.3)
             IF(EOF(LUN2) .EQ. 0) GO TO 420
                IEOF = 1
                GO TO 600
420          CONTINUE
          IF((SEARCH .EQ. 1) .AND. (NTUX .NE. NPAT)) GO TO 400
C      5. CONVERT FROM CLASSE TO REAL UNITS
500       CONTINUE
C         A. TERRAIN UNIT NUMBER
             NTU=NPAT
C         B. TERRAIN UNIT TYPE
             ITUT=1
C         C. SOIL TYPE
C            IST
C         D. SURFACE STRENGTH   LB/SQ-IN
             RCIC(1)=FNRCI(IRCI1)
             RCIC(2)=FNRCI(IRCI2)
             RCIC(3)=FNRCI(IRCI3)
C         E. GRADE   PERCENT
             GRADE=FNGRAD(IGRADE)
C         F. SURFACE ROUGHNESS   IN
             ACTRMS=FNRMS(IRMS)
C         G. VISIBILITY   FT
             RD=FNRD(IRD)
C         H. DEPTH OF STANDING WATER   FT
             WD=0.0
C         I. ELEVATION   FT
             ELEV=0.0
C         K. OBSTACLE SPACING   FT
             OBS=FNOBS(IOS)
C         L. OBSTACLE AVOIDABILITY POTENTIAL
C            IOST
C         M. OBSTACLE APPROACH ANGLE   DEGREES
             AA=FNAA(IOA)
C         N. OBSTACLE HEIGHT   IN
             OBH=FNOBH(IOH)
C         O. OBSTACLE WIDTH   FT
             OBW=FNOBW(IOW)
C         P. OBSTACLE LENGTH   FT
             OBL=FNOBL(IOL)
C         R. NUMBER OF STEM CLASSES
             NI=8
C         S. MEAN SPACING OF STEMS
             S(1)=FNSPAC(ISTEM1)
```

```
                S(2)=FNSPAC( ISTEM2)
                S(3)=FNSPAC( ISTEM3)
                S(4)=FNSPAC( ISTEM4)
                S(5)=FNSPAC( ISTEM5)
                S(6)=FNSPAC( ISTEM6)
                S(7)=FNSPAC( ISTEM7)
                S(8)=FNSPAC( ISTEM8)
C          T. MAXIMUM STEM DIAMETER    IN
                SDL(1)=0.98
                SDL(2)=2.36
                SDL(3)=3.94
                SDL(4)=5.51
                SDL(5)=7.09
                SDL(6)=8.66
                SDL(7)=9.84
                SDL(8)=15.00
C          U. MEAN STEM DIAMETER    IN
                SD(1)=0.49
                SD(2)=1.67
                SD(3)=3.15
                SD(4)=4.73
                SD(5)=6.30
                SD(6)=7.88
                SD(7)=9.25
                SD(8)=12.42
C          V. BARREN VEGATATION FLAG
                NI1=NI-1
                DO 540 I=1,NI1
                   IF( S(I) .EQ. S(I+1) ) GO TO 530
                      IS(I)=0
                      GO TO 540
530                CONTINUE
                   IS(I)=1
540             CONTINUE
                IS(NI)=0
C       6. EXIT ROUTINE
600         CONTINUE
            RETURN
            END
        SUBROUTINE MAP74
     + (AA      ,ACTRMS ,AREA   ,ELEV   ,GRADE  ,IEOF   ,IOST   ,IS
     + ,IST     ,ITUT   ,LUN2   ,MONTH  ,NI     ,NTU    ,NTUX   ,OBH
     + ,OBL     ,OBS    ,OBW    ,RCIC   ,RD     ,RDA    ,RDA1   ,RDA2
     + ,RDA3    ,RDA4   ,S      ,SD     ,SDL    ,SEARCH ,WD )
C
C       -------------------------------------------------
C       MAP LEGEND INPUT ROUTINE (AMC74 FORMAT)
C       -------------------------------------------------
C
C       1. GLOSSARY
C          AA        DEGREE     OBSTACLE - APPROACH ANGLE
C          ACTRMS    IN         RMS ROUGHNESS
```

```
C          AREA      SQ. MI.    AREA
C          ELEV      FT         ELEVATION
C          GRADE     PERCENT    SLOPE
C          IOBS      #          BARREN OBSTACLE FLAG
C          IOST      1-RANDOM   OBSTACLE - SPACING TYPE
C                    2-LINEAR
C          IS(I)     #          BARREN VEGETATION FLAG, CLASS I
C          ITUT      #          TERRAIN UNIT TYPE
C          IEOF      #          END OF INFORMATION FLAG
C          KMAP      #          CALL LIST DUMP FLAG
C          LUN2      #          LOGICAL UNIT FLAG NUMBER 2, TERRAIN
C          MONTH     #          MONTH ( 1 THROUGH 12 )
C          NI        #          NUMBER OF VEGETATION CLASSES
C          NTU       #          TERRAIN UNIT NUMBER
C          OBH       INCH       OBSTACLE - HEIGHT
C          OBL       FEET       OBSTACLE LENGTH
C          OBS       FEET       OBSTACLE - SPACING
C          OBW       INCH       OBSTACLE - WIDTH
C          RCIC(1)   RCI        SOIL STRENGTH - DRY
C          RCIC(2)   RCI        SOIL STRENGTH - AVERAGE
C          RCIC(3)   RCI        SOIL STRENGTH - WET
C          RCIC(4)   RCI        SOIL STRENGTH - WET,WET
C          RD        FEET       VISIBILITY
C          S(I)      FEET       STEM SPACING OF STEMS OF DIAMETER CLASS
C          SD(I)     IN         MEAN STEM DIAMETER, CLASS I
C          SDL(I)    IN         MAXIMUM STEM DIAMETER, CLASS I
C          WD        FEET       DEPTH OF STANDING WATER
C      2. VARIABLE DECLARATION
           INTEGER    IS      (9)
           REAL       RCIC    (4)
           REAL       RDA     (12)
           REAL       S       (9)
           REAL       SD      (9)
           REAL       SDL     (9)
           INTEGER    SEARCH
           INTEGER    DATA    (24)
C      4. ALGORITHM
4000       CONTINUE
              IEOF = 0
              READ(LUN2,400)NTU,IST,DATA,AREA
400           FORMAT(I5,I2,20X,12I4,/,5X,12I4,F8.4)
                IF(EOF(LUN2) .EQ. 0) GO TO 4020
                  IEOF = 1
                  GO TO 5000
4020            RCIC(1)   = DATA(1)
                RCIC(2)   = DATA(2)
                RCIC(3)   = DATA(3)
                RCIC(4)   = DATA(4)
                GRADE     = DATA(5)
                AA        = DATA(6)
                OBH       = DATA(7)
                OBW       = DATA(8)
```

```
              OBW       = OBW / 12.
              OBL       = DATA(9)
              OBS       = DATA(10)
              IOST      = DATA(11)
              ACTRMS    = DATA(12)
              DO 4005 I=1,8
              S(I)      = DATA(I+12)
4005          CONTINUE
              RDA1      = DATA(21)
              RDA2      = DATA(22)
              RDA3      = DATA(23)
              RDA4      = DATA(24)
           CONTINUE
        IF((SEARCH .EQ. 1) .AND. (NTUX .NE. NTU)) GO TO 4000
        RDA(1)  = RDA1
        RDA(2)  = RDA1
        RDA(3)  = RDA1
        RDA(4)  = RDA2
        RDA(5)  = RDA2
        RDA(6)  = RDA2
        RDA(7)  = RDA3
        RDA(8)  = RDA3
        RDA(9)  = RDA3
        RDA(10) = RDA4
        RDA(11) = RDA4
        RDA(12) = RDA4
        ACTRMS  = ACTRMS/10.0
        ELEV    = 0.0
        ITUT    = 1
        NI      = 8
        NI1     = NI-1
        DO 4040 I=1,NI1
           IF(S(I) .EQ. S(I+1)) GO TO 4030
              IS(I) = 0
              GO TO 4040
4030       CONTINUE
           IS(I) = 1
4040    CONTINUE
        IS(NI)  = 0
        RD      = RDA(MCNT+)
        SD(1)   =  0.49
        SD(2)   =  1.67
        SD(3)   =  3.15
        SD(4)   =  4.73
        SD(5)   =  6.30
        SD(6)   =  7.88
        SD(7)   =  9.25
        SD(8)   = 12.42
        SDL(1)  =  0.98
        SDL(2)  =  2.36
        SDL(3)  =  3.94
        SDL(4)  =  5.51
```

```
         SDL(5)  =  7.09
         SDL(6)  =  8.66
         SDL(7)  =  9.84
         SDL(8)  = 15.00
         WD      =  0.
5000     CONTINUE
         RETURN
         END
       SUBROUTINE MPRD74
     + (ACTRMS ,CURVV  ,DIST   ,EANG   ,ELEV   ,FMU    ,GRADE
     + ,IEOF   ,IROAD  ,IST    ,ITUT   ,IURB   ,LUN2   ,NTU    ,NTUX
     + ,NVASHO ,MONTH  ,RADC   ,RCURV  ,RD     ,RDA    ,RDFOG  ,RCIC
     + ,SEARCH ,SURFF  ,VCURV )
C
C      ----------------------------------------
C      MAP LEGEND INPUT ROUTINE (ROAD MAP AMC74)
C      ----------------------------------------
C
C      1. GLOSSARY
C         ACTRMS    INCH       RMS ROUGHNESS
C         CURVV     MPH        AASHO CURVATURE SPEED LIMIT
C         DIST      MILES      ROAD SEGMENT LENGTH
C         EANG      DEGREE     SUPERELEVATION ANGLE
C         ELEV      FEET       ELEVATION ABOVE SEALEVEL
C         FMU(1)    #          COEFFICIENT OF FRICTION - DRY
C         FMU(2)    #          COEFFICIENT OF FRICTION - WET
C         FMU(3)    #          COEFFICIENT OF FRICTION - ICE
C         GRADE     PERCENT    SLOPE
C         IROAD     #          ROAD TYPE
C         ITUT      #          TERRAIN UNIT TYPE
C         IST       #          SOIL TYPE
C         IURB      #          URBAN CODE
C         NTU       #          TERRAIN UNIT NUMBER
C         NTUX      #          SPECIFIC TERRAIN UNIT NUMBER
C         MONTH     #          MONTH (1 THROUGH 12)
C         RADC      FEET       RADIUS OF CURVATURE
C         RD        FEET       VISIBILITY DISTANCE
C         RDA(12)   FEET       VISIBILITY DISTANCE PER MONTH
C         RDFOG     FEET       WEATHER RECOGNITION DISTANCE
C         RCIC(1)   RCI        SOIL STRENGHT - DRY
C         RCIC(2)   RCI        SOIL STRENGHT - AVERAGE
C         RCIC(3)   RCI        SOIL STRENGTH - WET
C         RCIC(4)   RCI        SOIL STRENGTH - WET,WET
C         SURFF     #          ROAD SURFACE ROUGHNESS FLAG
C
C      2. VARIABLE DECLARATION
         REAL      RCIC    (4)
         REAL      RCURV   (11)
         REAL      RDA     (12)
         REAL      FMU     (3)
         REAL      VCURV   (4,11)
         INTEGER   RDATA   (11)
```

```
      INTEGER   SEARCH
C
C     3. ALGORITHM
C        A. AASHO TABLE RADIUS OF CURVATURE           FT
                    RCURV(1)   =5730.
                    RCURV(2)   =1910.
                    RCURV(3)   =1146.
                    RCURV(4)   = 819.
                    RCURV(5)   = 637.
                    RCURV(6)   = 458.
                    RCURV(7)   = 327.
                    RCURV(8)   = 229.
                    RCURV(9)   = 164.
                    RCURV(10)  = 115.
                    RCURV(11)  =  82.
C        B. AASHO TABLE SPEED LIMIT SUPERHIGHWAYS     MPH
                    VCURV(1,1)   =100.
                    VCURV(1,2)   =70.
                    VCURV(1,3)   =60.
                    VCURV(1,4)   =54.
                    VCURV(1,5)   =48.
                    VCURV(1,6)   =41.
                    VCURV(1,7)   =34.
                    VCURV(1,8)   =29.
                    VCURV(1,9)   =25.
                    VCURV(1,10) =19.
                    VCURV(1,11) =13.
C        C. AASHO TABLE SPEED LIMIT SECONDARY ROADS   MPH
                    VCURV(3,1)   =70.
                    VCURV(3,2)   =60.
                    VCURV(3,3)   =58.
                    VCURV(3,4)   =50.
                    VCURV(3,5)   =43.
                    VCURV(3,6)   =36.
                    VCURV(3,7)   =31.
                    VCURV(3,8)   =26.
                    VCURV(3,9)   =23.
                    VCURV(3,10) =19.
                    VCURV(3,11) =13.
C        D. AASHO TABLE SPEED LIMIT TRAILS            MPH
                    VCURV(4,1)   =55.
                    VCURV(4,2)   =49.
                    VCURV(4,3)   =44.
                    VCURV(4,4)   =42.
                    VCURV(4,5)   =39.
                    VCURV(4,6)   =34.
                    VCURV(4,7)   =29.
                    VCURV(4,8)   =23.
                    VCURV(4,9)   =19.
                    VCURV(4,10) =14.
                    VCURV(4,11) =10.
C        E. AASHO TABLE SPEED LIMIT PRIMARY ROADS     MPH
```

```
                  VCURV(2,1)  = VCURV(1,1)
                  VCURV(2,2)  = VCURV(1,2)
                  VCURV(2,3)  = VCURV(1,3)
                  VCURV(2,4)  = VCURV(1,4)
                  VCURV(2,5)  = VCURV(1,5)
                  VCURV(2,6)  = VCURV(1,6)
                  VCURV(2,7)  = VCURV(1,7)
                  VCURV(2,8)  = VCURV(1,8)
                  VCURV(2,9)  = VCURV(1,9)
                  VCURV(2,10) = VCURV(1,10)
                  VCURV(2,11) = VCURV(1,11)
6000      CONTINUE
             IEOF = 0
             READ(LUN2,600) NTU,IROAD,IST,IURB,RDATA,DIST
600          FORMAT(I5,3I2,11I4,F8.4)
               IF(EOF(LUN2) .EQ. 0) GO TO 6020
                 IEOF = 1
                 GO TO 7000
6020           RCIC(1)   = RDATA(1)
               RCIC(2)   = RDATA(2)
               RCIC(3)   = RDATA(3)
               RCIC(4)   = RDATA(4)
               GRADE     = RDATA(5)
               RDA1      = RDATA(6)
               RDA2      = RDATA(7)
               RDA3      = RDATA(8)
               RDA4      = RDATA(9)
               ACTRMS    = RDATA(10)
               CURVV     = RDATA(11)
             CONTINUE
          IF((SEARCH .EQ. 1) .AND. (NTUX .NE. NTU)) GO TO 6000
              RDA(1)   = RDA1
              RDA(2)   = RDA1
              RDA(3)   = RDA1
              RDA(4)   = RDA2
              RDA(5)   = RDA2
              RDA(6)   = RDA2
              RDA(7)   = RDA3
              RDA(8)   = RDA3
              RDA(9)   = RDA3
              RDA(10)  = RDA4
              RDA(11)  = RDA4
              RDA(12)  = RDA4
              ACTRMS   = ACTRMS/10.
              ELEV     = 0.
              EANG     = 0.
              FMU(1)   = 0.75
              FMU(2)   = 0.35
              FMU(3)   = 0.1
              ITUT     = IROAD + 10
              SURFF  = 1.
              IF(ITUT .EQ. 13) SURFF = 2.
```

```
              NVASHO  = 11
              IF(ITUT .EQ. 11) RDROAD = 1000.
                 IF(ITUT .EQ. 12) RDROAD = 500.
                    IF(ITUT .EQ. 13) RDROAD = 250.
                       IF(ITUT .EQ. 14) RDROAD = 150.
              RD = AMIN1(RDA(MONTH),RDFOG,RDROAD)
              IF(CURVV .GT. VCURV(IROAD,1)) GO TO 6060
                 IF(CURVV .LT. VCURV(IROAD,NVASHO)) GO TO 6070
                    DO 6030 NV=2,NVASHO
                       IF(CURVV .LT. VCURV(IROAD,NV)) GO TO 6030
                          RADC = RCURV(NV) + (CURVV - VCURV(IROAD,NV))*
     +                          (RCURV(NV) - RCURV(NV-1))/
     +                          (VCURV(IROAD,NV) - VCURV(IROAD,NV-1))
                          GO TO 7000
6030                CONTINUE
                    GO TO 7000
6060          RADC = RCURV(1) + (CURVV - VCURV(IROAD,1))*
     +               (RCURV(1) - RCURV(2))/
     +               (VCURV(IROAD,1) - VCURV(IROAD,2))
              GO TO 7000
6070             RADC = RCURV(NVASHO) + (CURVV - VCURV(IROAD,NVASHO))*
     +                  (RCURV(NVASHO) - RCURV(NVASHO-1))/
     +                  (VCURV(IROAD,NVASHO) - VCURV(IROAD,NVASHO-1))
7000        CONTINUE
           RETURN
           END
        SUBROUTINE TPP
     + (AA      ,ACTRMS ,AREAG  ,CI     ,ECF    ,ELEV   ,GAMMA  ,GRADE
     + ,IOBS    ,ISEASN ,ISNOW  ,IST    ,ITUT   ,NI     ,OAW    ,OBAA
     + ,OBH     ,OBL    ,OBMINW ,OBS    ,OBW    ,ODIA   ,PHI    ,RADC
     + ,RCI     ,RCIC   ,RD     ,S      ,TANPHI ,THETA  ,WA     ,ZSNOW )
C
C       -------------------
C       TERRAIN PREPROCESSOR
C       -------------------
C
C       1. VARIABLE DECLARATION
           REAL      RCIC   (4)
           REAL      S      (9)
           REAL      THETA  (3)
C       3. ALGORITHM
C          A. UNITS CONVERSION
              IF(ITUT .GE. 11) GO TO 3005
              IOBS= 0
              IF( OBS .GE. 197.0 ) IOBS = 1
             IF( AA .GE. 179. .AND. AA .LE. 181. ) IOBS = 1
              OBL = 12.*OBL
              OBS = 12.*OBS
              OBW = 12.*OBW
3005          RD  = 12.*RD
              RADC= 12.*RADC
              IF(ITUT .GE. 11) GO TO 3015
```

```
            DO 3010 I=1,NI
               S(I) = 12.*S(I)
3010        CONTINUE
C        B. SLOPE ANGLE
3015        THETA(1) = ATAN(GRADE/100.)
            THETA(2) = 0.0
            THETA(3) = -THETA(1)
C        C. CONE INDEX
            CI = RCIC(ISEASN)
C        D. RATING CONE INDEX
            RCI = CI
C        E. ELEVATION CORRECTION FACTOR
            ECF = 1.-.04*ELEV/1000.
            IF(ITUT .GE. 11) GO TO 3030
C        F. OBSTACLE APPROACH ANGLE
            IF(IOBS .EQ. 1) GO TO 3020
            OBAA = AA*3.14159265/180.
C        G. OBSTACLE GEOMETRY VARIABLES
            IF(AA .GT. 180.) GO TO 3017
               IF((OBW/2.) .GT. (OBH*ABS(COS(OBAA)/SIN(OBAA))))
     *            GO TO 3016
                  OBW = 2.*OBH*ABS(COS(OBAA)/SIN(OBAA))
3016           OBMINW  = OBW - 2.*OBH*ABS(COS(OBAA)/SIN(OBAA))
               WA      = OBW
               GO TO 3018
3017        OBMINW = OBW
            WA     = OBW + 2.*OBH*ABS(COS(OBAA)/SIN(OBAA))
3018           AREAO = 3.14159265*OBS*OBS/4.
               OAW   = 2.*(OBL + WA)/3.14159265
               ODIA  = (OBL*OBL + WA*WA)**0.5
               GO TO 3030
3020        CONTINUE
            WA     = 0.0
            ODIA   = 0.0
            OAW    = 0.0
            AREAO = 0.0
3030        CONTINUE
C        H. SNOW MACHINE
            IF(ITUT .EQ. 2) GO TO 3060
            IF(ISNOW .EQ. 0) GO TO 3060
               IST = 4
               TANPHI = SIN(PHI)/COS(PHI)
C              OBSTACLE ATTENUATION
               IF((OBAA .GT. 3.14159265) .OR. (ITUT .GE. 11)) GO TO 3040
                  OBH = OBH-ZSNOW*GAMMA/0.8
3040           CONTINUE
C              SURFACE ROUGHNESS ATTENUATION
               IF(ZSNOW .LT. 2.0*ACTRMS) GO TO 3050
                  ACTRMS = ACTRMS*( 1.0-(1.0-GAMMA/0.4) )
                  GO TO 3060
3050           CONTINUE
               ACTRMS = ACTRMS*(1.-.5*(1.-GAMMA/.4)*(ZSNOW/ACTRMS))
```

```
3060         CONTINUE
             RETURN
          END
       SUBROUTINE IV1
      + (ADO      ,ADT      ,AREAO    ,CL       ,EWDTH    ,IOBS     ,IOST     ,IS
      + ,NEVERO  ,NI       ,CAW      ,OBAA     ,OBH      ,OBL      ,OBS      ,OBSE
      + ,ODIA    ,PAV      ,PWTE     ,S        ,SD       ,TDEN     ,WA       ,WDTH
      + ,WI )
C
C        ---------------------------------------------------------------------
C        EFFECTIVE OBSTACLE SPACING AND SPEED REDUCTION FACTORS DUE
C        TO VEGETATION AND/OR OBSTACLE AVOIDANCE
C        ---------------------------------------------------------------------
C
C        1. VARIABLE DECLARATION
            REAL        ADT      (9)
            INTEGER     STEM
            INTEGER     IS       (9)
            REAL        PAV      (9)
            REAL        S        (9)
            REAL        SD       (9)
            REAL        TDEN     (9)
C        3. ALGORITHM
            NEVERO = 0
C           A. OBSTACLE SPACEING AND STUMP/BOULDER INTERFERENCE CHECK
               IF(IOBS .NE. 1) GO TO 3010
C              1. PATCH BARE OF OBSTACLES
                  NEVERO = 2
                  ADO    = 0.
                  GO TO 3070
C              2. PATCH CONTAINS OBSTACLES
3010              CONTINUE
                  IF(IOST .NE. 2) GO TO 3020
C                 A. OBSTACLE ARE UNAVOIDABLE, UNABLE TO MANEUVER
                     OBSE   = OBS
                     ADO    = 100.
                     GO TO 3070
C                 B. OBSTCLE ARE POTENTIALY AVOIDABLE
3020                 IF(ODIA .LT. WI) GO TO 3030
C                    1. OBSTACLE IS WIDER THAN MINIMUM WIDTH
C     + BETWEEN RUNNING GEAR ELEMENTS.
                        EWDTH = WDTH+CAW
                        OBSE  = AREAO/EWDTH
                        GO TO 3060
C                    2. OBSTACLE NARROWER THAN THE MINIMUM WIDTH
C     + BETWEEN RUNNING GEAR ELEMENTS.
3030                    CONTINUE
                        EWDTH = PWTE+CAW
                        OBSE  = AREAO/EWDTH
                        IF( OBAA .GE. 3.14159265 ) GO TO 3050
C                       A. OBSTACLE IS CONVEX, (BOULDER/STUMP)
                           IF((CL-OBH) .GT. 0.) GO TO 3040
```

```
C                             1. BELLY HANGUP
                                 NEVERO = 1
                                 EWDTH  = WDTH+OAW
                                 OBSE   = AREAO/EWDTH
C                             2. NO BELLY INTERFERENCE
3040                             CONTINUE
C                          B. OBSTACLE IS CONCAVE, (TRENCH/DEPRESSION)
3050                          CONTINUE
C           B. AREA DENIED DUE TO AVOIDING OBSTACLES, (ADO)
3060           CONTINUE
               ADO = 100.*(OBL*WA+(OBL+WA)*WDTH+WDTH*WDTH*3.14159265/4.)
     + /( OBSE*OBSE*3.14159265/4. )
C           C. VEGETATION DENSITY
3070           CONTINUE
               NIA = NI-1
               NIB = NI+1
               DO 3090 I=1,NIA
                  IF( IS(I) .NE. 1) GO TO 3080
                     TDEN(I)=0.
                     GO TO 3090
3080              TDEN(I) = (4./3.14159265)*(1./S(I)**2-1./S(I+1)**2)
3090           CONTINUE
               IF(IS(NI) .NE. 1) GO TO 3100
                  TDEN(NI)=0.
                  GO TO 3110
3100           TDEN(NI)  = 4./(3.14159265*S(NI)**2)
               TDEN(NIB) = 0.0
C           D. AREA DENIED BY VEGETATION, PAV(I)
3110           CONTINUE
               DO 3160 I=1,NI
                  SUMA = 0.
                  DO 3130 STEM=I,NI
                     SUMA = SUMA+TDEN(STEM)
3130              CONTINUE
                  IF(SUMA .NE. 0.) GO TO 3140
                     PAV(I) = 0.
                     GO TO 3160
3140              CONTINUE
                  SUMB = 0.
                  DO 3150 STEM=I,NI
                     SUMB = SUMB+SD(STEM)*TDEN(STEM)
3150              CONTINUE
                  PAV(I)=100.*((SUMB/SUMA+WDTH)/S(I))**2
3160           CONTINUE
               PAV(NI+1) = 0.
C           E. TOTAL AREA DENIED DUE TO OBSTACLES AND VEGETATION
3210           CONTINUE
               DO 3240 I=1,NIB
                  ADT(I)=ADO+PAV(I)*(100.-ADO)/100.
3240           CONTINUE
C           F. IS THERE ANY PENALTY FOR OBSTACLE AVOIDANCE
               DO 3250 I=1,NIB
```

```
                IF(ADT(I) .NE. PAV(I)) GO TO 3260
3250          CONTINUE
              NEVERO = 2
3260          CONTINUE
              RETURN
            END
          SUBROUTINE IV2
         + (CL      ,DAREA  ,CRAFT  ,FORDD  ,GRADE  ,IFLOAT ,IOBS   ,IST
         + ,ITUT    ,JPSI   ,NOGOWD ,NOPP   ,NWR    ,PWTE   ,SRFV   ,VSEL
         + ,VSS     ,WD     ,WDPTH  ,WDTH   ,WRAT   ,WRATIO ,WRFORD )
C
C         ---------------------------
C         LAND/MARSH OPERATING FACTORS
C         ---------------------------
C
C         1. VARIABLE DECLARATION
            REAL        SRFV    (9)
            REAL        WDPTH   (20)
            REAL        WRAT    (20)
C         3. ALGORITHM
C            A. SET OPERATION TYPE
               NOGOWD = 0
               IF(ITUT .EQ. 1) GO TO 3010
               IF(IST  .EQ. 4) GO TO 3010
                  GO TO 3020
C              1. DRY LAND OPERATION
3010              CONTINUE
                  IFLOAT =    0
                  WRATIO = 1.0
                  DAREA  = 0.0
                  GO TO 2100
C              2. MARSH OPERATION
3020              CONTINUE
                  IOBS   =    1
                  GRADE = 0.0
                  IF(WD     .LE. FORDD) GO TO 3040
C                 A. WATER TO DEEP TO FORD
                     IF(CRAFT .NE.    0.0) GO TO 3030
C                    1. WATER TOO DEEP FOR OPERATION
                        NOGOWD =    1
                        VSEL   = 0.0
                        RETURN
C                    2. VEHICLE FULLY FLOATING
3030                    CONTINUE
                        IFLOAT = 0
                        VSEL    = VSS*SRFV(2)
                        RETURN
C                 B. VEHICLE IS FORDING
3040                 CONTINUE
                     IFLOAT = 1
                     IF(WD .GT. WDPTH(1)) GO TO 3050
                        WRATIO = 1.0+WD*( WRAT(1)-1.0 )/WDPTH(1)
```

```
                    GO TO 3090
3050            CONTINUE
                DO 3080 N=2,NWR
                   IF(WD .GT. WDPTH(N)) GO TO 3070
                      WRATIO = WRAT(N-1)+(WRAT(N)-WRAT(N-1))*
     + (WD-WDPTH(N-1))/(WDPTH(N)-WDPTH(N-1))
                      GO TO 3090
3070               CONTINUE
3080            CONTINUE
                WRATIO = WRAT(NWR)+(WRFORD-WRAT(NWR))*
     + (WD-WDPTH(NWR))/(FORDE-WDPTH(NWR))
3090            CONTINUE
                DAREA  = 2.0*PWTE*WD
                IF(WD .GT. CL) DAREA=WDTH*(WD-CL)+2.0*PWTE*CL
2100            CONTINUE
C        8. TIRE PRESSURE INDEX
            IF(NOPP .NE. 0) GO TO 2110
               JPSI = 2
               IF(IST .EQ. 1) JPSI = 1
               IF(IST .EQ. 3) JPSI = 1
               IF(IST .EQ. 4) JPSI = 1
               IF(IST .EQ. 6) JPSI = 1
               GO TO 3120
2110        CONTINUE
               JPSI = NOPP
3120        CONTINUE
            RETURN
         END
      SUBROUTINE IV3
     + (CHARLN ,CI     ,COHES  ,CPFFG  ,DIAW   ,DOWPB  ,DRAT   ,GAMMA
     + ,GCA    ,IB     ,ID     ,IP     ,IST    ,JPSI   ,LUN1   ,NAMBLY
     + ,NPAD   ,NSLIP  ,NVEH   ,NWHL   ,RCI    ,RTOWPB ,RTOWT  ,SECTW
     + ,TANPHI ,TRAKLN ,TRAKWD ,VCIFG  ,VCIMUK ,WGHT   ,WRATIO ,ZSNOW )
C
C     ----------------------------------
C     PULL AND RESISTANCE COEFFICIENTS
C     ----------------------------------
C
C     1. VARIABLE DECLARATION
         REAL      B
         REAL      CHARLN (20,3)
         REAL      CI
         REAL      COHES
         REAL      CPFFG  (20,3)
         REAL      D
         REAL      DIAW   (20)
         REAL      DOWCO
         REAL      DOWCS
         REAL      DOWPB  (20)
         REAL      DOWS
         REAL      DRAT   (20,3)
         REAL      G
```

```
      REAL       GAMMA
      REAL       GCA      (20,3)
      INTEGER    I
      INTEGER    IB       (20)
      INTEGER    ID       (20)
      INTEGER    IP       (20)
      INTEGER    IST
      INTEGER    JPSI
      INTEGER    KIV3
      INTEGER    LUN1
      INTEGER    NAMBLY
      INTEGER    NPAD     (20)
      INTEGER    NSLIP
      INTEGER    NVEH     (20)
      INTEGER    NWHL     (20)
      REAL       PID
      REAL       PIT
      REAL       RCI
      REAL       RCIO
      REAL       RCIS
      REAL       RCIX
      REAL       RT
      REAL       RTOWPB   (20)
      REAL       RTOWT    (20)
      REAL       SECTW    (20)
      REAL       TANPHI
      REAL       TOWMAX
      REAL       TRAKLN   (20)
      REAL       TRAKWD   (20)
      REAL       VCIFG    (20,3)
      REAL       VCIMUK   (20)
      REAL       W
      REAL       WGHT     (20)
      REAL       WRATIO
      REAL       XK
      REAL       XKDELT
      REAL       XN
      REAL       XNVEH
      REAL       ZSNOW
C     3. DETERMINE SOIL TYPE
      IF( IST .EQ. 1) GO TO 3100
      IF( IST .EQ. 2) GO TO 3600
      IF( IST .EQ. 3) GO TO 3700
      IF( IST .EQ. 4) GO TO 3800
      IF( IST .EQ. 6) GO TO 3100
      STOP 6
C     4. FINE GRAINED SOIL PULL AND RESISTANCE COEFFICIENTS
3100  CONTINUE
      DO 3414 I=1,NAMBLY,1
         RCIX=RCI-VCIFG(I,JPSI)
         IF((IP(I) .EQ. 1) .OR. (IB(I) .EQ. 1)) GO TO 3132
            RTOWPB(I) = 0.0
```

```
              DOWPB (I) = 0.0
              CALL FGSTR (
     +            DIAW    ( I )
     +           ,DRAT    ( I , JPSI )
     +           ,IB      ( I )
     +           ,IP      ( I )
     +           ,NVEH    ( I )
     +           ,NWHL    ( I )
     +           ,RCI
     +           ,RTOWT   ( I )
     +           ,SECTW   ( I )
     +           ,WGHT    ( I )
     +           ,WRATIO
     +           ,LUN1 )
              GO TO 3414
3132       CONTINUE
           IF(NSLIP .NE. 0) GO TO 3134
              CALL FGSPC (
     +            CPFFG   ( I , JPSI )
     +           ,D
     +           ,NVEH    ( I )
     +           ,RCIX )
              DOWPB(I)=C
              CALL FGSPR (
     +            CPFFG   ( I , JPSI )
     +           ,NVEH    ( I )
     +           ,RCIX
     +           ,RTOWPE ( I )     )
              CALL FGSTR (
     +            DIAW    ( I )
     +           ,DRAT    ( I , JPSI )
     +           ,IB      ( I )
     +           ,IP      ( I )
     +           ,NVEH    ( I )
     +           ,NWHL    ( I )
     +           ,RCI
     +           ,RTOWT   ( I )
     +           ,SECTW   ( I )
     +           ,WGHT    ( I )
     +           ,WRATIC
     +           ,LUN1 )
              GO TO 3414
3134       CONTINUE
           IF(NVEH(I) .EQ. 1) GO TO 3306
C             A. TRACKED SLIPPERY ROUTINE
                 RCIO   =  20.0
                 DOWCO  =  0.55
                 IF( RCIX .GT. 20.0 ) GO TO 3168
                    CALL FGSPC (
     +               CPFFG  ( I , JPSI )
     +              ,D
     +              ,NVEH   ( I )
```

```
     +                    ,RCIX )
                          DOWPB( I)=D
                          CALL FGSPR (
     +                     CPFFG  ( I , JPSI )
     +                    ,NVEH   ( I )
     +                    ,RCIX
     +                    ,RTGWPE ( I )    )
                          CALL FGSTR (
     +                     DIAW   ( I )
     +                    ,DRAT   ( I , JPSI )
     +                    ,IB     ( I )
     +                    ,IP     ( I )
     +                    ,NVEH   ( I )
     +                    ,NWHL   ( I )
     +                    ,RCI
     +                    ,RTOWT  ( I )
     +                    ,SECTW  ( I )
     +                    ,WGHT   ( I )
     +                    ,WRATIC
     +                    ,LUN1)
                          GO TO 3614
 3168                   CONTINUE
                        IF(IST .NE. 6) GC TO 3224
 3174                     IF(NSLIR .NE. 1) GO TO 3182
                             DOWCS  =   0.5
                             RCIS   = 200.0
                             GO TO 3252
 3182                     IF(NSLIR .NE. 2) GO TO 3190
                             COWCS  =   0.3
                             RCIS   = 150.0
                             GO TO 3252
 3190                     IF(NSLIR .NE. 3) GO TO 3198
                             COWCS  =   0.3
                             RCIS   = 200.0
                             GO TO 3252
 3198                     IF(NSLIR .NE. 4) GO TO 3206
                             COWCS  =   0.1
                             RCIS   = 200.0
                             GO TO 3252
 3206                     IF(NSLIR .NE. 5) GO TO 3214
                             COWCS  =   0.1
                             RCIS   = 300.0
                             GO TO 3252
 3214                     IF(NSLIR .NE. 6) GC TO 3222
                             COWCS  =   0.15
                             RCIS   = 500.0
                             GO TO 3252
 3222                     STOP 7
 3224                   CONTINUE
                        IF(IST .NE. 1) GC TO 3251
                          RCIS = 100.0
                          IF(NSLIR .NE. 1) GC TO 3230
```

```
                    DOWCS = 0.45
                    GO TO 3252
3230             IF( NSLIE .NE. 2) GO TO 3234
                    DOWCS = 0.3
                    GO TO 3252
3234             IF( NSLIE .NE. 3) GO TO 3238
                    DOWCS = 0.2
                    GO TO 3252
3238             IF( NSLIE .NE. 4) GO TO 3242
                    DOWCS = 0.1
                    GO TO 3252
3242             IF( NSLIE .NE. 5) GO TO 3246
                    DOWCS = 0.1
                    GO TO 3252
3246             IF( NSLIE .NE. 6) GO TO 3250
                    DOWCS = 0.15
                    GO TO 3252
3250             STOP
3251          CONTINUE
                 STOP 10
3252          IF( RCIX .GE. RCIS ) GO TO 3282
                 XN=ALOG10( DOWCO/DOWCS )/ALOG10( RCIS/RCIO )
                 XK=DOWS*RCIS**XN
                 DOWS=XK*((1./RCIX)**XN)
                 IF( NPAB(I) .EQ. 0 ) GO TO 3274
                    DOWPB(I)=DOWS
                    CALL FGSPR (
     +               CPFFG   ( I , JPSI )
     +              ,NVEH    ( I )
     +              ,RCIX
     +              ,RTCMPB  ( I )     )
                    CALL FGSTR (
     +               DIAW    ( I )
     +              ,DRAT    ( I , JPSI )
     +              ,IB      ( I )
     +              ,IP      ( I )
     +              ,NVEH    ( I )
     +              ,NWHL    ( I )
     +              ,RCI
     +              ,RTCWT   ( I )
     +              ,SECTW   ( I )
     +              ,WGHT    ( I )
     +           ,WRATIC
     +           ,LUN1)
                    GO TO 3414
3274             CONTINUE
                 CALL FGSPC (
     +            CPFFG  ( I , JPSI )
     +           ,D
     +           ,NVEH   ( I )
     +           ,RCIX )
                 DOWPB(I)=0.5*( D+DOWS )
```

```
                     GO TO 3414
3282             IF(NPAD(I) .EQ. Ø) GO TO 3296
                     DOWPB(I)=DOWCS
                     CALL FGSPR (
     +                CPFFG  ( I , JPSI )
     +               ,NVEH   ( I )
     +               ,RCIX
     +               ,RTCWPB ( I )    )
                     CALL FGSTR (
     +                DIAW   ( I )
     +               ,DRAT   ( I , JPSI )
     +               ,IB     ( I )
     +               ,IP     ( I )
     +               ,NVEH   ( I )
     +               ,NWHL   ( I )
     +               ,RCI
     +               ,RTCWT  ( I )
     +               ,SECTW  ( I )
     +               ,WGHT   ( I )
     +               ,WRATIC
     +               ,LUN1)
                     GO TO 3414
3296             CONTINUE
                 CALL FGSPC (
     +            CPFFG  ( I , JPSI )
     +           ,D
     +           ,NVEH   ( I )
     +           ,RCIX )
                 DOWPB(I)=Ø.5*( D+DOWS )
                 CALL FGSPF (
     +            CPFFG  ( I , JPSI )
     +           ,NVEH   ( I )
     +           ,RCIX
     +           ,RTCWPB ( I )    )
                 CALL FGSTF (
     +            DIAW   ( I )
     +           ,DRAT   ( I , JPSI )
     +           ,IB     ( I )
     +           ,IP     ( I )
     +           ,NVEH   ( I )
     +           ,NWHL   ( I )
     +           ,RCI
     +           ,RTCWT  ( I )
     +           ,SECTW  ( I )
     +           ,WGHT   ( I )
     +              ,WRATIC
     +              ,LUN1)
                 GO TO 3414
3306         CONTINUE
C            B. WHEELED SLIPPERY ROUTINE
                XKDELT=( DRAT(I,JPSI)/Ø.4 )-Ø.375
                RCIO=18.Ø
```

```
                DOWCO=0.4
                IF( RCIX .GT. 20.0 ) GO TO 3324
                   RCIX=RCIX-2.0
                   CALL FGSPC (
     +                 CPFFG  ( I , JPSI )
     +                ,D
     +                ,NVEH   ( I )
     +                ,RCIX )
                   DOWPB(I)=D
                   CALL FGSPF (
     +                 CPFFG  ( I , JPSI )
     +             ,NVEH   ( I )
     +                ,RCIX
     +             ,RTCWPB ( I )     )
                   CALL FGSTF (
     +                 DIAW   ( I )
     +                ,DRAT   ( I ,JPSI )
     +                ,IB     ( I )
     +                ,IP     ( I )
     +                ,NVEH   ( I )
     +                ,NWHL   ( I )
     +                ,RCI
     +                ,RTCWT  ( I )
     +                ,SECTW  ( I )
     +                ,WGHT   ( I )
     +                ,WRATIC
     +                ,LUN1)
                   GO TO 3414
3324            CONTINUE
                IF(IST .NE. 6) GO TO 3378
                   IF(NSLIP .NE. 1) GO TO 3338
                      DOWCS  =   0.35
                      RCIS   = 300.0
                      GO TO 3394
3338               IF(NSLIP .NE. 2) GO TO 3346
                      DOWCS  =   0.25+XKDELT
                      RCIS   = 150.0
                      GO TO 3394
3346               IF(NSLIP .NE. 3) GO TO 3354
                      DOWCS  =   0.2+XKDELT
                      RCIS   = 200.0
                      GO TO 3394
3354               IF(NSLIP .NE. 4) GO TO 3362
                      DOWCS  =   0.15+XKDELT
                      RCIS   = 150.0
                      GO TO 3394
3362               IF(NSLIP .NE. 5) GO TO 3370
                      DOWCS  =   0.15+XKDELT
                      RCIS   = 150.0
                      GO TO 3394
3370               IF(NSLIP .NE. 6) GO TO 3376
                      DOWCS  =   0.15
```

```
                    RCIS    = 100.0
                    GO TO 3394
3376             STOP 11
3378          CONTINUE
              RCIS=80.0
              IF(NSLIP .NE. 1) GO TO 3386
                 DOWCS=0.3
                 GO TO 3394
3386          IF(NSLIP .NE. 4) GO TO 3392
                 DOWCS=0.1+XKDELT
                 GO TO 3394
3392          CONTINUE
                 DOWCS = 0.1
3394          CONTINUE
              IF(RCIX .GE. RCIS) GO TO 3408
                 XN=ALOG10( DOWCO/DOWCS )/ALOG10( RCIS/RCIO )
                 XK=DOWCS*RCIS**XN
                 DOWPB(I)=XK/( RCIX**XN )
                 CALL FGSPR (
     +                CPFFG  ( I ,JPSI )
     +               ,NVEH   ( I )
     +               ,RCIX
     +               ,RTOWPB ( I )    )
                 CALL FGSTR (
     +                DIAW   ( I )
     +               ,DRAT   ( I , JPSI )
     +               ,IB     ( I )
     +               ,IP     ( I )
     +               ,NVEH   ( I )
     +               ,NWHL   ( I )
     +               ,RCI
     +               ,RTCWT  ( I )
     +               ,SECTW  ( I )
     +               ,WGHT   ( I )
     +               ,WRATIC
     +               ,LUN1)
                 GO TO 3414
3408          CONTINUE
              DOWPB(I) = DOWCS
              CALL FGSPR (
     +             CPFFG  ( I , JPSI )
     +            ,NVEH   ( I )
     +            ,RCIX
     +            ,RTOWPB ( I )    )
              CALL FGSTR (
     +             DIAW   ( I )
     +            ,DRAT   ( I , JPSI )
     +            ,IB     ( I )
     +            ,IP     ( I )
     +            ,NVEH   ( I )
     +            ,NWHL   ( I )
     +            ,RCI
```

```
     +              ,RTOWT   ( I )
     +              ,SECTW   ( I )
     +              ,WGHT    ( I )
     +                ,WRATIC
     +                ,LUN1)
3414       CONTINUE
           GO TO 3999
C       5. COARSE GRAINED SOIL PULL AND RESISTANCE COEFFICIENTS
3600       CONTINUE
           G = CI*.8645/3.
           DO 3630 I=1,NAMBLY,1
              IF(NVEH(I) .EQ. 0) GO TO 3620
C                A. WHEELED ELEMENT ALGORITHM
                    IF((IP(I) .EQ. 0) .OR. (IB(I) .EQ. 0)) GO TO 3610
                      RTOWT(I) = 0.
                      GO TO 3612
3610                CONTINUE
                    W    = WGHT(I)/FLOAT(NWHL(I))
                    PIT = G*((SECTW(I)*DIAW(I))**1.5)*(FLOAT(I)**(1./2.))
     + /( W*((1.0-DRAT(I,JPSI))**3.0)*(1.0*SECTW(I)/DIAW(I)) )
                    RTOWT(I) = 0.44-0.01*PIT
     + +SQRT(  ((0.44-0.01*PIT)**2)+0.0002*PIT+0.08   )
3612                CONTINUE
                    IF((IP(I) .EQ. 0) .AND. (IB(I) .EQ. 0)) GO TO 3618
                       IF( ID(I) .EQ. 1) GO TO 3614
                          B = SECTW(I)
                          W = WGHT(I)/FLOAT(NWHL(I))
                          GO TO 3616
3614                   CONTINUE
                       B = 2.0*SECTW(I)
                       W = 2.0*WGHT(I)/FLOAT(NWHL(I))
3616                   CGNTINUE
                       PIC = G*((B*DIAW(I))**1.5)
                       PIC = G*((B*DIAW(I))**1.5)
     + *(DRAT(I,JPSI))/(W*(FLOAT(I)**0.5))
                       DOWPB (I) = .53-4.5/(PIC+3.7)
                       RTOWPB(I) = .6-DOWPB(I)
                       GO TO 3630
3618                CONTINUE
                    DOWPB (I) = 0.0
                    RTOWPB(I) = 0.0
                    GO TO 3630
C             B. TRACKED ELEMENT ALGORITHM
3620             CONTINUE
                 RTOWT(I) = 0.0
                 IF((IP(I) .EQ. 1) .AND. (IB(I) .EQ. 1)) GO TO 3621
         WRITE(LUN1,1000)
1000                FORMAT(
     + /,37H THERE IS NO SOIL PULL AND RESISTANCE,
     + /,37H ALGORITHM FOR TOWED TRACKED ELEMENTS)
                    STOP
3621             CONTINUE
```

```
                PIT = 0.6*G*( (TRAKWD(I)*TRAKLN(I))**1.5 )
     + /WGHT(I)/2.
                IF(PIT .GT. 25.) GO TO 3622
                   DOWPB(I) = .121+.258*ALOG10(PIT)
                   GO TO 3628
 3622              IF(PIT .GT. 100.) GO TO 3624
                   DOWPB(I) = .339+.109*ALOG10(PIT)
                   GO TO 3628
 3624              IF(PIT .GT. 1000.) GO TO 3626
                   DOWPB(I) = .481+.038*ALOG10(PIT)
                   GO TO 3628
 3626           CONTINUE
                   DOWPB (I) = .595
 3628           CONTINUE
                   RTOWPB(I) = .6 - DOWPB(I) + .045
 3630      CONTINUE
           GO TO 3999
C      6. MUSKEG PULL AND RESISTANCE COEFFICIENTS
 3700      CONTINUE
           DO 3728 I=1,NAMBLY,1
              RCIX = RCI-VCIMUK(I)
              IF(RCIX .GT. -100.) GO TO 3710
                 RT = 1.
                 GO TO 3714
 3710         IF(RCIX .GT. 0.) GO TO 3712
                 RT = 1.-.006*(RCIX+100.)
                 GO TO 3714
 3712         CONTINUE
                 RT = .045+2.3075/(6.5+RCIX)
 3714         IF((IP(I) .EC. 0) .OR. (IB(I) .EQ. 0)) GO TO 3716
                 RTOWT(I)  = 0.
                 RTOWPB(I) = RT
                 GO TO 3720
 3716         IF((IP(I) .EC. 1) .OR. (IB(I) .EQ. 1)) GO TO 3718
                 RTOWT(I)  = RT
                 RTOWPB(I) = 0.
                 DOWPB(I)  = 0.0
                 GO TO 3720
 3718         CONTINUE
                 RTOWT (I) = RT
                 RTOWPB(I) = RT
 3720         IF(RCIX .GT. -100.) GO TO 3722
                 DOWPB(I) = -1.0
                 GO TO 3728
 3722         IF(RCIX .GT. 0.0) GO TO 3724
                 DOWPB(I) = -1.+.01*(RCIX+100.)
                 GO TO 3728
 3724         IF
     + ((NVEH(I) .EQ. 0) .AND. (CPFFG(I,JPSI) .LT. 4.0)) GO TO 3726
              DOWPB(I)=0.3537+0.02258*RCIX
     + -((((.3537+.02258*RCIX)**2.)-(.03071*RCIX))**.5)
              GO TO 3728
```

```
3726           COWPB(I)=0.5464+0.1091*RCIX
     + -((((.5464+.1091*RCIX)**2.)-(.192*RCIX))**.5)
3728        CONTINUE
            GO TO 3999
C        7. SHALLOW SNOW PULL AND RESISTANCE COEFFICIENTS
3800        CONTINUE
            XNVEH = 0.0
            DO 3805 I=1,NAMBLY
               XNVEH = FLOAT( NVEH(I) )+XNVEH
3805        CONTINUE
            DO 3820 I=1,NAMBLY,1
               IF(NVEH(I) .EQ. 0) GO TO 3814
                  RT = 10.*(FLOAT(NWHL(I))*SECTW(I)/DIAW(I))
     + *(GAMMA*ZSNOW/CHARLN(I,JPSI))/XNVEH
                  IF((IP(I) .EQ. 0) .OR. (IB(I) .EQ. 0)) GO TO 3810
                     RTOWT(I)  = 0.
                     RTOWPB(I) = RT
                     GO TO 3813
3810              IF((IP(I) .EQ. 1) .OR. (IB(I) .EQ. 1)) GO TO 3812
                     RTOWT(I)  = RT
                     RTOWPB(I) = 0.
                     DOWPB(I)  = 0.
                     GO TO 3820
3812              CONTINUE
                     RTOWT (I) = RT
                     RTOWPB(I) = RT
3813              CONTINUE
                     TOWMAX    = TANPHI+COHES*GCA(I,JPSI)
     +                          *FLOAT(NWHL(I))/WGHT(I)
                     DOWPB(I)  = TOWMAX-RT
                     GO TO 3820
3814           CONTINUE
               RT = 5.0*GAMMA*(ZSNOW/CHARLN(I,JPSI)-0.15)
               IF(RT .GE. 0.0) GO TO 3815
                  RT = 0.0
3815           IF((IP(I) .EQ. 0) .OR. (IB(I) .EQ. 0)) GO TO 3816
                  RTOWT (I)  = 0.
                  RTOWPB(I)  = RT
                  GO TO 3819
3816           IF((IP(I) .EQ. 1) .OR. (IB(I) .EQ. 1)) GO TO 3818
                  RTOWT (I)  = RT
                  RTOWPB(I)  = 0.
                  DOWPB (I)  = 0.
                  GO TO 3820
3818           CONTINUE
                  RTOWT (I)  = RT
                  RTOWPB(I)  = RT
3819           CONTINUE
               TOWMAX    = TANPHI+COHES*GCA(I,JPSI)/WGHT(I)
               DOWPB(I)  = TOWMAX-RT
3820        CONTINUE
3999        CONTINUE
```

```
      RETURN
      END
      SUBROUTINE IV4
     + (DOWB    ,DOWP    ,DOWPB   ,GCW     ,GCWB    ,GCWP    ,IB      ,IP
     + ,NAMBLY ,RTOWB   ,RTOWNB  ,RTOWNP  ,RTOWP   ,RTOWPB  ,RTOWT   ,WGHT )
C
C     -----------------------------------------
C     SUMMED PULL AND RESISTANCE COEFFICIENTS
C     -----------------------------------------
C
C     1. VARIABLE DECLARATION
         REAL        DOWPB  (20)
         INTEGER     IB     (20)
         INTEGER     IP     (20)
         REAL        RTOWPB (20)
         REAL        RTOWT  (20)
         REAL        WGHT   (20)
C     3. ALGORITHM
         DOWB    = 0.0
         DOWP    = 0.0
         RTOWB   = 0.0
         RTOWP   = 0.0
         RTOWNB  = 0.0
         RTOWNP  = 0.0
         DO 3050 I=1,NAMBLY
            IF(GCWP .EQ. 0.0) GO TO 3010
               RTOWP   = RTOWP+FLOAT(IP(I))*RTOWPB(I)*WGHT(I)/GCWP
               DOWP    = DOWP+FLOAT(IP(I))*DOWPB(I)*WGHT(I)/GCWP
3010        IF((GCW-GCWP) .EQ. 0.0) GO TO 3020
               RTOWNP  = RTOWNP+FLOAT(1-IP(I))
     + *RTOWT(I)*WGHT(I)/(GCW-GCWP)
3020        IF(GCWB .EQ. 0.0) GO TO 3030
               RTOWB   = RTOWB+FLOAT(IB(I))*RTOWPB(I)*WGHT(I)/GCWB
               DOWB    = DOWB+FLOAT(IB(I))*DOWPB(I)*WGHT(I)/GCWB
3030        IF((GCW-GCWB) .EQ. 0.0) GO TO 3040
               RTOWNB  = RTOWNB+FLOAT(1-IB(I))
     + *RTOWT(I)*WGHT(I)/(GCW-GCWP)
3040        CONTINUE
3050     CONTINUE
         RETURN
         END
      SUBROUTINE IV5
     + (ATF     ,AVGC    ,BTF     ,CD      ,CPFCCG  ,CPFCFG  ,CTF     ,DAREA
     + ,DOWP    ,EANG    ,ECF     ,FA      ,FB      ,FC      ,FORMX   ,GCWR
     + ,IFLOAT  ,IST     ,ITUT    ,JPSI    ,LOCDIF  ,NAMBLY  ,NFL     ,NGR
     + ,NTRAV   ,NVEH    ,NVEHC   ,NWHL    ,RACC    ,RTOWP   ,STRACT  ,TFOR
     + ,THETA   ,TRACTF  ,VFMAX   ,VG      ,VGV     ,WGHT    ,WRATIO  ,LUNI )
C
C     ----------------------------
C     SLIP MODIFIED TRACTIVE EFFORT
C     ----------------------------
C
```

```
C       1. VARIABLE DECLARATION
           REAL         ATF      (20)
           REAL         BTF      (20)
           REAL         CPFCCG   (3)
           REAL         CPFCFG   (3)
           REAL         CTF      (20)
           REAL         FA       (20,3)
           REAL         FATEMP   (20,3)
           REAL         FB       (20,3)
           REAL         FBTEMP   (20,3)
           REAL         FC       (20,3)
           REAL         FCTEMP   (20,3)
           REAL         FORMX    (3)
           REAL         FQUAD5   (5)
           REAL         FTEMP    (20,5)
           REAL         FTEMPC   (20,5)
           INTEGER      NVEH     (20)
           INTEGER      NWHL     (20)
           REAL         STRACT   (20,3,3)
           REAL         THETA    (3)
           REAL         TRACTF   (20,5)
           REAL         VFMAX    (3)
           REAL         VG       (20,3,3)
           REAL         VGTEMP   (20,3,3)
           REAL         VGV      (20,5)
           REAL         VQUAD5   (5)
           REAL         VTEMP    (20,5)
           REAL         VTEMPC   (20,5)
           REAL         WGHT     (20)
C       3. ALGORITHM
        FCC  = 0.
        CPFC = CPFCFG(JPSI)
        IF (IST .EQ. 2) CPFC=CPFCCG(JPSI)
        IF (IST .EQ. 4) GO TO 3000
           CALL TFORCF(CF,CPFC,DCWP,GCWP,IST,NFL,NVEHC,RTOWP,TFOR,LUN1)
 3000   DO 3600    K=1,NTRAV
           COSX = COS(THETA(K))
           IF (IST .EQ. 4)TFOR=(DCWP+RTOWP)*GCWP*WRATIO
           NG1 = 1
              DO 3060    NG=1,NGR
                 FATEMP(NG,K) = ATF(NG)
                 FBTEMP(NG,K) = BTF(NG)
                 FCTEMP(NG,K) = CTF(NG)
                 DO 3050    L=1,5
                    VTEMP(NG,L) = VGV(NG,L)
                    FTEMP(NG,L) = TRACTF(NG,L)
 3050            CONTINUE
                 L=1
                 DO 3060 L1=1,5,2
                    VGTEMP(NG,L,K) = VGV(NG,L1)
                    L=L+1
 3060         CONTINUE
```

```
              DO 3500   NG=1,NGR
          IF(FTEMP(NG,1)*ECF .LT. TFOR*COSX) GO TO 3300
              IF(FTEMP(NG,5)*ECF .LT. TFOR*COSX) GO TO 3200
C
C                   100% AT TOP SPEED IN GEAR
C
                    NG1 = NG1+1
                    FC(NG,K)=0.
                    FB(NG,K) = 0.
                    FA(NG,K)=TFOR*COSX
              DO 3100 L=1,3
                      VG(NG,L,K) = 0.
                      STRACT(NG,L,K)=FA(NG,K)
 3100               CONTINUE
                    GO TO 3500
C
C                 100% SLIP AT OTHER THAN TOP SPEED
C
 3200             IF (VTEMP(NG,5) .EQ. VTEMP(NG,1)) GO TO 3260
                    FX = TFOR*COSX
                    CALL VELFOR(FX,FATEMP,FBTEMP,FCTEMP,
     +                     FX,K,NGR,VX,VFMAX,VGTEMP)
C
C                   RESET LOW POINT TO LOWEST SPEED THAT IS NOT
C                   100% SLIP
C
                    VTEMP(NG,1) = VX
                    FTEMP(NG,1) = FX/ECF
                    XINT = (VTEMP(NG,5) - VTEMP(NG,1))/4.
                    DO 3250   L=2,4
                       VTEMP(NG,L) = VTEMP(NG,L-1) + XINT
                       FTEMP(NG,L) = CTF(NG)*VTEMP(NG,L)*VTEMP(NG,L) +
     +                               BTF(NG)*VTEMP(NG,L) + ATF(NG)
 3250               CONTINUE
                    GO TO 3300
C
C                SPEEDS EQUAL - INCREMENT ON FORCE
C
 3260            FTEMP(NG,1) = TFOR*COSX
                 YINT         = (FTEMP(NG,1) - FTEMP(NG,5))/4.
                  DO 3380   L=2,4
                     FTEMP(NG,L) = FTEMP(NG,L-1) - YINT
 3380             CONTINUE
C
C              COMPUTE SLIP FOR ALL POINTS IN GEAR
C
 3300          DO 3400   L=1,5
                  VX = VTEMP(NG,L)
                  FX = FTEMP(NG,L)*ECF
                  FSL = FX/COSX*WRATIO
                  YX = FSL/GCWP - CF
                  IF (IST .EQ. 4)YX=FX/TFOR
```

```
                    CALL SLIP(CPFC,IST,LOCDIF,NFL,NVEHC,SLIPX,YX)
                    VTEMPO(NG,L) = VX*(1.-SLIPX)
                    WDRAG = .5*.00111*CD*DAREA*VTEMPO(NG,L)**2
                   IF(ITUT .NE. 14) GC TO 3390
C
C                  DRAG CORNERING RESISTANCE FOR TRAILS (WHEELED)
C
                       FCC = 0.
                       FE  = 1. - 7.495*(RACC/12.)*EANG
                       DO 3350 I=1,NAMBLY
                          IF(NVEH(I) .LT. 1) GO TO 3350
                             F1  = ((WGHT(I)*COSX*VTEMPO(NG,L)**2)/
     +                             (111.1*RADC))*(12./17.6/17.6)
                             FCC = ((FE*F1*F1)/(FLOAT(NWHL(I))*AVGC))*
     +                             .75/(TFOR/GCWP) + FCC
3350                   CONTINUE
3390               FTEMPO(NG,L) = FX - FLOAT(IFLOAT)*WDRAG - FCC
 3400           CONTINUE
C
C               COMPUTE NEW COEFFICIENTS
C
                IF((NG .NE. 1) .OR.
     +             (FTEMP(NG,1) .NE. FTEMP(NG,5))) GO TO 3410
                   FA(NG,K) = ATF(NG)
                   FB(NG,K) = BTF(NG)
                   FC(NG,K) = CTF(NG)
                   GO TO 3430
3410            DO 3420   L=1,5
                   FQUAD5(L) = FTEMPC(NG,L)
                   VQUAD5(L) = VTEMPC(NG,L)
 3420           CONTINUE
                CALL QUAD5(VQUAD5,FQUAD5,A,B,C)
                FA(NG,K) = A
                FB(NG,K) = B
                FC(NG,K) = C
 3430           L = 1
                DO 3450 L1=1,5,2
                   VG(NG,L,K) = VTEMPO(NG,L1)
                   STRACT(NG,L,K) = FC(NG,K)*VG(NG,L,K)*VG(NG,L,K) +
     +                              FB(NG,K)*VG(NG,L,K) + FA(NG,K)
                   L = L+1
 3450           CONTINUE
 3500        CONTINUE
             VFMAX(K) = VG(NG1,1,K)
             FORMX(K) = STRACT(NG1,1,K)
 3600 CONTINUE
             RETURN
          END
       SUBROUTINE IV6
     + (FAT     ,FAT1    ,FMT     ,NI      ,PBHT    ,SD      ,SDL     ,TDEN
     + ,WDTH )
C
```

```
C        -------------------------------
C        RESISTANCE DUE TO VEGETATION
C        -------------------------------
C
C        1. VARIABLE DECLARATION
C
            REAL        FAT     (9)
            REAL        FAT1    (9)
            REAL        FMT     (9)
            REAL        SD      (9)
            REAL        SDL     (9)
            REAL        TDEN    (9)
            REAL        TFAT    (9)
C        3. ALGORITHM
            FAT (1) = 0.0
            FAT1(1) = 0.0
            FMT (1) = 0.0
            TFAT(1) = 0.0
            DO 3030 I=1,NI
               IF(TDEN(I) .EQ. 0.0) GO TO 3020
                  FAT1(I+1) = (56./5.8)*SDL(I)**3.
                  FMT (I+1) = (40.-PBHT/2.)*SDL(I)**3.
                  TFAT(I+1) = 0.0
                  DO 3010 K=1,I
                     TFAT(I+1) = TFAT(I+1)+TDEN(K)*100.0*SD(K)**3
3010              CONTINUE
                  FAT (I+1) = 12.*TFAT(I+1)*WDTH
                  GO TO 3030
3020           CONTINUE
               FAT1(I+1) = 0.0
               FMT (I+1) = 0.0
               FAT (I+1) = FAT(1)
3030        CONTINUE
            RETURN
            END
         SUBROUTINE IV7
       + (FMT     ,GCW     ,IMPACT ,MAXI    ,NI      ,PBF )
C
C        -------------------------------------------------
C        DRIVER-DEPENDENT VEHICLE VEGETATION OVERRIDE CHECK
C        -------------------------------------------------
C
C        1. VARIABLE DECLARATION
            REAL        FMT     (9)
            REAL        IMPACT  (9)
C        3. ALGORITHM
            NI1 = NI+1
            DO 3040 I=1,NI1
               IMPACT(I) = 0
               IF(FMT(I) .LE. PBF) GO TO 3010
                  IMPACT(I) = 1
3010           IF((FMT(I)/GCW) .LE. 2.) GO TO 3020
```

```
               IMPACT(I) = IMPACT(I)+2.
 3020          IF(IMPACT(I) .NE. 0.) GO TO 3030
               MAXI = I
3030        CONTINUE
3040     CONTINUE
         RETURN
         END
      SUBROUTINE IV8
     + (FAT     ,FAT1    ,GCW     ,GCWNP   ,GCWP    ,MAXI    ,NTRAV   ,RTOWNP
     + ,RTOWP   ,STR     ,THETA   ,TRESIS  ,WRATIO )
C
C     ----------------------------------
C     TOTAL RESISTANCE BETWEEN OBSTACLES
C     ----------------------------------
C
C     1. VARIABLE DECLARATION
         REAL        FAT     (9)
         REAL        FAT1    (9)
         REAL        STR     (3,9)
         REAL        THETA   (3)
         REAL        TRESIS  (3,9)
C     3. ALGORITHM
         DO 3020 K=1,NTRAV
            DUMMY = GCW*SIN(THETA(K))
     + +(RTOWP*GCWP+RTOWNP*GCWNP)*COS(THETA(K))*WRATIO
            DO 3010 I=1,MAXI
               TRESIS(K,I) = DUMMY*FAT  (I)
               STR   (K,I) = DUMMY*FAT1(I)
3010        CONTINUE
3020     CONTINUE
         RETURN
         END
      SUBROUTINE IV9
     + (FA      ,FB      ,FC      ,FORMX   ,MAXI    ,NGR     ,NI      ,NTRAV
     + ,TRESIS  ,VFMAX   ,VG      ,VSOIL )
C
C     ---------------------------------------------
C     SPEED LIMITED BY RESISTANCE BETWEEN OBSTACLES
C     ---------------------------------------------
C
C     1. VARIABLE DECLARATION
         REAL        FA      (20,3)
         REAL        FB      (20,3)
         REAL        FC      (20,3)
         REAL        FORMX   (3)
         REAL        TRESIS  (3,9)
         REAL        VFMAX   (3)
         REAL        VG      (20,3,3)
         REAL        VSOIL   (3,9)
C     3. ALGORITHM
         M2 = MAXI
         DO 3030 I=1,M2
```

```
            CALL VELFOR
     + (TRESIS(1,I),FA,FB,FC,FCRMX,1,NGR,VSOIL(1,I),VFMAX,VG)
            IF(VSOIL(1,I) .NE. 0.0) GO TO 3010
               MAXI = I
3010        IF(NTRAV .EQ. 1) GO TO 3020
               CALL VELFOR
     + (TRESIS(2,I),FA,FB,FC,FCRMX,2,NGR,VSOIL(2,I),VFMAX,VG)
               CALL VELFOR
     + (TRESIS(3,I),FA,FB,FC,FCRMX,3,NGR,VSOIL(3,I),VFMAX,VG)
3020        CONTINUE
3030     CONTINUE
         M1 = MAXI+1
         M2 = NI  +1
         DO 3050 I=M1,M2
            VSOIL(1,I) = 0.0
            IF(NTRAV .EQ. 1) GO TO 3040
               VSOIL(2,I) = 0.0
               VSOIL(3,I) = 0.0
3040        CONTINUE
3050     CONTINUE
         RETURN
         END
      SUBROUTINE IV10
     + (ACTRMS ,LAC     ,MAXIPR ,RMS     ,VRID    ,VRIDE )
C
C     -----------------------------------
C     SPEED LIMITED BY SURFACE ROUGHNESS
C     -----------------------------------
C
C     1. VARIABLE DECLARATION
         REAL        RMS      (20)
         REAL        VRIDE    (20,3)
C     3. ALGORITHM
         DO 3020 NR=2,MAXIPR
            IF(ACTRMS .GE. RMS(NR)) GO TO 3010
               VRID = VRIDE(NR-1,LAC)+(ACTRMS-RMS(NR-1))
     + *(VRIDE(NR,LAC)-VRIDE(NR-1,LAC))/(RMS(NR)-RMS(NR-1))
               GO TO 3030
3010        CONTINUE
3020     CONTINUE
         VRID = VRIDE(MAXIPR,LAC)
3030     CONTINUE
         RETURN
         END
      SUBROUTINE IV11
     + (DOWB    ,GCW     ,GCWB    ,GCWNB   ,NOGOBF ,NTRAV   ,RTOWB   ,RTOWNB
     + ,TBF     ,THETA   ,WRATIO ,XBR )
C
C     ------------------------------------------
C     TOTAL BRAKING FORCE - SOIL/SLOPE/VEHICLE
C     ------------------------------------------
C
```

```
C       1. VARIABLE DECLARATION
           REAL        THETA   (3)
           REAL        TBF     (3)
C       3. ALGORITHM
           X1 = (RTOWB*GCWB+RTGWNB*GCWNB)*WRATIO
           X2 = (DCWB+RTOWB)*GCWB*WRATIO
           TBF(1) = GCW*SIN(THETA(1))*WRATIO+X1*COS(THETA(1))
     + +AMIN1(XBR,X2*COS(THETA(1)))
           IF(NTRAV .NE. 1) GO TO 3020
              IF(TBF(1) .GE. 0.0) GO TO 3010
                 NOGOBF = 0
                 GO TO 4000
3010          CONTINUE
              NOGOBF = 1
              GO TO 4000
3020       CONTINUE
           TBF(2) = X1+AMIN1(XBR,X2)
           TBF(3) = GCW*SIN(THETA(3))*WRATIO+X1*COS(THETA(3))
     + +AMIN1(XBR,X2*COS(THETA(3)))
           IF(TBF(3) .GE. 0.0) GO TO 3030
              NOGOBF = 1
              GO TO 4000
3030       CONTINUE
           NOGOBF = 0
4000       CONTINUE
           RETURN
           END
        SUBROUTINE IV12
     + (BFMX    ,DCLMAX ,GCW    ,NTRAV  ,SFTYPC ,TBF )
C
C       ---------------------------------------------
C       MAXIMUM BRAKING FORCE - SOIL/SLOPE/VEHICLE/DRIVER
C       ---------------------------------------------
C
C       1. VARIABLE DECLARATION
           REAL        BFMX    (3)
           REAL        TBF     (3)
C       3. ALGORITHM
           BFMX(1) = AMIN1(DCLMAX*GCW , TBF(1)*SFTYPC/100.)
           IF(NTRAV .EQ. 1) GO TO 0010
              BFMX(2) = AMIN1(DCLMAX*GCW , TBF(2)*SFTYPC/100.)
              BFMX(3) = AMIN1(DCLMAX*GCW , TBF(3)*SFTYPC/100.)
3010       CONTINUE
           RETURN
           END
        SUBROUTINE IV13
     + (BFMX    ,EYEHGT ,GCW    ,NTRAV  ,RD     ,REACT  ,VELV   ,VISMNV )
C
C       --------------------------
C       SPEED LIMITED BY VISIBILITY
C       --------------------------
C
```

```
C       1. VARIABLE DECLARATION
           REAL        BFMX    (3)
           REAL        VELV    (3)
C       3. ALGORITHM
           DO 3030 K=1,NTRAV
              IF(BFMX(K) .GT. 0.0) GO TO 3010
                 VELV(K) = 0.0
                 GO TO 3030
3010          CONTINUE
              RECD     = RD*EYEHGT/60.
              ACC      = BFMX(K)*385.9/GCW
              D        = ((REACT*ACC)**2)+2.0*RECD*ACC
              C        = ACC*REACT
              VELV(K) = -(C-SQRT(D))
              IF(VELV(K) .GE. VISMNV) GO TO 3020
                 VELV(K)   =   VISMNV
3020          CONTINUE
3030       CONTINUE
           RETURN
           END
        SUBROUTINE IV14
       + (JPSI    ,NI      ,NTRAV   ,NVEHC   ,VELV    ,VRID    ,VSOIL   ,VTIRE
       + ,VTT )
C
C       -----------------------------------------------------------
C       SELECTED SPEED BETWEEN OBSTACLES LIMITED BY, VISIBILITY,
C       RIDE, TIRES, AND SOIL/SLOPE/VEGETATION RESISTANCE.
C       -----------------------------------------------------------
C
C       1. VARIABLE DECLARATION
           REAL        VELV    (3)
           REAL        VSOIL   (3,9)
           REAL        VTIRE   (3)
           REAL        VTT     (3,9)
C       3. ALGORITHM
           NI1 = NI+1
           DO 3030 K=1,NTRAV
              DO 3020 I=1,NI1
                 V1 = .99*VSOIL(K,I)
                 IF(NVEHC .EQ. -1) GO TO 3010
                    VTT(K,I) =
       + AMIN1(VTIRE(JPSI), V1, VRID, VELV(K))
                    GO TO 3020
3010             CONTINUE
                 VTT(K,I) =
       + AMIN1(V1, VRID, VELV(K))
3020          CONTINUE
3030       CONTINUE
           RETURN
           END
        SUBROUTINE IV15
       + (ADT     ,NI      ,NTRAV   ,NVEHC   ,PAV     ,VAVOID ,VBO     ,VTT )
```

```
C
C      ------------------------------------------------------
C      MAXIMUM SPEED BETWEEN AND AROUND OBSTACLES
C      ------------------------------------------------------
C
C      1. VARIABLE DECLARATION
          REAL       ADT     (9)
          REAL       PAV     (9)
          REAL       VAVOID  (3,9)
          REAL       VBO     (3,9)
          REAL       VTT     (3,9)
C      3. ALGORITHM
          NI1 = NI+1
          DO 3200 K=1,NTRAV
              DO 3190 I=1,NI1
                  VBO(K,I)=VTT(K,I)
                  VAVOID(K,I)=VTT(K,I)
                  IF(VTT(K,I) .EQ. 0.) GO TO 3190
                  IF(NVEHC .EQ. 1) GO TO 3060
C
C             A. TRACKED VEHICLE ROUTINE FOR MANEUVERING AROUND
C                AND BETWEEN OBSTACLES
C
                      IF(PAV(I) .LE. 3.) GO TO 3030
                          IF(PAV(I) .GE. 7.) GO TO 3010
                              SMG=((392.93-VTT(K,I))/4.)*PAV(I)
     +                            +(7.*VTT(K,I)-3.*392.93)/4.
                              VBO(K,I)=AMIN1(SMG,VTT(K,I))
                              GO TO 3030
3010                      IF(PAV(I) .GT. 52.5) GO TO 3020
                              SMG=453.15-8.603*PAV(I)
                              VBO(K,I)=AMIN1(SMG,VTT(K,I))
                              GO TO 3030
3020                      VBO(K,I)=0.
3030                  IF(ADT(I) .LE. 3.) GO TO 3190
                          IF(ADT(I) .GE. 7.) GO TO 3040
                              SMG=((392.93-VTT(K,I))/4.)*ADT(I)
     +                            +(7.*VTT(K,I)-3.*392.93)/4.
                              VAVOID(K,I)=AMIN1(SMG,VTT(K,I))
                              GO TO 3190
3040                      IF(ADT(I) .GT. 52.5) GO TO 3050
                              SMG=453.15-8.603*ADT(I)
                              VAVOID(K,I)=AMIN1(SMG,VTT(K,I))
                              GO TO 3190
3050                      VAVOID(K,I)=0.
                          GO TO 3190
C
C             B. WHEELED VEHICLE ROUTINE FOR MANEUVERING AROUND
C                AND BETWEEN OBSTACLES
C
3060              IF(PAV(I) .LE. 3.) GO TO 3090
                      IF(PAV(I) .GE. 7.) GO TO 3070
```

```
                      SMG=((450.33-VTT(K,I))/4.)*PAV(I)
     +                            +(7.*VTT(K,I)-3.*450.33)/4.
                      VBO(K,I)=AMIN1(SMG,VTT(K,I))
                      GO TO 3090
3070              IF(PAV(I) .GT. 41.3) GO TO 3080
                      SMG=542.11-13.112*PAV(I)
                      VBO(K,I)=AMIN1(SMG,VTT(K,I))
                      GO TO 3090
3080              VBO(K,I)=0.
3090           IF(ADT(I) .LE. 3.) GO TO 3190
                  IF(ADT(I) .GE. 7.) GO TO 3100
                      SMG=((450.33-VTT(K,I))/4.)*ADT(I)
     +                          +(7.*VTT(K,I)-3.*450.33)/4.
                      VAVOID(K,I)=AMIN1(SMG,VTT(K,I))
                      GO TO 3190
3100              IF(ADT(I) .GT. 41.3) GO TO 3110
                      SMG=542.11-13.112*ADT(I)
                      VAVOID(K,I)=AMIN1(SMG,VTT(K,I))
                      GO TO 3190
3110              VAVOID(K,I)=0.
3190           CONTINUE
3200     CONTINUE
         RETURN
         END
      SUBROUTINE IV16
     + (AVALS  ,CLEAR  ,FOM    ,FOMMAX ,FOO    ,FOOMAX ,HOVALS ,NANG
     + ,NEVERO ,NOHGT  ,NWDTH  ,OBAA   ,OBH    ,OBMINW ,WVALS )
C
C     ----------------------------------------
C     OBSTACLE OVERRIDE INTERFERENCE AND RESISTANCE
C     ----------------------------------------
C
C     1. VARIABLE DECLARATION
         REAL       AVALS  (14)
         REAL       CLEAR  (7,14,5)
         REAL       CLR
         REAL       FOM
         REAL       FOMMAX
         REAL       FOO    (7,14,5)
         REAL       FOOMAX (7,14,5)
         REAL       HOVALS (7)
         INTEGER    I
         INTEGER    II
         INTEGER    J
         INTEGER    JJ
         INTEGER    K
         INTEGER    KK
         INTEGER    N
         INTEGER    NANG
         INTEGER    NEVERO
         INTEGER    NOHGT
         INTEGER    NWDTH
```

```
      REAL      OBAA
      REAL      OBH
      REAL      OBMINW
      REAL      WVALS  (5)
C     3. ALGORITHM
      IF(NEVERO .EQ. 0) GO TO 3010
         FOM    = 0.0
         FOMMAX = 0.0
         GO TO 3150
3010  CONTINUE
      IF((NOHGT .NE. 1) .AND. (OBH .GT. HOVALS(1))) GO TO 3020
         I  = 1
         II = 1
         GO TO 3050
3020  CONTINUE
      DO 3030 N=2,NOHGT
         IF(OBH .LE. HOVALS(N)) GO TO 3040
3030  CONTINUE
         I  = NOHGT
         II = NOHGT
      GO TO 3050
3040  CONTINUE
         I  = N-1
         II = N
3050  CONTINUE
      IF((NANG .NE. 1) .AND. (OBAA .GT. AVALS(1))) GO TO 3060
         J  = 1
         JJ = 1
         GO TO 3090
3060  CONTINUE
      DO 3070 N=2,NANG
         IF(OBAA .LE. AVALS(N)) GO TO 3080
3070  CONTINUE
         J  = NANG
         JJ = NANG
         GO TO 3090
3080  CONTINUE
         J  = N-1
         JJ = N
3090  IF((NWDTH .NE. 1) .AND. (OBMINW .GT. WVALS(1))) GO TO 3100
         K  = 1
         KK = 1
         GO TO 3130
3100  CONTINUE
      DO 3110 N=2,NWDTH
         IF(OBMINW .LE. WVALS(N)) GO TO 3120
3110  CONTINUE
         K  = NWDTH
         KK = NWDTH
         GO TO 3130
3120  CONTINUE
         K  = N-1
```

```
          KK = N
3130    CONTINUE
        CALL D3LINC
     + (CLR     ,CLEAR   ,I       ,II      ,J
     + ,JJ      ,K       ,KK      ,HOVALS  ,AVALS
     + ,WVALS   ,OBH     ,OBAA    ,OBMINW  )
        IF(CLR .GE. 0.0) GO TO 3140
          NEVERO = 3
          GO TO 3150
3140    CONTINUE
        CALL D3LINC
     + (FOMMAX  ,FOOMAX  ,I       ,II      ,J
     + ,JJ      ,K       ,KK      ,HOVALS  ,AVALS
     + ,WVALS   ,OBH     ,CBAA    ,OBMINW  )
        CALL D3LINC
     + (FOM     ,FOO     ,I       ,II      ,J
     + ,JJ      ,K       ,KK      ,HOVALS  ,AVALS
     + ,WVALS   ,OBH     ,CBAA    ,OBMINW  )
3150    CONTINUE
        RETURN
        END
      SUBROUTINE IV17
     + (HVALS   ,NEVERO  ,NHVALS  ,NSVALS  ,OBH     ,OBSE    ,SVALS   ,TL
     + ,VOLA    ,VOOB    ,VOOBS   ,WA )
C
C     ----------------------------------------
C     DRIVER-DEPENDENT VEHICLE SPEED OVER OBSTACLES
C     ----------------------------------------
C
C     1. VARIABLE DECLARATION
        REAL       HVALS   (25)
        REAL       SVALS   (25)
        REAL       VOOB    (25)
        REAL       VOOBS   (25)
C     3. ALGORITHM
        VOLA = 0.0
        IF( NEVERO .NE. 0 ) GOTO 3090
          IF((OBSE-WA) .LT. TL) GO TO 3030
            DO 3020 NH=2,NHVALS
              IF(OBH .GT. HVALS(NH)) GO TO 3010
                VOLAB = VOOB(NH-1)+(OBH-HVALS(NH-1))
     + *(VOOB(NH)-VOOB(NH-1))/(HVALS(NH)-HVALS(NH-1))
                GO TO 3030
3010          CONTINUE
3020        CONTINUE
            VOLAB = VOOB(NHVALS)
3030      CONTINUE
          IF((OBSE-WA) .LT. (2.*TL)) GO TO 3040
            VOLA = VOLAB
            GO TO 3090
3040      CONTINUE
          DO 3060 NS=2,NSVALS
```

```
                IF(OBSE .GT. SVALS(NS)) GO TO 3050
                   VOLAS = V(OBS(NS-1)+(OBSE-SVALS(NS-1))
     + *(VOOBS(NS)-VOOBS(NS-1))/(SVALS(NS)-SVALS(NS-1))
                   GO TO 3070
3050            CONTINUE
3060         CONTINUE
             VOLAS = VOOBS(NSVALS)
3070         CONTINUE
             IF((OBSE-WA) .LT. TL) GO TO 3080
                VOLA = AMIN1(VOLAB,VOLAS)
                GO TO 3090
3080         CONTINUE
             VOLA = VOLAS
3090      CONTINUE
          RETURN
          END
       SUBROUTINE IV18
     + (FA      ,FB      ,FC      ,FOM     ,FOMMAX  ,FORMX   ,GCW     ,JPSI
     + ,MAXI    ,NEVERO  ,NGR     ,NI      ,NTRAV   ,NVEHC   ,OBSE    ,PAV
     + ,TL      ,TRESIS  ,VA      ,VBO     ,VELV    ,VFMAX   ,VG      ,VOLA
     + ,VRID    ,VSOIL   ,VTIRE   ,VXT     ,WA )
C
C      ---------------------------
C      SPEED ONTO AND OFF OBSTACLES
C      ---------------------------
C
C      1. VARIABLE DECLARATION
          REAL       FA      (20,3)
          REAL       FB      (20,3)
          REAL       FC      (20,3)
          REAL       FORMX   (3)
          REAL       PAV     (9)
          REAL       TRESIS  (3,9)
          REAL       VA      (3,9)
          REAL       VBO     (3,9)
          REAL       VELV    (3)
          REAL       VFMAX   (3)
          REAL       VG      (20,3,3)
          REAL       VSOIL   (3,9)
          REAL       VTIRE   (3)
          REAL       VTT     (3,9)
          REAL       VXT     (3,9)
C      3. ALGORITHM
          NI1=NI+1
          MAXI1=MAXI+1
          IF(NEVERO .EQ. 0) GO TO 3030
             DO 3020 K=1,NTRAV
                DO 3010 I=1,NI1
                   VBO(K,I)  = 0.
                   VA (K,I)  = 0.
                   VXT(K,I)  = 0.
3010            CONTINUE
```

```
3020          CONTINUE
              GO TO 3150
3030          CONTINUE
              IF(TL .LT. (OBSE+WA)) GO TO 3090
C             A. MULTIPLE OBSTACLE ROUTINE
                DO 3080 K=1,NTRAV
                  DO 3060 I=1,MAKI
                      RESIST = TRESIS(K,I)+FOM
                      CALL VELFOR
     + (RESIST,FA,FB,FC,FORMX,K,NGR,VSOIL(K,I),VFMAX,VG)
                      IF(NVEHC .EQ. -1) GO TO 3040
                        VTT(K,I) =
     + AMIN1(VSOIL(K,I),VRIDEVELV(K),VTIRE(JPSI),VOLA)
                        GO TO 3050
3040                  CONTINUE
                       VTT(K,I) =
     + AMIN1(VSOIL(K,I),VRIDEVELV(K),VOLA)
3050                  CONTINUE
                      VBO(K,I)=VTT(K,I)
                      IF(VBO(K,I) .EQ. 0.) GO TO 3059
                      IF(NVEHC .EQ. 1) GO TO 3056
C
C                     TRACKED VEHICLE ROUTINE FOR MANEUVERING
C                     BETWEEN OBSTACLES
C
                      IF(PAV(I) .LE. 3.) GO TO 3059
                          IF(PAV(I) .GE. 7.) GO TO 3051
                            SMG=((392.93-VTT(K,I))/4.)*PAV(I)
     +                           +(7.*VTT(K,I)-3.*392.93)/4.
                            VBO(K,I)=AMIN1(SMG,VTT(K,I))
                            GO TO 3059
3051                      IF(PAV(I) .GT. 52.5) GO TO 3052
                             SMG=453.15-8.603*PAV(I)
                             VBO(K,I)=AMIN1(SMG,VTT(K,I))
                             GO TO 3059
3052                      VBO(K,I)=0.
                          GO TO 3059
C
C                     WHEELED VEHICLE ROUTINE FOR MANEUVERING
C                     BETWEEN OBSTACLES
C
3056                  IF(PAV(I) .LE. 3.) GO TO 3059
                        IF(PAV(I) .GE. 7.) GO TO 3057
                          SMG=((450.33-VTT(K,I))/4.)*PAV(I)
     +                         +(7.*VTT(K,I)-3.*450.33)/4.
                          VBO(K,I)=AMIN1(SMG,VTT(K,I))
                          GO TO 3059
3057                    IF(PAV(I) .GT. 41.3) GO TO 3058
                          SMG=542.11-13.112*PAV(I)
                          VBO(K,I)=AMIN1(SMG,VTT(K,I))
                          GO TO 3059
3058                    VBO(K,I)=0.
```

```
3059                VA (K,I)=VBO(K,I)
                    VXT(K,I)=VBO(K,I)
3060             CONTINUE
                 DO 3070 I=MAXI1,NI1
                    VA (K,I) = 0.
                    VXT(K,I) = 0.
                    VBO(K,I) = 0.
3070             CONTINUE
3080          CONTINUE
              GO TO 3150
C           B. SINGLE OBSTACLE ROUTINE
3090          CONTINUE
              DO 3140 K = 1,NTRAV
                 FORRQ = TRESIS(K,1) + FOM
                 DO 3130 I = 1,MAXI
                    VA(K,I) = AMIN1(VBC(K,I) ,VOLA)
                    CALL FORVEL
     + (F,FA,FB,FC,FORMX,K,NGR,VA(K,I),VFMAX,VG)
                    FORCEF = FORRQ-F
                    IF(FORCEF .GT. 0.) GO TO 3100
                       VXT(K,I) = VA(K,I)
                       GO TO 3130
3100                CONTINUE
                    VBSC = VA(K,I)**2-FORCEF*(WA*TL)*385.9/GCW
                    IF(VBSC .GT. 0.) GO TO 3120
                       IF
     + ((FORMX(K)-TRESIS(K,1)-FOMMAX) .GE. 0.) GO TO 3110
                          VXT(K,I) = 0.
                          VA (K,I) = 0.
                          GO TO 3130
3110                   CONTINUE
                       CALL VELFOR
     +((TRESIS(K,1)+FOMMAX),FA,FB,FC,FORMX,K,NGR,VXT(K,I),VFMAX,VG)
                       VA(K,I) = VXT(K,I)
                       GO TO 3130
3120                CONTINUE
                    VXT(K,I) = SQRT(VBSC)
3130             CONTINUE
                 DO 3140 I = MAXI1,NI1
                    VXT(K,I) = 0.
                    VA(K,I) = 0.
                    VBC (K,I) = 0.
3140             CONTINUE
3150          CONTINUE
              DO 3170 K=1,NTRAV
                 DO 3160 I=1,MAXI
                    IF( VA(K,I) .GT. VBO(K,I))  VA(K,I)=VBO(K,I)
                    IF(VXT(K,I) .GT.  VA(K,I)) VXT(K,I)= VA(K,I)
3160             CONTINUE
3170          CONTINUE
         RETURN
         END
```

```
      SUBROUTINE IV19
     + (BFMX    ,FA      ,FB      ,FC      ,FOMMAX ,FORMX   ,GCW     ,LUN1
     + ,NEVERO ,NGR     ,NI      ,NTRAV   ,OBSE    ,RMX     ,STRACT  ,TL
     + ,TRESIS ,VA      ,VBO     ,VFMAX   ,VG      ,VOVER   ,VXT     ,WA )
C
C     ------------------------------------------------------------
C     AVERAGE PATCH SPEED ACCELERATING/DECELERATING BETWEEN
C     OBSTACLES AND CROSSING OBSTACLES
C     ------------------------------------------------------------
C
C     1. VARIABLE DECLARATION
         REAL        BFMX    (3)
         REAL        FA      (20,3)
         REAL        FB      (20,3)
         REAL        FC      (20,3)
         REAL        FORMX   (3)
         REAL        RMX     (20)
         REAL        STRACT  (20,3,3)
         REAL        TRESIS  (3,9)
         REAL        VA      (3,9)
         REAL        VBO     (3,9)
         REAL        VFMAX   (3)
         REAL        VG      (20,3,3)
         REAL        VOVER   (3,9)
         REAL        VXT     (3,9)
C        3. ALGORITHM
            NI1 = NI+1
            DO 3020 K = 1,NTRAV
               DO 3010 I = 1,NI1
                  VOVER(K,I) = 0.
3010           CONTINUE
3020        CONTINUE
            IF(NEVERO .NE. 0) GO TO 4000
            CONTINUE
            VM = GCW/385.9
            DO 3150 K = 1,NTRAV
               DO 3140 I = 1,NI1
C                 DETERMINE IF OBSTACLE EXIT SPEED, IS SPEED
C    + BETWEEN OBSTACLES, IF SO IT IS "VOVER"
                  IF(VA(K,I) .EQ. 0. .AND. VXT(K,I) .EQ. 0.)
     + GO TO 3140
                  IF(VBO(K,I) .GT. VXT(K,I)) GO TO 3040
                     VOVER(K,I) = VXT(K,I)
                     GO TO 3140
3040              CONTINUE
C                 DETERMINE IF THERE IS ENOUGH DISTANCE
C    + BETWEEN OBSTACLES TO ACCELERATE TO OBSTACLE APPROACH SPEED.
                  IF(VXT(K,I) .GE. VA(K,I)) GO TO 3070
                     CALL ACCEL
     + (FA,FB,FC,GCW,I,K,NGF,NGR,NV2FLG,RMX,STRACT,
     +  TA,TRESIS,VXT(K,I),VA(K,I),V2F,VG,XA,LUN1)
                     IF
```

```
     + ((XA .LE. (OBSE-WA-TL)) .AND. (NV2FLG .EQ. 0)) GO TO 3060
                              DUMMY = TRESIS(K,1)*FOMMAX
                              IF(FORMX(K) .LT. DUMMY) GO TO 3050
                                 CALL VELFOR
     +  (DUMMY,FA,FB,FC,FORMX,K,NGR,VOVER(K,I),VFMAX,VG)
                                 IF(VOVER(K,I) .GE. VXT(K,I)) GO TO 3140
                                    VOVER(K,I) = VXT(K,I)
                                    GO TO 3140
3050                          CONTINUE
                              VOVER(K,I) = 0.
                              GO TO 3140
3060                       CONTINUE
3070                    CONTINUE
C                       ACCELERATION/DECELERATION REQUIRED; DETERMINE
C    + IF "VBO" CAN BE REACHED IN TIME TO DECELERATE TO "VA"
                        CALL ACCEL
     + (FA,FB,FC,GCW,I,K,NGF,NGR,NV2FLG,RMX,STRACT,
     +  TA,TRESIS,VXT(K,I),VBO(K,I),V2F,VG,XA,LUN1)
                        IF(NV2FLG .EQ. 2) GO TO 3090
C VBO                      CAN BE REACHED
                           TB = VM*(VBO(K,I)-VA(K,I))/BFMX(K)
                           XB = .5*(VBO(K,I)+VA(K,I))*TB
                           IF((XA+XB) .GT. (OBSE-WA-TL)) GO TO 3080
C                             ENOUGH SPACE BETWEEN OBSTACLES
                              TOO          = 2.*(WA+TL)/(VA(K,I)+VXT(K,I))
                              TBO          = (OBSE-WA-TL-XA-XB)/VBO(K,I)
                              VOVER(K,I)  = OBSE/(TA+TBO+TB+TOO)
                              GO TO 3140
3080                       CONTINUE
3090                    CONTINUE
C                       ACCELERATION/DECELERATION REQUIRED;
C    + DETERMINE VELOCITY WHICH CAN BE REACHED BEFORE
C    + DECELERATION IS REQUIRED AND TIME BETWEEN AND
C    + OVER OBSTACLES.
                        VLOW = VA(K,I)
                        VHGH = VBO(K,I)
                        DO 3130 J = 1,10
                           VMID = (VLOW+VHGH)/2.
                           CALL ACCEL
     + (FA,FB,FC,GCW,I,K,NGF,NGR,NV2FLG,RMX,STRACT,
     +  TA,TRESIS,VXT(K,I),VMID,V2F,VG,XA,LUN1)
                           IF(NV2FLG .NE. 2) GO TO 3100
                              VHGH = VMID
                              GO TO 3130
3100                       IF(NV2FLG .NE. 1) GO TO 3110
                              VHGH = VLOW
                              VLOW = VA(K,I)
                              GO TO 3130
3110                       CONTINUE
                           TB = VM*(VMID-VA(K,I))/BFMX(K)
                           XB = 0.5*(VMID+VA(K,I))*TB
                           IF((XA+XB) .LE. (OBSE-WA-TL)) GO TO 3120
```

```
                        VHGH = VMID
                        GO TO 3130
3120                 CONTINUE
                     VLOW = VMID
3130              CONTINUE
                  IF((XA+XB) .LE. (OBSE-WA-TL)) GO TO 3145
                      VBO(K,I) = VBC(K,I)-17.6
                      GO TO 3070
3145              TBO = (OBSE-WA-TL-XA-XB)/VMID
                  TOO          = 2.*(WA+TL)/(VA(K,I)+VXT(K,I))
                  VOVER(K,I) = OBSE/(TA+TBO+TB+TOO)
3140           CONTINUE
3150           CONTINUE
4000      CONTINUE
          RETURN
          END
       SUBROUTINE IV20
      + (FA      ,FB      ,FC      ,FORMX   ,GCW     ,MAXI    ,NGR     ,NOGOVA
      + ,NOGOVO ,NTRAV   ,STR     ,VAVOID  ,VG      ,VOVER )
C
C      ---------------------------------------
C      KINEMATIC VEGETATION OVERRIDE CHECK
C      ---------------------------------------
C
C      1.VARIABLE DECLARATION
         REAL      FA      (20,3)
         REAL      FB      (20,3)
         REAL      FC      (20,3)
         REAL      FORMX   (3)
         INTEGER   NOGOVA  (3,9)
         INTEGER   NOGOVO  (3,9)
         REAL      STR     (3,9)
         REAL      VAVOID  (3,9)
         REAL      VG      (20,3,3)
         REAL      VOVER   (3,9)
C      3. ALGORITHM
         DO 3050 K = 1,NTRAV
            DO 3040 I = 1,MAXI
               CALL FORVEL (F,FA,FB,FC,FORMX,K,NGR,VOVER(K,I),VFMAX,VG)
               DUMMY  =   F+(.5*GCW*VOVER(K,I)**2.)/385.9
               IF(DUMMY .GT. STR(K,I)) GO TO 3010
                  NOGOVO(K,I) = 0
                  GO TO 3020
3010           CONTINUE
               NOGOVO(K,I) = 1
3020           CONTINUE
               CALL FORVEL (F,FA,FB,FC,FORMX,K,NGR,VAVOID(K,I),VFMAX,VG)
               DUMMY = F+(.5*GCW*VAVOID(K,I)**2.)/385.9
               IF(DUMMY .GT. STR(K,I)) GO TO 3030
                  NOGOVA(K,I) = 0
                  GO TO 3050
3030           CONTINUE
```

```
                NOGOVA(K,I) = 1
3040          CONTINUE
3050      CONTINUE
          RETURN
          END
      SUBROUTINE IV21
     + (DOWN    ,FMT     ,GCW     ,IMAX    ,IOVER   ,ISAFE   ,LEVEL   ,NI
     + ,NOGOVA  ,NOGOVC  ,NTRAV   ,UP      ,VAVOID  ,VMAX    ,VMAX1   ,VMAX2
     + ,VOVER   ,VSEL    ,VSEL1   ,VSEL2   ,VWALK )
C
C     ------------------------------
C     MAXIMUM AVERAGE SPEED ALGORITHM
C     ------------------------------
C
C     1. VARIABLE DECLARATION
C
          REAL        A
          REAL        B
          INTEGER     DOWN
          REAL        FMT     (9)
          INTEGER     I
          INTEGER     IOVER
          INTEGER     IMAX    (3)
          INTEGER     ISAFE   (3)
          INTEGER     K
          INTEGER     KIV21
          INTEGER     LEVEL
          INTEGER     LUN1
          INTEGER     NI
          INTEGER     NI1
          INTEGER     NOGOVA (3,9)
          INTEGER     NOGOVC (3,9)
          INTEGER     NTRAV
          INTEGER     UP
          REAL        VAVOID (3,9)
          REAL        VMAX
          REAL        VMAX1   (3)
          REAL        VMAX2   (3,9)
          REAL        VOVER   (3,9)
          REAL        VSEL
          REAL        VSEL1 (3)
          REAL        VSEL2 (3,9)
          REAL        VWALK
C
C     3. ALGORITHM
C
          NI1 =NI+1
          DO 3020 K=1,NTRAV
             DO 3010 I=1,NI1
                A = VOVER (K,I)*FLOAT(NOGOVC(K,I))
                B = VAVOID(K,I)*FLOAT(NOGOVA(K,I))
                VMAX2(K,I) = AMAX1(A,B)
```

```
3010          CONTINUE
3020       CONTINUE
           DO 3070 K=1,NTRAV
              DO 3060 I=1,NI1
                 IF(VMAX2(K,I) .GT. VWALK) GO TO 3030
                    VSEL2(K,I) = VMAX2(K,I)
                    GO TO 3050
3030             CONTINUE
                 IF((FMT(I)/GCW).GT.1..OR.I.GE.IOVER) GO TO 3040
                    VSEL2(K,I) = VMAX2(K,I)
                    GO TO 3050
3040             CONTINUE
                    VSEL2(K,I) = 0.0
3050             CONTINUE
3060          CONTINUE
3070       CONTINUE
           DO 3100 K=1,NTRAV
              VMAX1(K) = VMAX2(K,1)
              IMAX(K)  = 1
              DO 3090 I=2,NI1
                 IF(VMAX1(K) .GE. VMAX2(K,I)) GO TO 3080
                    VMAX1(K) = VMAX2(K,I)
                    IMAX(K)  = I
3080             CONTINUE
3090          CONTINUE
3100       CONTINUE
           DO 3130 K=1,NTRAV
              VSEL1(K) = VSEL2(K,1)
              ISAFE(K)  = 1
              DO 3120 I=2,NI1
                 IF(VSEL1(K) .GE. VSEL2(K,I)) GO TO 3110
                    VSEL1(K) = VSEL2(K,I)
                    ISAFE(K)  = I
3110             CONTINUE
3120          CONTINUE
3130       CONTINUE
C
           IF(NTRAV .EQ. 1) GO TO 3150
              IF(VMAX1(UP)     .EQ. 0.0) GO TO 3140
              IF(VMAX1(LEVEL)  .EQ. 0.0) GO TO 3140
              IF(VMAX1(DOWN)   .EQ. 0.0) GO TO 3140
                 VMAX =
     + 3.0/((1./VMAX1(UP))+(1./VMAX1(LEVEL))+(1./VMAX1(DOWN)))
                 GO TO 3160
3140          CONTINUE
              VMAX = 0.0
              GO TO 3160
3150       CONTINUE
           VMAX = VMAX1(UP)
3160       CONTINUE
           IF(NTRAV .EQ. 1) GO TO 3180
              IF(VSEL1(UP)     .EQ. 0.0) GO TO 3170
```

```
            IF(VSEL1(LEVEL) .EQ. 0.0) GO TO 3170
            IF(VSEL1(DOWN)  .EQ. 0.0) GO TO 3170
              VSEL =
     + 3.0/((1./VSEL1(UP))+(1./VSEL1(LEVEL))+(1./VSEL1(DOWN)))
              GO TO 3190
3170        CONTINUE
            VSEL = 0.0
            GO TO 3190
3180      CONTINUE
          VSEL = VSEL1(UP)
3190      CONTINUE
          RETURN
          END
C         ----------------------
C         ROADWAY MODULE - AMC 74
C         ----------------------
C
C
      SUBROUTINE ROAD
C
          INTEGER             MO
          COMMON /INDEX/      MO
          INTEGER             COWN
          COMMON /INDEX/      CEWN
          INTEGER             EFF
          COMMON /INDEX/      EFF
          INTEGER             FORCE
          COMMON /INDEX/      FORCE
          INTEGER             GR
          COMMON /INDEX/      GR
          COMMON /INDEX/      LEVEL
          INTEGER             MN
          COMMON /INDEX/      MN
          INTEGER             RPM
          COMMON /INDEX/      RPM
          INTEGER             SPEED
          COMMON /INDEX/      SPEED
          INTEGER             SR
          COMMON /INDEX/      SR
          INTEGER             TR
          COMMON /INDEX/      TR
          INTEGER             TORCUE
          COMMON /INDEX/      TORQUE
          INTEGER             UP
          COMMON /INDEX/      UP
          INTEGER             MX
          COMMON /INDEX/      MX
          COMMON /VEHICL/     ACD
          COMMON /VEHICL/     ASHOE      (20)
          COMMON /VEHICL/     AVGC
          COMMON /VEHICL/     AXLSP      (20)
          COMMON /VEHICL/     CD
```

```
      COMMON /VEHICL/   CGH
      COMMON /VEHICL/   CGLAT
      COMMON /VEHICL/   CGR
      COMMON /VEHICL/   CID
      COMMON /VEHICL/   CW
      COMMON /VEHICL/   CLRMIN     (20)
      COMMON /VEHICL/   CONV1      (2,25)
      COMMON /VEHICL/   CONV2      (2,25)
      COMMON /VEHICL/   CSLCT      (20,3)
      COMMON /VEHICL/   DIAW       (20)
      COMMON /VEHICL/   DRAFT
      COMMON /VEHICL/   ENGINE     (2,50)
      COMMON /VEHICL/   EXEWGT
      COMMON /VEHICL/   FD         (2)
      COMMON /VEHICL/   FORDD
      COMMON /VEHICL/   GROUSH     (20)
      COMMON /VEHICL/   HVALS      (25)
      COMMON /VEHICL/   IAPG
      COMMON /VEHICL/   IB         (20)
      COMMON /VEHICL/   ID         (20)
      REAL              IDIESL
      COMMON /VEHICL/   IDIESL
      COMMON /VEHICL/   IENGIN
      COMMON /VEHICL/   IR         (20)
      COMMON /VEHICL/   ICONST     (20)
      COMMON /VEHICL/   ICONV1
      COMMON /VEHICL/   ICONV2
      COMMON /VEHICL/   IPOWER
      COMMON /VEHICL/   IT         (20)
      COMMON /VEHICL/   ITCASE
      COMMON /VEHICL/   ITRAN
      COMMON /VEHICL/   ITVAR
      COMMON /VEHICL/   LOCKUP
      COMMON /VEHICL/   MAXIPR
      COMMON /VEHICL/   MAXL
      COMMON /VEHICL/   NAMBLY
      COMMON /VEHICL/   NBOGIE     (20)
      COMMON /VEHICL/   NCHAIN     (20)
      REAL              NCYL
      COMMON /VEHICL/   NCYL
      REAL              NENG
      COMMON /VEHICL/   NENG
      COMMON /VEHICL/   NONET
      COMMON /VEHICL/   NGL        (20)
      COMMON /VEHICL/   NGR
      COMMON /VEHICL/   NHVALS
      COMMON /VEHICL/   NPAD       (20)
      COMMON /VEHICL/   NSVALS
      COMMON /VEHICL/   NVEH       (20)
      COMMON /VEHICL/   NWHL       (20)
      COMMON /VEHICL/   NWR
      COMMON /VEHICL/   PBF
```

```
COMMON /VEHICL/   FBHT
COMMON /VEHICL/   FPA
COMMON /VEHICL/   POWER    (2,201)
COMMON /VEHICL/   CMAX
COMMON /VEHICL/   FDIAM    (20)
COMMON /VEHICL/   REVM     (20)
COMMON /VEHICL/   RIMW     (20)
COMMON /VEHICL/   RMS      (20)
COMMON /VEHICL/   RW       (20)
COMMON /VEHICL/   SAE
COMMON /VEHICL/   SAI
COMMON /VEHICL/   SECTH    (20)
COMMON /VEHICL/   SECTW    (20)
COMMON /VEHICL/   SWALS    (25)
COMMON /VEHICL/   TCASE    (2)
COMMON /VEHICL/   TI
COMMON /VEHICL/   TPLY     (20)
COMMON /VEHICL/   TPSI     (20,3)
COMMON /VEHICL/   TQINC
COMMON /VEHICL/   TRAKLN   (20)
COMMON /VEHICL/   TRAKWD   (20)
COMMON /VEHICL/   TRANS    (2,20)
COMMON /VEHICL/   VAA
COMMON /VEHICL/   VBA
COMMON /VEHICL/   VBS
COMMON /VEHICL/   VOGB     (25)
COMMON /VEHICL/   VOOBS    (25)
COMMON /VEHICL/   VRIDE    (20,3)
COMMON /VEHICL/   VSS
COMMON /VEHICL/   WSSAXP
COMMON /VEHICL/   WG
COMMON /VEHICL/   WOAXP
COMMON /VEHICL/   WDPTH    (20)
COMMON /VEHICL/   WDTH
COMMON /VEHICL/   WGHT     (20)
COMMON /VEHICL/   WRAT     (20)
COMMON /VEHICL/   WRFORD
COMMON /VEHICL/   WT       (20)
COMMON /VEHICL/   WTE      (20)
COMMON /VEHICL/   WMAXF
COMMON /VEHICL/   XBRCOF
COMMON /VEHICL/   WI
COMMON /VEHICL/   LOCDIF
COMMON /VEHICL/   SHF
COMMON /PREP/     A        (3,4)
COMMON /PREP/     AIF      (20)
COMMON /PREP/     BTF      (20)
COMMON /PREP/     CHARLN   (20,3)
COMMON /PREP/     CRFCFG   (3)
COMMON /PREP/     CRFCCG   (3)
COMMON /PREP/     CRFCG    (20,3)
COMMON /PREP/     CRFFG    (20,3)
```

```
      COMMON /PREP/       CFF         (2Ø)
      COMMON /PREP/       CRAT        (2Ø,3)
      COMMON /PREP/       CGA         (2Ø,3)
      COMMON /PREP/       CGW
      COMMON /PREP/       CGWB
      COMMON /PREP/       CGWNB
      COMMON /PREP/       CGWNP
      COMMON /PREP/       CGWP
      COMMON /PREP/       HGT
      COMMON /PREP/       NBF         (38)
      COMMON /PREP/       NVEHC
      COMMON /PREP/       PNTE
      COMMON /PREP/       R           (3)
      COMMON /PREP/       RMX         (2Ø)
      COMMON /PREP/       RR
      COMMON /PREP/       TRACTF      (2Ø,5)
      COMMON /PREP/       TRAPSI      (3)
      COMMON /PREP/       VCICG       (2Ø,3)
      COMMON /PREP/       VCIFG       (2Ø,3)
      COMMON /PREP/       VCIMUK      (2Ø)
      COMMON /PREP/       VGV         (2Ø,5)
      COMMON /PREP/       VI          (2Ø,3)
      COMMON /PREP/       VTIRE       (3)
      COMMON /PREP/       WTMAX
      COMMON /PREP/       X           (3)
      COMMON /PREP/       XBR
      COMMON /DERIVE/     ADT         (9)
      COMMON /DERIVE/     ACGOBF
      COMMON /DERIVE/     CAREA
      COMMON /DERIVE/     CGWB
      COMMON /DERIVE/     CGWP
      COMMON /DERIVE/     CGWPB       (2Ø)
      COMMON /DERIVE/     FA          (2Ø,3)
      COMMON /DERIVE/     FAT         (9)
      COMMON /DERIVE/     FAT1        (9)
      COMMON /DERIVE/     FB          (2Ø,3)
      COMMON /DERIVE/     FC          (2Ø,3)
      COMMON /DERIVE/     FNT         (9)
      COMMON /DERIVE/     FOM
      COMMON /DERIVE/     FOMMAX
      COMMON /DERIVE/     FORMX       (3)
      COMMON /DERIVE/     IFLOAT
      COMMON /DERIVE/     IMAX        (3)
      COMMON /DERIVE/     ISAFE       (3)
      COMMON /DERIVE/     J
      COMMON /DERIVE/     MAXI
      COMMON /DERIVE/     REMX        (3)
      COMMON /DERIVE/     NEVERO
      COMMON /DERIVE/     OBSE
      COMMON /DERIVE/     PAV         (9)
      COMMON /DERIVE/     RIGWB
      COMMON /DERIVE/     RIGWNB
```

```
      COMMON /DERIVE/   RTOWNP
      COMMON /DERIVE/   RTOWP
      COMMON /DERIVE/   RTOWP8    (2Ø)
      COMMON /DERIVE/   RTOWT     (2Ø)
      COMMON /DERIVE/   STRACT    (2Ø,3,3)
      COMMON /DERIVE/   SRFO      (9)
      COMMON /DERIVE/   SRFV      (9)
      COMMON /DERIVE/   STR       (3,9)
      COMMON /DERIVE/   TBF       (3)
      COMMON /DERIVE/   TDEN      (9)
      COMMON /DERIVE/   TRESIS    (3,9)
      COMMON /DERIVE/   VA        (3,9)
      COMMON /DERIVE/   NOGOVA    (3,9)
      COMMON /DERIVE/   VAVOID    (3,9)
      COMMON /DERIVE/   VBO       (3,9)
      COMMON /DERIVE/   IMPACT    (9)
      COMMON /DERIVE/   VELV      (3)
      COMMON /DERIVE/   VEMAX     (3)
      COMMON /DERIVE/   VG        (2Ø,3,3)
      COMMON /DERIVE/   NOGOVO    (3,9)
      COMMON /DERIVE/   VMAX
      COMMON /DERIVE/   VMAX1     (3)
      COMMON /DERIVE/   VMAX2     (3,9)
      COMMON /DERIVE/   VOLA
      COMMON /DERIVE/   VOVER     (3,9)
      COMMON /DERIVE/   VRID
      COMMON /DERIVE/   VSEL
      COMMON /DERIVE/   VSEL1     (3)
      COMMON /DERIVE/   VSEL2     (3,9)
      COMMON /DERIVE/   VSOIL     (3,9)
      COMMON /DERIVE/   VTT       (3,9)
      COMMON /DERIVE/   VXT       (3,9)
      COMMON /DERIVE/   NOGONO
      COMMON /DERIVE/   WRATIO
      COMMON /TERRAN/   AA
      COMMON /TERRAN/   ACTRMS
      COMMON /TERRAN/   AREA
      COMMON /TERRAN/   AREAO
      COMMON /TERRAN/   CI
      COMMON /TERRAN/   CIST
      COMMON /TERRAN/   EANG
      COMMON /TERRAN/   ECF
      COMMON /TERRAN/   ELEV
      COMMON /TERRAN/   FMU       (3)
      COMMON /TERRAN/   GRADE
      COMMON /TERRAN/   IOBS
      COMMON /TERRAN/   IOST
      COMMON /TERRAN/   IROAD
      COMMON /TERRAN/   IS        (9)
      COMMON /TERRAN/   IST
      COMMON /TERRAN/   ITUT
      COMMON /TERRAN/   NI
```

```
      COMMON /TERRAN/   NJU
      COMMON /TERRAN/   CAW
      COMMON /TERRAN/   CBAA
      COMMON /TERRAN/   CBH
      COMMON /TERRAN/   CBL
      COMMON /TERRAN/   CBS
      COMMON /TERRAN/   CBW
      COMMON /TERRAN/   CBMINW
      COMMON /TERRAN/   CDIA
      COMMON /TERRAN/   NVASHO
      COMMON /TERRAN/   RADC
      COMMON /TERRAN/   RCI
      COMMON /TERRAN/   RCIC       (4)
      COMMON /TERRAN/   RCURV      (11)
      COMMON /TERRAN/   RB
      COMMON /TERRAN/   RDA        (12)
      COMMON /TERRAN/   S          (9)
      COMMON /TERRAN/   SB         (9)
      COMMON /TERRAN/   SDL        (9)
      COMMON /TERRAN/   SURFF
      COMMON /TERRAN/   TANPHI
      COMMON /TERRAN/   THETA      (3)
      COMMON /TERRAN/   VCURV      (4,11)
      COMMON /TERRAN/   WB
      COMMON /TERRAN/   WD
      COMMON /SCEN/     COHES
      COMMON /SCEN/     VWALK
      COMMON /SCEN/     COLMAX
      COMMON /SCEN/     GAMMA
      COMMON /SCEN/     ICVER
      COMMON /SCEN/     ISEASN
      COMMON /SCEN/     ISURF
      COMMON /SCEN/     ISNOW
      COMMON /SCEN/     KII1
      COMMON /SCEN/     KII2
      COMMON /SCEN/     KII3
      COMMON /SCEN/     KII4
      COMMON /SCEN/     KII5
      COMMON /SCEN/     KII6
      COMMON /SCEN/     KII7
      COMMON /SCEN/     KII8
      COMMON /SCEN/     KII9
      COMMON /SCEN/     KII10
      COMMON /SCEN/     KII11
      COMMON /SCEN/     KII12
      COMMON /SCEN/     KII13
      COMMON /SCEN/     KII14
      COMMON /SCEN/     KII15
      COMMON /SCEN/     KII16
      COMMON /SCEN/     KII17
      COMMON /SCEN/     KMAP
      COMMON /SCEN/     KSCEN
```

```
      COMMON /SCEN/     KIPP
      COMMON /SCEN/     KVEH
      COMMON /SCEN/     KIV1
      COMMON /SCEN/     KIV2
      COMMON /SCEN/     KIV3
      COMMON /SCEN/     KIV4
      COMMON /SCEN/     KIV5
      COMMON /SCEN/     KIV6
      COMMON /SCEN/     KIV7
      COMMON /SCEN/     KIV8
      COMMON /SCEN/     KIV9
      COMMON /SCEN/     KIV10
      COMMON /SCEN/     KIV11
      COMMON /SCEN/     KIV12
      COMMON /SCEN/     KIV13
      COMMON /SCEN/     KIV14
      COMMON /SCEN/     KIV15
      COMMON /SCEN/     KIV16
      COMMON /SCEN/     KIV17
      COMMON /SCEN/     KIV18
      COMMON /SCEN/     KIV19
      COMMON /SCEN/     KIV20
      COMMON /SCEN/     KIV21
      COMMON /SCEN/     LAC
      INTEGER           CDTAIL
      COMMON /SCEN/     CDTAIL
      COMMON /SCEN/     MAP
      COMMON /SCEN/     MAPG
      COMMON /SCEN/     MONTH
      COMMON /SCEN/     NOPP
      COMMON /SCEN/     NSLIP
      COMMON /SCEN/     NTRAV
      COMMON /SCEN/     NOUX
      COMMON /SCEN/     PHI
      COMMON /SCEN/     REACT
      COMMON /SCEN/     RDFCG
      INTEGER           SEARCH
      COMMON /SCEN/     SEARCH
      COMMON /SCEN/     SETYPC
      COMMON /SCEN/     VBRAKE
      COMMON /SCEN/     VISMNV
      COMMON /SCEN/     VMIN
      COMMON /SCEN/     ZSNOW
C     1. VARIABLE DECLARATIONS
      REAL    FATEMP (20,3)
      REAL    FBTEMP (20,3)
      REAL    FCTEMP (20,3)
      REAL    FTEMP  (20,5)
      REAL    FTEMPC (20,5)
      REAL    FGRADE (3)
      REAL    FQUAD  (5)
      REAL    K1
```

```
            REAL    MERITK
            REAL    VGTEMP (20,3,3)
            REAL    VPOWER (3)
            REAL    VTEMP  (20,5)
            REAL    VTEMPO (20,5)
            REAL    VROAD  (3)
            REAL    VQUAD  (5)
C
C     2. ALGORITHMS
C
C           ----------------------------------------------------------------
C           ROUTINE 1 - SPEED LIMITED BY AEROCYNAMIC, ROLLING, CORNERING
C                       AND GRADE RESISTANCE
C           ----------------------------------------------------------------
C
C           SECTION 1A - INITIALIZATION
C
          FCC        = 0.
          FCTRAK     = 0.
          FR         = 0.
          FTC        = 0.
          IGAP       = 0
          WTSPA      = 0.
          WRATIO     = 1
C
C           1A.1 - SUPERELEVATION EFFECT FACTOR
C
          FE = 1. - 7.495*(RADC/12.)*EANG
C
          DO 6005 NG=1,NGR
             DO 6000 L=1,5
                VTEMP(NG,L) = VGV(NG,L)
                FTEMP(NG,L) = TRACTF(NG,L)
6000         CONTINUE
6005      CONTINUE
          IF(NCPP .NE. 0) JPSI = NCPP
          IF(NCPP .EQ. 0) JPSI = 3
C
C           SECTION 1B - VELOCITY DEPENDENT RESISTANCES
C
C           1B.1 - SOFT SURFACES  (TRAILS)
C
          IF(ITUT .NE. 14) GO TO 6102
             CALL IV3(
     +    CHARLN ,CI     ,COMES  ,CPFFG  ,DIAW   ,DOWPB  ,DRAT   ,GAMMA
     +   ,GCA    ,IB     ,IC     ,IP     ,IST    ,JPSI   ,LUN1   ,NAMBLY
     +   ,NPAD   ,NSLIP  ,NVEH   ,NWHL   ,RCI    ,RTOWPB ,RTOWT  ,SECTW
     +   ,TANPHI ,TRAKLN ,TRAKWD ,VCIFG  ,VCIMUK ,WGHT   ,WRATIO ,ZSNOW)
             CALL IV4(
     +    DOWB   ,DOWP   ,CCWPB  ,CCW    ,GCWB   ,GCWP   ,IB     ,IR
     +   ,NAMBLY ,RTOWB  ,RTOWNB ,RTOWNP ,RTOWP  ,RTOWPB ,RTOWT  ,WGHT )
             CALL IV5(
```

```
     +      ATF      ,AVGC     ,BTE      ,CD       ,CPFCCG ,CPFCFG ,CTF      ,DAREA
     +     ,DOWP     ,EANG     ,ECE      ,FA       ,FB       ,FC       ,FORMX    ,GCWP
     +     ,IFLOAT   ,IST      ,ITWT     ,JPSI     ,LOCDIF ,NAMBLY ,NFL      ,NGR
     +     ,NTRAV    ,NVEH     ,NVEHC    ,NWHL     ,RADC     ,RTOWP    ,STRACT ,TEOR
     +     ,THETA    ,TRACTF ,VFMAX    ,VG       ,VGV      ,WGHT     ,WRATIO ,LUN1 )
        GO TO 6161
C
C       18.2 HARD SURFACE (PRIMARY AND SECONDARY ROADS)
C
6102    DO 6155 K=1,NTRAV,2
           NG1  = 1
           COSX = COS(THETA(K))
           DO 6154 NG=1,NGR
              FA(NG,K) = 0.
              FB(NG,K) = 0.
              FC(NG,K) = 0.
              FATEMP(NG,K) = ATF(NG)
              FBTEMP(NG,K) = BTF(NG)
              FCTEMP(NG,K) = CTF(NG)
C
C       18.2.1 - TRACTIVE EFFORT MODIFIED FOR SURFACE TRACTION
C                AND SPEED DEPENDENT RESISTANCES
C
                 IF(TRACTF(NG,1)*ECF .LT. FMU(ISURF)*GCWP*COSX)
     +           GO TO 6130
                  IF(TRACTF(NG,5)*ECF .LT. FMU(ISURF)*GCWP*COSX)
     +            GO TO 6110
C
C        MINIMUM GEAR LIMITED BY SURFACE TRACTION
C
                  NG1 = NG1 + 1
                  FC(NG,K) = 0.
                  FB(NG,K) = 0.
                  FA(NG,K) = FMU(ISURF)*GCWP*COSX
                  DO 6105 L1=1,3
                     VG(NG,L1,K) = 0.
                     STRACT(NG,L1,K) = FA(NG,K)
6105              CONTINUE
                  GO TO 6154
C
C        SURFACE TRACTION LIMIT SOMEWHERE WITHIN GEAR
C
6110              IF(VTEMP(NG,1) .EQ. VTEMP(NG,5)) GO TO 6120
                     FX = FMU(ISURF)*GCWP*COSX
                     FORMX(K) = FX
                     L = 1
                     DO 6111 L1=1,5,2
                        VGTEMP(NG,L,K) = VGV(NG,L1)
                        L = L + 1
6111                 CONTINUE
                      CALL VELFOR(
     +               FX      ,FATEMP ,FBTEMP , FCTEMP ,FORMX
```

```
     +                    ,K        ,NGR      ,VX       ,VFMAX     ,VGTEMP )
C
C       RESET MINIMUM SPEED IN GEAR TO THE POINT THAT OCCURS
C       AT THE SURFACE TRACTION LIMIT
C
                          VTEMP(NG,1) = VX
                          FTEMP(NG,1) = FX/ECF
                          XINT = (VTEMP(NG,5) - VTEMP(NG,1))/4.
                          DO 6115 L=2,4
                             VTEMP(NG,L) = VTEMP(NG,L-1) + XINT
                             FTEMP(NG,L) = CTF(NG)*VTEMP(NG,L)**2 +
     +                                     BTF(NG)*VTEMP(NG,L) +
     +                                     ATF(NG)
6115                      CONTINUE
                          GO TO 6130
C
C       SPEEDS EQUAL, INCREMENT ON FORCE
C
6120                   FTEMP(NG,1) = FMU(ISURF)*GCWP*COSX
                       YINT = (FTEMP(NG,1) - FTEMP(NG,5))/4.
                       DO 6125 L=2,4
                          FTEMP(NG,L) = FTEMP(NG,L-1) - YINT
6125                   CONTINUE
C
C       1B.2.2 - COMPUTE VELOCITY DEPENDENT RESISTANCES
C
6130                   DO 6140 L=1,5
C
C       1B.2.2.1 - AERODYNAMIC DRAG
C
                          FCC = 0.
                          FAD = .0026*ACD*PFA*VTEMP(NG,L)*VTEMP(NG,L)/
     +                          144./17.6/17.6
C
                          DO 6135 I=1,NAMBLY
                             IF(NVEH(I) .LT. 1) GO TO 6135
C
C       1B.2.2.2 - TURNING RESISTANCE (WHEELED)
C
                                F1 = ((WGHT(I)*COSX*VTEMP(NG,L)**2)/
     +                                (111.1*RADC))*(12./17.6/17.6)
                                FCC = (FE*F1*F1)/(FLOAT(NWHL(I))*AVGC) +
     +                                FCC
6135                      CONTINUE
                          FV = FAD + FCC
                          FTEMPO(NG,L) = FTEMP(NG,L) - FV
                          VTEMPO(NG,L) = VTEMP(NG,L)
6140                   CONTINUE
                       IF( VGV(NG,1) .NE. VGV(NG,5)) GO TO 6141
                          IGAF = 1
                          GO TO 6152
6141                   IF((NG .NE. 1) .OR.
```

```
     +                   (FTEMP(NG,1) .NE. FTEMP(NG,5))) GO TO 6145
                          FA(NG,K) = FTEMPO(NG,1)
                          FB(NG,K) = 0.
                          FC(NG,K) = 0.
                          GO TO 6152
6145                  DO 6150 L=1,5
                         FQUAD(L) = FTEMPO(NG,L)
                         VQUAD(L) = VTEMPO(NG,L)
6150                  CONTINUE
                      CALL QUAD5(VQUAD,FQUAD,A0,B1,C2)
                      FA(NG,K) = A0
                      FB(NG,K) = B1
                      FC(NG,K) = C2
6152                  L        = 1
                      DO 6153 L1=1,5,2
                         VG(NG,L,K)       = VTEMPO(NG,L1)
                         STRACT(NG,L,K)   = FC(NG,K)*VG(NG,L,K)**2 +
     +                                      FB(NG,K)*VG(NG,L,K) +FA(NG,K)
                         L=L+1
6153                  CONTINUE
6154           CONTINUE
               VFMAX(K) = VG(NG1,1,K)
               FORMX(K) = STRACT(NG1,1,K)
6155        CONTINUE
C
C        1C - NON-VELOCITY DEPENDENT RESISTANCES
C
C        1C.1 - DRAG FORCE FOR TANDEM ALIGNING (WHEELED)
C
            NSPACE = NAMBLY - 1
            DO 6160 I=1,NSPACE
               IF(NVEH(I) .LT. 1) GO TO 6160
                  IF(IT(I) .NE. 1) GO TO 6160
                     WTSPA = WTSPA + (WGHT(I) + WGHT(I+1))*AXLSP(I)
6160        CONTINUE
            FTC = FE*(0.5*FML(ISURF)*WTSPA/RADC)
C
C        1C.2 - TURNING RESISTANCE (TRACKED)
C
6161        DO 6165 I=1,NAMBLY
               IF((NVEH(I) .GT. 0) .OR. (RADC/12. .GT. 305.)) GO TO 6165
                  ALPHA  = WT(I)/TRAKLN(I)
                  MERITK = 1.1624 - 0.6999*ALPHA +
     +                     0.051848*ALPHA*ALPHA +
     +                     0.054848*ALPHA*ALPHA*ALPHA
                  XK1    = 1.18 - 0.0090895*RADC/12. +
     +                     0.00003779*(RADC/12.)**2 -
     +                     (6.70476E-08)*(RADC/12.)**3
                  K1     = MERITK*XK1
                  IF(ITUT .NE. 14) GO TO 6164
C
C        1C.2.1 - TURNING RESISTANCE SOFT SURFACE (TRACKED)
```

```
C
                    FCTRAK = K1*(TFOR/GCW)*WGHT(I) + FCC
                    GO TO 6185
C
C         1C.2.2 - TURNING RESISTANCE HARD SURFACE (TRACKED)
C
6164            FCTRAK = K1*FMU(ISURF)*WGHT(I) + FCC
6165        CONTINUE
C
C         1C.3 - ROLLING RESISTANCE
C
            IF(LTUT .NE. 14) GO TO 6170
C
C         1C.3.1 - SOFT SURFACE (TRAILS)
C
            FR = (RTOWP*GCWP + RTOWNP*GCWNP)*COS(THETA)*SURFF
            GO TO 6185
C
C         1C.3.2.1 - HARD SURFACE (WHEELED)
C
6170        DO 6178 I=1,NAMBLY
               IF(NVEH(I) .LT. 1) GO TO 6175
                  F0 = .007 + .0939/TPSI(I,JPSI)
                  FR = F0*WGHT(I)*SURFF + FR
                  GO TO 6178
C
C         1C.3.2.2 - HARD SURFACE (TRACKED)
C
6175           FR = .045*WGHT(I)*SURFF + FR
6178        CONTINUE
C
C         1C.4 - GRADE RESISTANCE
C
6185        DO 6190 K=1,NTRAV+2
               FGRADE(K) = GCW*SIN(THETA(K))
6190        CONTINUE
C
C         1C.5 - TOTAL NON-VELOCITY DEPENDENT RESISTANCES
C
            DO 6199 K=1,NTRAV+2
               RESIST = FGRADE(K) + FR + FTC + FCTRAK
C
C          D - SPEED LIMITED BY TOTAL RESISTANCES
C
               IF(IGAP .NE. 1) GO TO 6198
                  DO 6197 NG=1,NGR
                     IF(VGV(NG,1) .NE. VGV(NG,5)) GO TO 6197
                        IF((FTEMPO(NG,1) .LT. RESIST) .OR.
     +                     (FTEMPO(NG,5) .GT. RESIST)) GO TO 6197
                           VPOWER(K) = VTEMPO(NG,1)
                           GO TO 6199
6197              CONTINUE
```

```
6198          CALL VELFCR(
     +       RESIST ,FA      ,FB      ,FC      ,FORMX
     +      ,K      ,NGR     ,VEL     ,VFMAX   ,VG    )
              VPOWER(K) = VEL
6199        CONTINUE
C
C        ------------------------------------------------
C        ROUTINE 2 - SPEED LIMITED BY SURFACE ROUGHNESS
C        ------------------------------------------------
C
6200        CONTINUE
            DO 6210 NR=2,MAXIRR
               IF(ACTRMS .GE. RMS(NR)) GO TO 6210
                  VRID = VRIDE(NR-1,LAC) + (ACTRMS - RMS(NR-1))*
     +                   (VRIDE(NR,LAC) - VRIDE(NR-1,LAC))/
     +                   (RMS(NR) - RMS(NR-1))
                  GO TO 6300
6210        CONTINUE
            VRID = VRIDE(MAXIRR,LAC)
C
C        ------------------------------------------------
C        ROUTINE 3 - SPEED LIMITED BY SLIDING ON CURVES
C        ------------------------------------------------
C
6300        CONTINUE
            TANG = TAN(EANG)
            IF(ITUT .NE. 4) GO TO 6305
C
C        3A.1 - SLIDING ON CURVES SOFT SURFACES (TRAILS)
C
               VSLID = SQRT(385.9*RADC*(TANG + TFOR/GCWP)/
     +                     (1. - (TFOR/GCWP)*TANG))
               GO TO 6400
C
C        3A.2 - SLIDING ON CURVES HARD SURFACES
C                (PRIMARY AND SECONDARY ROADS)
C
6305        VSLID = SQRT(385.9*RADC*(TANG + FMU(ISURF))/
     +                  (1. - FMU(ISURF)*TANG))
C
C        ------------------------------------------------
C        ROUTINE 4 - SPEED LIMITED BY TIPPING ON CURVES
C        ------------------------------------------------
C
6400        CONTINUE
            VTIP = SQRT(385.9*RADC*(WTMAX + CGH*TANG)/
     +                 (CGH - WTMAX*TANG))
C
C        ------------------------------
C        ROUTINE 5 - TOTAL BRAKING FORCE
C        ------------------------------
C
```

```
6500       CONTINUE
           IF(ITUT .NE. 14) GO TO 6510
C
C        5A - TOTAL BRAKING SOFT SURFACES
C
            CALL IV11(
     +     DCWB    ,GCW     ,GCWB    ,GCWNB   ,NOGOBF  ,NTRAV   ,
     +     RTOWB   ,RTOWNB  ,TBF     ,THETA   ,WRATIO  ,XBR     )
           GO TO 6600
C
C        5B - TOTAL BRAKING HARD SURFACES
C
6510       TBF(1) = GCW*SIN(THETA(1)) +
     +              AMIN1(XBR,FMU(ISURF)*GCWB*COS(THETA(1)))
           IF(NTRAV .NE. 1) GO TO 6530
              IF(TBF(1) .GE. 0.) GO TO 6520
                 NOGOBF = 1
                 VSEL   = 0.
                 RETURN
6520          NOGOBF = 0
              GO TO 6600
6530       TBF(3) = GCW*SIN(THETA(3)) +
     +              AMIN1(XBR,FMU(ISURF)*GCWB*COS(THETA(3)))
           IF(TBF(3) .GE. 0.) GO TO 6540
              NOGOBF = 1
              VSEL   = 0.
              RETURN
6540       NOGOBF = 0
C
C        ---------------------------------------------------------
C        ROUTINE 6 - DRIVER DICTATED BRAKING LIMITS
C        ---------------------------------------------------------
C
6600       CONTINUE
           BFMX(1) = AMIN1(DCLMAX*GCW,TBF(1)*SFTYPC/100.)
           IF(NTRAV .EQ. 1) GO TO 6700
              BFMX(3) = AMIN1(DCLMAX*GCW,TBF(3)*SFTYPC/100.)
C
C        -----------------------------------------------------
C        ROUTINE 7 - SPEED LIMITED BY VISIBILITY
C        -----------------------------------------------------
C
6700       CONTINUE
            CALL IV13(
     +     BFMX    ,EYEHGT  ,GCW     ,NTRAV   ,
     +     RD      ,REACT   ,VELV    ,VBRAKE  )
C
C        ---------------------------------------------------------
C        ROUTINE 8 - AASHO CURVATURE SPEED LIMIT
C                    ADJUSTED FOR SLIPPERY SURFACES
C        ---------------------------------------------------------
C
```

```
6800      CONTINUE
          IF(RADC/12. .LT. RCURV(NVASHO)) GO TO 6815
             IF(RADC/12. .GT. RCURV(1)) GO TO 6825
                DO 6810 NV=2,NVASHO
                   IF(RADC/12. .LT. RCURV(NV)) GO TO 6810
                      VAASHO = (VCURV(IROAD,NV) + (RADC/12.-RCURV(NV))*
     +                         (VCURV(IROAD,NV-1) - VCURV(IROAD,NV))/
     +                         (RCURV(NV-1) - RCURV(NV)))*17.6
                      GO TO 6830
6810            CONTINUE
                GO TO 6830
6815      VAASHO = (VCURV(IROAD,NVASHO) + (RADC/12. - RCURV(NVASHO))*
     +             (VCURV(IROAD,NVASHO) - VCURV(IROAD,NVASHO-1))/
     +             (RCURV(NVASHO) - RCURV(NVASHO-1)))*17.6
          GO TO 6830
6825         VAASHO = (VCURV(IROAD,1) + (RADC/12. - RCURV(1))*
     +                (VCURV(IROAD,1) - VCURV(IROAD,2))/
     +                (RCURV(1) - RCURV(2)))*17.6
6830      IF(((ITUT .NE. 14) .AND. (FMU(ISURF) .GE. .7)) .OR.
     +       ((ITUT .EQ. 14) .AND. (TFOR/GCWP .GE. .7))) GO TO 6900
               IF((ITUT .NE. 14) .AND. (FMU(ISURF) .LT. .7)) GO TO 6840
C
C        8A - SOFT SURFACES
C
                   VAASHC = VAASHO*SQRT((TFOR/GCWP)/0.7)
                   GO TO 6900
C
C        8B - HARD SURFACES
C
6840            VAASHC = VAASHO*SQRT(FMU(ISURF)/0.7)
C
C        ----------------------------------------
C        ROUTINE 9 - MAXIMUM ROADWAY SPEED
C        ----------------------------------------
6900      CONTINUE
          VSEL = 0.
          DO 6950 K=1,NTRAV,2
             IF(NVEHC .LE. 0) VTIRE(JPSI) = 1760.
             VROAD(K) = AMIN1(VAASHO,VELV(K),VRID,VSLID,VTIP)
             VSEL1(K) = AMIN1(VLIM,VPOWER(K),VROAD(K),VTIRE(JPSI))
             IF(NTRAV .NE. 1) GO TO 6950
                VSEL  = VSEL1(UP)
                RETURN
6950      CONTINUE
          IF((VSEL1(UP) .EQ. 0.) .OR. (VSEL1(DOWN) .EQ. 0.)) GO TO 7000
             VSEL = 2./((1./VSEL1(UP)) + (1./VSEL1(DOWN)))
7000      CONTINUE
          RETURN
             END
      SUBROUTINE BUFF0
C
C        ---------------
```

```
C      OUTPUT ALGORITHM
C      ----------------
C
C      1. LABLED COMMON ASSIGNMENTS
          COMMON /IO/         IEOF
          COMMON /IO/         KBUFF
          COMMON /IO/         LUN1
          COMMON /IO/         LUN2
          COMMON /IO/         LUN3
          COMMON /IO/         LUN4
          COMMON /IO/         LUN5
          COMMON /IO/         LUN6
          COMMON /IO/         LUN7
          COMMON /IO/         LUN8
          COMMON /IO/         LUN9
          COMMON /IO/         LUN10
          INTEGER             MD
          COMMON /INDEX/      MD
          INTEGER             COWN
          COMMON /INDEX/      COWN
          INTEGER             EOF
          COMMON /INDEX/      EOF
          INTEGER             FORCE
          COMMON /INDEX/      FORCE
          INTEGER             GR
          COMMON /INDEX/      GR
          COMMON /INDEX/      LEVEL
          INTEGER             MN
          COMMON /INDEX/      MN
          INTEGER             RPM
          COMMON /INDEX/      RPM
          INTEGER             SPEED
          COMMON /INDEX/      SPEED
          INTEGER             SR
          COMMON /INDEX/      SR
          INTEGER             TR
          COMMON /INDEX/      TR
          INTEGER             TORQUE
          COMMON /INDEX/      TORQUE
          INTEGER             UR
          COMMON /INDEX/      UR
          INTEGER             MX
          COMMON /INDEX/      MX
          COMMON /DERIVE/     ADT        (9)
          COMMON /DERIVE/     NEGOBF
          COMMON /DERIVE/     CAREA
          COMMON /DERIVE/     COWB
          COMMON /DERIVE/     COWP
          COMMON /DERIVE/     COWPB      (20)
          COMMON /DERIVE/     FA         (20,3)
          COMMON /DERIVE/     FAT        (9)
          COMMON /DERIVE/     FAT1       (9)
```

```
COMMON /DERIVE/   FB        (20,3)
COMMON /DERIVE/   FC        (20,3)
COMMON /DERIVE/   FNT       (9)
COMMON /DERIVE/   FOM
COMMON /DERIVE/   FOMMAX
COMMON /DERIVE/   FZRMX     (3)
COMMON /DERIVE/   IFLOAT
COMMON /DERIVE/   IMAX      (3)
COMMON /DERIVE/   ISAFE     (3)
COMMON /DERIVE/   J
COMMON /DERIVE/   MAXI
COMMON /DERIVE/   BEMX      (3)
COMMON /DERIVE/   NEVERO
COMMON /DERIVE/   CBSE
COMMON /DERIVE/   PAV       (9)
COMMON /DERIVE/   RTOWB
COMMON /DERIVE/   RTOWNB
COMMON /DERIVE/   RTOWNP
COMMON /DERIVE/   RTOWP
COMMON /DERIVE/   RTOWPB    (20)
COMMON /DERIVE/   RTOWT     (20)
COMMON /DERIVE/   STRACT    (20,3,3)
COMMON /DERIVE/   SRFO      (9)
COMMON /DERIVE/   SRFV      (9)
COMMON /DERIVE/   STR       (3,9)
COMMON /DERIVE/   TBF       (3)
COMMON /DERIVE/   TDEN      (9)
COMMON /DERIVE/   TRESIS    (3,9)
COMMON /DERIVE/   VA        (3,9)
COMMON /DERIVE/   NOGOVA    (3,9)
COMMON /DERIVE/   VAVOID    (3,9)
COMMON /DERIVE/   VBG       (3,9)
COMMON /DERIVE/   IMPACT    (9)
COMMON /DERIVE/   VELV      (3)
COMMON /DERIVE/   VFMAX     (3)
COMMON /DERIVE/   VE        (20,3,3)
COMMON /DERIVE/   NOGOVO    (3,9)
COMMON /DERIVE/   VMAX
COMMON /DERIVE/   VMAX1     (3)
COMMON /DERIVE/   VMAX2     (3,9)
COMMON /DERIVE/   VOLA
COMMON /DERIVE/   VOVER     (3,9)
COMMON /DERIVE/   VRID
COMMON /DERIVE/   VSEL
COMMON /DERIVE/   VSEL1     (3)
COMMON /DERIVE/   VSEL2     (3,9)
COMMON /DERIVE/   VSOIL     (3,9)
COMMON /DERIVE/   VWT       (3,9)
COMMON /DERIVE/   VXT       (3,9)
COMMON /DERIVE/   NOGONO
COMMON /DERIVE/   WRATIO
COMMON /TERRAN/   AA
```

```
      COMMON /TERRAN/    ACTRMS
      COMMON /TERRAN/    AREA
      COMMON /TERRAN/    AREAC
      COMMON /TERRAN/    CI
      COMMON /TERRAN/    CIST
      COMMON /TERRAN/    EANG
      COMMON /TERRAN/    EOF
      COMMON /TERRAN/    ELEV
      COMMON /TERRAN/    FNU       (3)
      COMMON /TERRAN/    GRADE
      COMMON /TERRAN/    IOBS
      COMMON /TERRAN/    IOST
      COMMON /TERRAN/    IROAD
      COMMON /TERRAN/    IS        (9)
      COMMON /TERRAN/    IST
      COMMON /TERRAN/    IIUT
      COMMON /TERRAN/    NI
      COMMON /TERRAN/    NBU
      COMMON /TERRAN/    CAW
      COMMON /TERRAN/    CBAA
      COMMON /TERRAN/    CBH
      COMMON /TERRAN/    CBL
      COMMON /TERRAN/    CBS
      COMMON /TERRAN/    CBW
      COMMON /TERRAN/    CBMINW
      COMMON /TERRAN/    CDIA
      COMMON /TERRAN/    NWASHO
      COMMON /TERRAN/    RADC
      COMMON /TERRAN/    RCI
      COMMON /TERRAN/    RCIC      (4)
      COMMON /TERRAN/    RCURV     (11)
      COMMON /TERRAN/    RG
      COMMON /TERRAN/    RDA       (12)
      COMMON /TERRAN/    S         (9)
      COMMON /TERRAN/    SD        (9)
      COMMON /TERRAN/    SDL       (9)
      COMMON /TERRAN/    SURFF
      COMMON /TERRAN/    TANPHI
      COMMON /TERRAN/    THETA     (3)
      COMMON /TERRAN/    VCURV     (4,11)
      COMMON /TERRAN/    WA
      COMMON /TERRAN/    WE
C     ----------------------
C     VARIABLE DECLARATIONS
C     ----------------------
C
      REAL MPH
      ZMPH1=MPH(VMAX)
      ZMPH2=MPH(VMAX1(UP))
      ZMPH3=MPH(VMAX1(LEVEL))
      ZMPH4=MPH(VMAX1(DOWN))
      ZMPH5=MPH(VSEL)
```

```
      ZMPH6=MPH(VSEL1(UP))
      ZMPH7=MPH(VSEL1(LEVEL))
      ZMPH8=MPH(VSEL1(DOWN))
      IF(ITUT .GE. 11) GO TO 2030
      IF(KBUFF .EQ. 1) GO TO 2020
      KBUFF = 1
      WRITE(LUN1,2000)
2000  FORMAT(70H1 NTU ITUT  VMAX    UP LEVEL  DOWN VSEL    UP LEVEL
     +DOWN GRADE   AREA,//)
C    3. DYNAMIC OUTPUT SECTION
2020  WRITE(LUN10) NTU,ITUT,ZMPH1,ZMPH2,ZMPH3,ZMPH4,ZMPH5,ZMPH6,
     +                ZMPH7,ZMPH8,GRADE,AREA
      WRITE(LUN1,3000) NTU,ITUT,ZMPH1,ZMPH2,ZMPH3,ZMPH4,ZMPH5,ZMPH6,
     +                   ZMPH7,ZMPH8,GRADE,AREA
      GO TO 4000
2030  IF(KBUFF .EQ. 1) GO TO 2040
         KBUFF = 1
         WRITE(LUN1,2900)
2900  FORMAT(42H1 NTU  ITUT   VSEL    UP DOWN GRADE DISTANCE,//)
2040  WRITE(LUN1,2950) NTU,ITUT,ZMPH5,ZMPH6,ZMPH8,GRADE,DIST
2950  FORMAT( 1X,I4,3X,I2,4(1X,F5.2),1X,F8.4)
3000  FORMAT( 1X,I4,4X,I1,8(1X,F5.2), 1X,F5.2,1X,F6.4)
C    3. TERMINUS
4000  CONTINUE
      RETURN
      END
     SUBROUTINE ACCEL
    + (FA      ,FB      ,FC      ,GCW     ,I
    + ,K       ,NGF     ,NGR     ,NV2FLG  ,RMX
    + ,STRACT  ,T       ,TRESIS  ,V1      ,V2
    + ,V2F     ,VG      ,X       ,LUN1 )
C    ------------------------------------------------------------
C    TIME AND DISTANCE TO ACCELERATE FROM ONE VELOCITY TO ANOTHER
C    ------------------------------------------------------------
C
C    1. VARIABLE DECLARATION
      REAL       FA      (20,3)
      REAL       FB      (20,3)
      REAL       FC      (20,3)
      REAL       RMX     (20)
      REAL       STRACT  (20,3,3)
      REAL       TRESIS  (3,9)
      REAL       VG      (20,3,3)
C    3. ALGORITHM
      VM = GCW/385.9
C     A. DETERMINE GEARS (NG1 ,NG2) OF THE INITIAL
C   + AND FINAL VELOCITIES.
         DO 3010 NG=1,NGF
            IF(V1 .LE. VG(NG,3,K)) GO TO 3020
3010     CONTINUE
3020     CONTINUE
         NG1 = NG
```

```
              DO 3030 NG=NG1,NER
                 IF(V2 .LE. VG(NG,3,K)) GO TO 3040
3030          CONTINUE
3040          CONTINUE
              NG2 = NG
              IF(NG1 .NE. NG2) GO TO 3080
C          B. SINGLE GEAR ROUTINE
              VL = V1
              VH = V2
              NG = NG1
              T  = 0.
              X  = 0.
              CALL TXGEAR
     + (FA,FB,FC,GCW,I,K,NG,NV2FLG,RMX,STRACT,TT,TRESIS,VL,VH,VG,XX)
              IF(NV2FLG .NE. 0) GO TO 3050
                 T = TT
                 X = XX
                 GO TO 4000
3050          IF(NV2FLG .NE. 1) GO TO 3060
                 GO TO 3260
3060          IF(NV2FLG .NE. 2) GO TO 3070
                 GO TO 4000
3070          CONTINUE
                 GO TO 4005
C          C. MULTIPLE GEAR ROUTINE
3080          CONTINUE
              T  = 0.
              X  = 0.
              VL = V1
              VH = VG(NG1,3,K)
              CALL TXGEAR
     + (FA,FB,FC,GCW,I,K,NG,NV2FLG,RMX,STRACT,TT,TRESIS,VL,VH,VG,XX)
              IF(NV2FLG .NE. 0) GO TO 3130
                 T  = TT
                 VS = VG(NG1,3,K)
                 X  = XX
                 IF(NG2 .LE. (NG1+1)) GO TO 3090
                    GO TO 3160
3090             CONTINUE
                 VL = VS
                 VH = V2
                 NG = NG2
                 CALL TXGEAR
     + (FA,FB,FC,GCW,I,K,NG,NV2FLG,RMX,STRACT,TT,TRESIS,VL,VH,VG,XX)
                 IF(NV2FLG .NE. 0) GO TO 3100
                    T = T+TT
                    X = X+XX
                    GO TO 4000
3100             IF(NV2FLG .NE. 1) GO TO 3110
                    VF2    = VG(NG1,3,K)
                    NGF    = NG1
                    NV2FLG = 2
```

```
                  GO TO 4000
3110          IF(NV2FLG .NE. 2) GO TO 3120
                  GO TO 3260
3120          CONTINUE
              GO TO 4005
3130      IF(NV2FLG .NE. 1) GO TO 3140
              GO TO 4000
3140      IF(NV2FLG .NE. 2) GO TO 3150
              GO TO 3260
3150      CONTINUE
              GO TO 4005
C       D. ACCELERATE THROUGH INTERMEDIATE GEARS
3160      CONTINUE
          M1 = NG1+1
          M2 = NG2-1
          DO 3200 NG=M1,M2
              VL = VS
              VH = VG(NG,3,K)
              CALL TXGEAR
     + (FA,FB,FC,GCW,I,K,NG,NV2FLG,RMX,STRACT,TT,TRESIS,VL,VH,VG,XX)
              IF(NV2FLG .NE. 0) GO TO 3170
                  T  = T + TT
                  VS = VG(NG,3,K)
                  X  = X + XX
                  GO TO 3200
3170          IF(NV2FLG .NE. 1) GO TO 3180
                  GO TO 3210
3180          IF(NV2FLG .NE. 2) GO TO 3190
                  GO TO 3260
3190          CONTINUE
              GO TO 4005
3200      CONTINUE
          GO TO 3220
3210      CONTINUE
              NGF    = NG-1
              V2F    = VG(NGF,3,K)
              NV2FLG = 2
              GO TO 4000
3220      CONTINUE
              NG = NG2
              VL = VS
              VH = V2
              CALL TXGEAR
     + (FA,FB,FC,GCW,I,K,NG,NV2FLG,RMX,STRACT,TT,TRESIS,VL,VH,VG,XX)
              IF(NV2FLG .NE. 0) GO TO 3230
                  T = T+TT
                  X = X+XX
                  GO TO 4000
3230          IF(NV2FLG .NE. 1) GO TO 3240
                  NGF    = NG-1
                  V2F    = VG(NGF,3,K)
                  NV2FLG = 2
```

```
                      GO TO 4000
3240               IF(NV2FLG .NE. 2) GO TO 3250
                      GO TO 3260
3250               CONTINUE
                   GO TO 4005
C         E. ERROR ROUTINE
3260            CONTINUE
                VAV=(VL+VH)/2.
                DO 3300 J=1,4
                   CALL TXGEAR
     + (FA,FB,FC,GCW,I,K,NG,NV2FLG,RMX,STRACT,TT,TRESIS,VL,VAV,VG,XX)
                   IF(NV2FLG .NE. 0) GO TO 3270
                      VH  = VAV
                      VAV = (VL+VH)/2.
                      GO TO 3300
3270               IF(NV2FLG .NE. 1) GO TO 3280
                      VAV = (VAV+VH)/2.
                      GO TO 3300
3280               IF(NV2FLG .NE. 2) GO TO 3290
                      VH  = VAV
                      VAV = (VL+VH)/2.
                      GO TO 3300
3290               CONTINUE
                   GO TO 4005
3300            CONTINUE
                V2F     = VAV
                NGF     = NG
                NV2FLG  = 2
                T       = T+TT
                X       = X+XX
C      4. TERMINUS
4000       CONTINUE
           RETURN
4005       WRITE(LUN1,4010)
       WRITE(LUN1,4020)
       WRITE(LUN1,4030) (FA(NG,K),NG=1,NGR)
       WRITE(LUN1,4040)
       WRITE(LUN1,4030) (FB(NG,K),NG=1,NGR)
       WRITE(LUN1,4050)
       WRITE(LUN1,4030) (FC(NG,K),NG=1,NGR)
       WRITE(LUN1,4060) GCW,K,NGR,NV2FLG
       WRITE(LUN1,4070)
       WRITE(LUN1,4030) ((STRACT(NG,M,K),M=1,3),NG=1,NGR)
       WRITE(LUN1,4080) T       ,TRESIS(K,I),V1  ,V2     ,V2F
     +                  ,X
4010   FORMAT(1H1,6H$ACCEL,/)
4020   FORMAT(/,1X,8HFA       =)
4030   FORMAT(9X,5(E15.8,5X))
4040   FORMAT(/,1X,8HFB       =)
4050   FORMAT(/,1X,8HFC       =)
4060   FORMAT(/,1X,8HGCW      =,E15.8,/,1X,8HK       =,I4,/,1X,8HNGR     =,
     +    I4,/,1X,8HNV2FLG =,I4)
```

```
4070  FORMAT(/,1X,8HSTRACT =)
4080  FORMAT(/,1X,8HT        =,E15.8,/,1X,8HTRES S  =,E15.8,/,
     +          1X,8HV1      =,E15.8,/,1X,8HV2      =,E15.8,/,
     +          1X,8HV2F     =,E15.8,/,1X,8HX       =,E15.8)
         RETURN
         END
      SUBROUTINE D3LINC
     + (D        ,A        ,I        ,II       ,J
     + ,JJ       ,K        ,KK       ,VALI     ,VALJ
     + ,VALK     ,VI       ,VJ       ,VK )
C     ------------------------------------------------------------
C     THREE-DIMENSIONAL LINEAR INTERPOLATION SUBROUTINE
C     EXTRAPOLATION AT CONSTANT LEVEL BEYOND LAST
C     ELEMENTS OF ARRAY.
C     ------------------------------------------------------------
C
C     1. VARIABLE DECLARATION
         REAL       A       (7,14,5)
         REAL       VALI    (7)
         REAL       VALJ    (14)
         REAL       VALK    (5)
C     2. ALGORITHM
         IF(I .NE. II) GO TO 2010
            ALL = A(I, J, K)
            ALH = A(I, J,KK)
            AHL = A(I,JJ, K)
            AHH = A(I,JJ,KK)
            GO TO 2020
2010     CONTINUE
         DUMMY = (VI-VALI(I))/(VALI(II)-VALI(I))
         ALL   = A( I, J, K)+DUMMY*(A(II, J, K)-A( I, J, K))
         ALH   = A( I, J,KK)+DUMMY*(A(II, J,KK)-A( I, J,KK))
         AHL   = A( I,JJ, K)+DUMMY*(A(II,JJ, K)-A( I,JJ, K))
         AHH   = A( I,JJ,KK)+DUMMY*(A(II,JJ,KK)-A( I,JJ,KK))
2020     IF(J .NE. JJ) GO TO 2030
            AL = ALL
            AH = ALH
            GO TO 2040
2030     CONTINUE
         DUMMY = (VJ-VALJ(J))/(VALJ(JJ)-VALJ(J))
         AL    = ALL+DUMMY*(AHL-ALL)
         AH    = ALH+DUMMY*(AHH-ALH)
2040     IF(K .NE. KK) GO TO 2050
            D = AL
            GO TO 3000
2050     CONTINUE
         D = AL+(AH-AL)*(VK-VALK(K))/(VALK(KK)-VALK(K))
C     3. TERMINUS
3000     CONTINUE
         RETURN
         END
      SUBROUTINE FGSPC
```

```
     + (CPF     ,D        ,NVEH    ,RCIX)
C       ------------------------------------------------
C       FINE GRAINED SOIL FULL COEFFICIENT ALGORITHM
C       ------------------------------------------------
C
C       1. ALGORITHM
           IF(NVEH .EQ. 1) GO TO 1040
C              TRACKED ASSEMBLY
               IF(CPF .GE. 4.) GO TO 1020
                  IF(RCIX .LT. 0.) GO TO 1010
                     D =
     + .544+.0463*RCIX-((.544+.0463*RCIX)**2.-.0702*RCIX)**.5
                     GO TO 2000
1010              CONTINUE
C                 D = .056*RCIX
                  D = .076*RCIX
                  GO TO 2000
1020           IF(RCIX .LT. 0.) GO TO 1030
                  D =
     + .4554+.0392*RCIX-((.4554+.0392*RCIX)**2-0.0526*RCIX)**0.5
                  GO TO 2000
1030           CONTINUE
               D = .056*RCIX
               GO TO 2000
1040       CONTINUE
C          WHEELED ASSEMBLY
           IF(CPF .GE. 4.) GO TO 1060
               IF(RCIX .LT. 0.) GO TO 1050
                  D =
     + .3885-.0265*RCIX-((.3885+.0265*RCIX)**2-.0358*RCIX)**.5
                  GO TO 2000
1050           CONTINUE
               D = .046*RCIX
               GO TO 2000
1060       IF(RCIX .LT. 0.) GO TO 1070
               D =
     + .379+.0219*RCIX-((.379+.0219*RCIX)**2-.0257*RCIX)**.5
               GO TO 2000
1070       CONTINUE
           D = .033*RCIX
C       2. TERMINUS
2000       CONTINUE
           RETURN
           END
        SUBROUTINE FGSPR
     + (CPF     ,NVEH    ,RCIX    ,RTOWPB)
C       ------------------------------------------------------------
C       FINE GRAINED SOIL POWERED ASSEMBLY MOTION RESISTANCE
C       ------------------------------------------------------------
C
C       1. ALGORITHM
           IF(NVEH .EQ. 1) GO TO 1030
```

```
C            TRACKED ASSEMBLY
             IF(RCIX .LT. 0.) GO TO 1010
                RTOWPB = .045+2.3075/(RCIX+6.5)
                GO TO 2000
1010         IF(CPF .GE. 4.) GO TO 1020
                RTOWPB = .4-.072*RCIX
                GO TO 2000
1020         CONTINUE
             RTOWPB = .4-.052*RCIX
             GO TO 2000
C         WHEELED ASSEMBLY
1030      IF(RCIX .LT. 0.) GO TO 1050
             IF(CPF .GE. 4.) GO TO 1040
                RTOWPB = .035+.861/(RCIX+3.249)
                GO TO 2000
1040         CONTINUE
             RTOWPB = .045+2.3075/(RCIX+6.5)
             GO TO 2000
1050      IF(CPF .GE. 4.) GO TO 1060
             RTOWPB = .3-.043*RCIX
             GO TO 2000
1060      CONTINUE
          RTOWPB = .4-.029*RCIX
C      2. TERMINUS
2000      CONTINUE
          RETURN
          END
       SUBROUTINE FGSTR
     + (DIAW    ,DRAT    ,IB      ,IP      ,NVEH
     + ,NWHL    ,RCI     ,RTOWT   ,SECTW   ,WGHT
     + ,WRATIO  ,LUN1  )
C      ------------------------------------------------
C      FINE GRAINED SOIL TOWED MOTION RESISTANCE
C      ------------------------------------------------
C      2. ALGORITHM
          IF((IP .EQ. 0) .AND. (IB .EQ. 0)) GO TO 2010
C            NEVER TOWED ASSEMBLY
             RTOWT = 0.
             GO TO 3000
2010      CONTINUE
C         TOWED ASSEMBLY
          IF(NVEH .NE. 0) GO TO 2020
C            TRACKED ASSEMBLY
       WRITE(LUN1,3010)
       WRITE(LUN1,3020) DIAW    ,DRAT    ,IB      ,IP      ,NVEH
     +                  ,NWHL    ,RCI     ,RTOWT   ,SECTW   ,WGHT
     +                  ,WRATIO
3010   FORMAT(1H1,6H$FGSTR,/)
3020   FORMAT(/,1X,8HDIAW    =,E15.8,/,1X,8HDRAT    =,E15.8,/,
     +          1X,8HIB      =,I5   ,/,1X,8HIP      =,I5,/,
     +          1X,8HNVEH    =,I5   ,/,1X,8HNWHL    =,I5,/,
     +          1X,8HRCI     =,E15.8,/,1X,8HRTOWT   =,E15.8,/,
```

```
     +          1X,8HSECTW  =,E15.8,/,1X,8HWGHT    =,E15.8,/,
     +          1X,8HWRATIO =,E15.8)
                STOP
2020      CONTINUE
C         WHEELED ASSEMBLY
         WPW   = WGHT*WRATIO/FLOAT(NWHL)
          BETA =
     + (RCI*SECTW*DIAW*DRAT**.5)/(WPW*(1.+SECTW/2./DIAW))
          IF(BETA .GT. 2.) GO TO 2030
             RTOWT = 1.-.3412*BETA
             GO TO 3000
2030      CONTINUE
          RTOWT = .04+.2/(BETA-1.35)
          GO TO 3000
C      3. TERMINUS
3000      CONTINUE
          RETURN
          END
       SUBROUTINE FORVEL
     + (F         ,FA        ,FB        ,FC        ,FORMX
     + ,K         ,NGR       ,V         ,VFMAX     ,VG )
C      -----------------------------------------------
C      FORCE AVALIABLE AT A GIVEN VELOCITY
C      -----------------------------------------------
C
C      1. VARIABLE DECLARATION
          REAL       FA        (20,3)
          REAL       FB        (20,3)
          REAL       FC        (20,3)
          REAL       FORMX     (3)
          REAL       VFMAX     (3)
          REAL       VG        (20,3,3)
C      2. ALGORITHM
          IF( V .GE. VFMAX(K)) GO TO 2010
             F = FORMX(K)
             GO TO 3000
2010      CONTINUE
          DO 2030 NG=1,NGR
             IF(V .GT. VG(NG,3,K)) GO TO 2020
                F = (FC(NG,K)*V+FB(NG,K))*V+FA(NG,K)
                GO TO 3000
2020         CONTINUE
2030      CONTINUE
          F = 0.
C      3. TERMINUS
3000      CONTINUE
          RETURN
          END
       SUBROUTINE QUAD
     +   (A          ,B          ,C
     +   ,X1         ,X2         ,X3
     +   ,Y1         ,Y2         ,Y3
```

```
     +   ,LUN1 )
C         ----------------------------------------------------
C         QUADRATIC FIT THROUGH THREE POINTS SUBROUTINE
C         ----------------------------------------------------
C
C         1. ALGORITHM
             AA = (Y2-Y1)/(X2-X1)
             BB = (Y3-Y1)/(X3-X1)
             CC = (BB-AA)/(X3-X2)
             A  = Y1-AA*X1+CC*X1*X2
             B  = AA-CC*(X1+X2)
             C  = CC
C         2. CHECK TO INSURE THAT THERE IS NO DIVISION BY ZERO (0)
               IF((X3-X1) .EQ. 0. .OR. (X3-X2) .EQ. 0. .OR.
     +         (X2-X1) .EQ. 0.) GO TO 3900
               GO TO 4000
3900           WRITE(LUN1,3910)
3910      FORMAT(1H1,39HDIVISION BY ZERO (0) IN SUBROUTINE QUAD./)
C         3. TERMINUS
4000         RETURN
             END
          SUBROUTINE QUAD5(X ,Y ,A ,B ,C)
C
C         ----------------------------------------------------
C         LEAST SQUARES FIT THROUGH FIVE POINTS PASSING
C         THROUGH THE FIRST AND LAST POINTS. EQUATION
C         OF THE FORM: Y= A + B*X + C*X**2
C         ----------------------------------------------------
C
C         1. VARIABLE DECLARATIONS
                REAL X     (5)
                REAL Y     (5)
                REAL A
                REAL B
                REAL C
C
C         --------------------------------------
C         TRANSFORM ARRAY INTO THE UNIT SQUARE
C         --------------------------------------
C
          CX=X(5) - X(1)
          CY=Y(5) - Y(1)
          XXSUM=0.
          XYSUM=0.
          DO 2000 I=1,5
             XTI=(X(I) - X(1))/CX
             YTI=Y(I) - Y(1)
             XYSUM=XYSUM + YTI - XTI*CY
             XXSUM=XXSUM + XTI*(XTI - 1.)
2000      CONTINUE
C
C         ---------------------------
```

```
C     UNIT SQUARE COEFFICIENTS
C     ------------------------
C
      C3=XYSUM/XXSUM
      C2=CY - C3
C
C     ----------------------------------------
C     INTERMEDIATE TERM (NOT A COEFFICIENT)
C     ----------------------------------------
C
      C4=1./(CX*CX)
C
C     -------------------------
C     FIVE POINT FIT COEFFICIENTS
C     -------------------------
C
      A=Y(1) + X(1)*C4*(X(1)*C3 - CX*C2)
      B=C4*(CX*C2 - 2.*X(1)*C3)
      C=C4*C3
      RETURN
      END
      SUBROUTINE SLIP
     + (CPFC    ,IST     ,LOCDIF ,NFL     ,NVEHC
     + ,S       ,Y       ,LUN1 )
C     --------------
C     SLIP ALGORITHM
C     --------------
C     2. ALGORITHM
         IF((IST .NE. 1) .AND. (IST .NE. 6)) GO TO 2060
C        A. FINE GRAINED SOIL
            IF(NVEHC .EQ. 1) GO TO 2020
C              TRACKED
               IF(CPFC .GE. 4.) GO TO 2010
                  S = .0257*Y-.0161+.01519/(.8353-Y)
                  GO TO 2160
2010           CONTINUE
               S = .0733*Y-.0063+.00734/(.7177-Y)
               GO TO 2160
C           WHEELED
2020        IF(CPFC .GE. 4.) GO TO 2040
               S = .0621*Y-.021+.01888/(.7794-Y)
               IF(LOCDIF .NE. 1) GO TO 2030
                  S = S/1.1
2030           CONTINUE
               GO TO 2160
2040        CONTINUE
            S = .084*Y-.016+.01414/(.6697-Y)
            IF(LOCDIF .NE. 1) GO TO 2050
               S = S/1.1
2050        CONTINUE
            GO TO 2160
2060     IF( IST .NE. 2) GO TO 2100
```

```
C         B. COARSE GRAINED SOIL
             IF(NVEHC .EQ. 1) GO TO 2080
C               TRACKED
                IF(NFL .EQ. 1) GO TO 2070
C                  RIGID TRACK
                   S = -.0083+.005312/(.573-Y)
                   GO TO 2160
2070            CONTINUE
C               FLEXIBLE TRACK
                YY = 1.074*Y-.72
                S  = YY+((YY**2)+.09*Y+.009)**.5
                GO TO 2160
2080         CONTINUE
C            WHEELED
             S = .0074*Y-.0061+.00374/(.5786-Y)
             IF(LOCDIF .NE. 1) GO TO 2090
                S = S/1.1
2090         CONTINUE
             GO TO 2160
2100      IF(IST .NE. 3) GO TO 2130
C         C. MUSKEG
             IF((NVEHC .EQ. 1) .OR. (CPFC .GE. 4.)) GO TO 2110
                S = .0585*Y-.0106+.01336/(.964-Y)
                GO TO 2160
2110         CONTINUE
             S = .1024*Y-.00864+.01062/(.7564-Y)
             IF((NVEHC .NE. 1) .OR. (LOCDIF .NE. 1)) GO TO 2120
                S = S/1.1
2120         CONTINUE
             GO TO 2160
2130      IF(IST .NE. 4) GO TO 2150
C         D. SHALLOW SNOW
             IF(Y .LT. 1.) GO TO 2140
                S = 1.
                GO TO 2160
2140         CONTINUE
             S = .3*(1.-(1.-Y)**.5)
                GO TO 2160
C         E. ERROR
 2150     WRITE(LUN1,2200)
        WRITE(LUN1,2210) CPFC    ,IST     ,LOCDIF ,NFL     ,NVEHC
     +                    ,S      ,Y
2200   FORMAT(1H1,6H$STICK,/)
2210   FORMAT(/,1X,8HCPFC     =,E15.8,/,1X,8HIST      =,I5    ,/,
     +          1X,8HLOCDIF   =,I5    ,/,1X,8HNFL      =,I5    ,/,
     +          1X,8HNVEHC    =,I5    ,/,1X,8HS        =,E15.8,/,
     +          1X,8HY        =,E15.8)
             STOP
C         F. FUNCTION LIMITS
2160         CONTINUE
             IF((S .GE. 0.) .AND. (S .LE. 1.)) GO TO 2170
                S = 1.
```

```
2170        CONTINUE
C        3. TERMINUS
            RETURN
         END
      SUBROUTINE TFORCF
     + (CF      ,CPFC    ,DCWP    ,GCWP    ,IST
     + ,NFL     ,NVEHC   ,RTOWP   ,TFOR    ,LUN1 )
C     ------------------------------------------
C     SOIL LIMITED TRACTIVE EFFORT ALGORITHM
C     ------------------------------------------
C     2. ALGORITHM
         IF((IST .NE. 1) .AND. (IST .NE. 6)) GO TO 2040
C        A. FINE GRAINED
            IF(NVEHC .EQ. 1) GO TO 2020
C              TRACKED
               IF(CPFC .GE. 4.) GO TO 2010
                  CF   = (DCWP-.758)+RTOWP
                  TFOR = (CF+.82)*GCWP
                  GO TO 3000
2010           CONTINUE
               CF   = (DCWP-.671)+RTOWP
               TFOR = (CF+.72)*GCWP
               GO TO 3000
2020        CONTINUE
C           WHEELED
            IF(CPFC .GE. 4.) GO TO 2030
               CF   = (DCWP-.674)+RTOWP
               TFOR = (CF+.78)*GCWP
               GO TO 3000
2030        CONTINUE
            CF   = (DCWP-.585)+RTOWP
            TFOR = (CF+.655)*GCWP
            GO TO 3000
2040     IF(IST .NE. 2) GO TO 2070
C        B. COARSE GRAINED
            IF(NVEHC .EQ. 1) GO TO 2060
C              TRACKED
               IF(NFL .EQ. 1) GO TO 2050
C                 RIGID TRACK
                  CF   = .076
                  TFOR = (CF+.568)*GCWP
                  GO TO 3000
2050           CONTINUE
C              FLEX TRACK
               CF   = .1
               TFOR = (CF+.695)*GCWP
               GO TO 3000
2060        CONTINUE
C           WHEELED
            CF   = RTOWP+DCWP-.56
            TFOR = (CF+.575)*GCWP
            GO TO 3000
```

```
2070      IF( IST .NE. 3) GO TO 2090
C         C. MUSKEG
             IF((NVEHC .NE. 0) .OR. (CPFC .GE. 4.)) GO TO 2080
                CF    = RTOWP*DOWP-.88
                TFOR = (CF+.92)*GCWP
                GO TO 3000
2080         CONTINUE
             CF    = RTOWP*DOWP-.68
             TFOR = (CF+.745)*GCWP
             GO TO 3000
2090      CONTINUE
C         D. ERROR
      WRITE( LUN1,2100)
      WRITE( LUN1,2110) CF      ,CPFC    ,DOWP    ,GCWP    ,IST
     +                 ,NFL     ,NVEHC   ,RTOWP   ,TFOR
2100  FORMAT(1H1,7H$TFCRCF,/)
2110  FORMAT(/,1X,8HCF      =,E15.8,/,1X,8HCPFC    =,E15.8,/,
     +          1X,8HDOWP    =,E15.8,/,1X,8HGCWP    =,E15.8,/,
     +          1X,8HIST     =,I5    ,/,1X,8HNFL     =,I5    ,/,
     +          1X,8HNVEHC   =,I5    ,/,1X,8HRTOWP   =,E15.8,/,
     +          1X,8HTFOR    =,E15.8)
             STOP 12
C      3. TERMINUS
3000      CONTINUE
          RETURN
          END
      SUBROUTINE TXGEAR
     + (FA      ,FB      ,FC      ,GCW     ,I
     + ,K       ,NG      ,NV2FLG  ,RMX     ,STRACT  ,T
     + ,TRESIS  ,V1      ,V2      ,VG      ,X )
C     ---------------------------------
C     TIME AND DISTANCE IN A GEAR
C     ---------------------------------
C
C     1. VARIABLE DECLARATION
         REAL        FA      (20,3)
         REAL        FB      (20,3)
         REAL        FC      (20,3)
         REAL        RMX     (20)
         REAL        STRACT  (20,3,3)
         REAL        TRESIS  (3,9)
         REAL        VG      (20,3,3)
C     2. ALGORITHM
C        A. SET COMMON VALUES
             VM      = (GCW/385.9)*RMX(NG)
             A       = FA(NG,K)
             B       = FB(NG,K)
             C       = FC(NG,K)
             F       = TRESIS(K,I)
             CSQ     = B*B-4.*(A-F)*C
             NV2FLG = 0.
C        B. SOLUTION TREE
```

```
      IF(C) 26, 17, 1
C     1. POSITIVE CURVATURE
1        IF(B) 2, 11, 14
C        A. NEGATIVE SLOPE AT V=0.
2           CONTINUE
            IF(A-F) 3,3,4
C          1. NEGATIVE/ZERO EXCESS TRACTION
3             R2=(-B+SQRT(DSQ))/(2.*C)
              R1=(A-F)/(C*R2)
              IF(V1.GT.R2) GO TO  36
              NV2FLG=1
              RETURN
C          2. POSITIVE EXCESS TRACTION
4             IF(DSQ) 38,8,5
C             A. POSITIVE DISCRIMINANT
5                R2=(-B+SQRT(DSQ))/(2.*C)
                 R1=(A-F)/(C*R2)
                 IF(V2.LE.R1.OR.V1.GE.R2) GO TO 36
                 IF(V2-R1) 7,7,6
6                NV2FLG=1
                 RETURN
7                NV2FLG=2
                 RETURN
C             B. ZERO DISCRIMINANT
8                R2=(-B+SQRT(DSQ))/(2.*C)
                 R1=(A-F)/(C*R2)
                 IF(V1.GT.R1.OR.V2.LT.R1) GO TO 37
                 IF(V1-R1) 7,6,7
C             C. NEGATIVE DISCRIMINANT
C        B. ZERO SLOPE AT V=0.
11          IF(A-F) 13,12,38
C           1. ZERO EXCESS TRACTION
12             T=VM*(1./V1-1./V2)/C
               X=(VM/C)*ALOG(VM/(VM-V1*C*T))
               RETURN
C           2. POSITIVE EXCESS TRACTION
13             R2=SQRT(DSQ)/(2.*C)
               R1=(A-F)/(C*R2)
               IF(V1.GT.R2) GO TO 36
               NV2FLG=1
               RETURN
C        C. POSITIVE SLOPE AT V=0.
14          IF(A-F) 15,15,16
C           1. NEGATIVE/ZERO EXCESS TRACTION
15             R1=(-B-SQRT(DSQ))/(2.*C)
               R2=(A-F)/(C*R1)
               IF(V1.GT.R2) GO TO 36
               NV2FLG=1
               RETURN
C           2. POSITIVE EXCESS TRACTION
16             IF(DSQ) 38,37,36
C     2. ZERO CURVATURE
```

```
17          IF(B) 18,22,25
C           A. NEGATIVE SLOPE AT V=0.
18             IF(A-F) 19,19,20
C              1. NEGATIVE/ZERO EXCESS TRACTION
19                NV2FLG=1
                  RETURN
C              2. POSITIVE EXCESS TRACTION
20                R1=-(A-F)/B
                  IF(V1-R1) 21,19,19
21                IF(V2.LT.R1) GO TO 35
                  NV2FLG=2
                  RETURN
C           B. ZERO SLOPE AT V=0.
22             IF(A-F) 23,23,24
C              1. NEGATIVE/ZERO EXCESS TRACTION
23                NV2FLG=1
                  RETURN
C              2. POSITIVE EXCESS TRACTION
24                T=VM*(V2-V1)/(A-F)
                  X=((A-F)*T/(2.*VM)+V1)*T
                  RETURN
C           C. POSITIVE SLOPE AT V=0.
25             IF(A-F.GE.0.) GO TO 35
                  R1=-(A-F)/B
                  IF(V1.GT.R1) GO TO 35
                  NV2FLG=1
                  RETURN
C        3. NEGATIVE CURVATURE
26          IF(B) 27,27,31
C           A. NEGATIVE/ZERO CURVATURE
27             IF(A-F) 28,28,29
C              1. NEGATIVE/ZERO EXCESS TRACTION
28                NV2FLG=1
                  RETURN
C              2. POSITIVE EXCESS TRACTION
29                R1=(-B+SQRT(DSQ))/(2.*C)
                  R2=(A-F)/(C*R1)
                  IF(V1.LT.R2) GO TO 30
                  NV2FLG=1
                  RETURN
30                IF(V2.LE.R2) GO TO 36
                  NV2FLG=3
                  RETURN
C           B. POSITIVE CURVATURE
31             IF(A-F.GE.0..OR.DSQ.GT.0.) GO TO 32
                  NV2FLG=1
                  RETURN
32                R2=(-B-SQRT(DSQ))/(2.*C)
                  R1=(A-F)/(C*R2)
                  IF(V1.LE.R1) GO TO 33
                  IF(V1.GE.R2) GO TO 33
                  GO TO 34
```

```
33                    NV2FLG=1
                      RETURN
34                    IF(V2.LE.R2) GO TO 36
                      NV2FLG=2
                      RETURN
C          C. ### ROUTINE
35              T=(VM/B)*ALOG((B*V2+A-F)/(B*V1+A-F))
                X=-(A-F)*T/B+(VM/(B*B))*(B*V1+A-F)*(EXP(T*B/VM)-1.)
                RETURN
C          D. LOG ROUTINE - POSITIVE DISCRIMINANT
36              D=SQRT(DSQ)
                V1BAR=(2.*C*V1+B-D)/(2.*C*V1+B+D)
                V2BAR=(2.*C*V2+B-D)/(2.*C*V2+B+D)
                T=(VM/D)*ALOG(V2BAR/V1BAR)
                X=(.5*(D-B)*T-VM*ALOG((1.-V1BAR*EXP(T*D/VM))/
     + (1.-V1BAR)))/C
                RETURN
C          E. RECIPROCAL ROUTINE - ZERO DISCRIMINANT
37              T=2.*VM*(1./(2.*C*V1+B)-1./(2.*C*V2+B))
                X=(VM/C)*ALOG(2.*VM/(2.*VM-T*(2.*C*V1+B)))-.5*B*T/C
                RETURN
C          F. NEGATIVE ROUTINE - MAKE TWO GEARS FITTED BYW STRAIGHT
C    + LINES OUT OF ONE FITTED BY A QUADRATIC.
38              SH=(STRACT(NG,3,K)-STRACT(NG,2,K))/(VG(NG,3,K)-VG(NG,2,K))
                ZH=(STRACT(NG,2,K)*VG(NG,3,K)-STRACT(NG,3,K)*VG(NG,2,K))/
     + (VG(NG,3,K)-VG(NG,2,K))
                SL=(STRACT(NG,2,K)-STRACT(NG,1,K))/(VG(NG,2,K)-VG(NG,1,K))
                ZL=(STRACT(NG,1,K)*VG(NG,2,K)-STRACT(NG,2,K)*VG(NG,1,K))/
     + (VG(NG,2,K)-VG(NG,1,K))
                IF(V2.GE.VG(NG,2,K)) GO TO 39
                   S=SL
                   Z=ZL
                   GO TO 42
39              IF(V1.LE.VG(NG,2,K)) GO TO 40
                   S=SH
                   Z=ZH
                   GO TO 42
40              IF(V2.LE.AMAX1(-(ZH-F)/SH,-(ZL-F)/SL)) GO TO 41
                   NV2FLG=2
                   RETURN
41              TL=(VM/SL)*ALOG((SL*VG(NG,2,K)+ZL-F)/(SL*V1+ZL-F))
                TH=(VM/SH)*ALOG((SH*V2+ZH-F)/(SH*VG(NG,2,K)+ZH-F))
                X=(SL*V1+ZL-F)*VM*(EXP(SL*TL/VM)-1.)/(SL*SL)
     + +(SH*VG(NG,2,K)+ZH-F)*VM*(EXP(SH*TH/VM)-1.)/(SH*SH)
     + -(ZL-F)*TL/SL-(ZH-F)*TH/SH
                T=TL+TH
                RETURN
42              VZ=-(Z-F)/S
                IF(V2.LT.VZ) GO TO 43
                   NV2FLG=0
                   RETURN
43              T=(VM/S)*ALOG((S*V2+Z-F)/(S*V1+Z-F))
```

```
            X=((S*V1+Z-F)*VM*(EXP(S*T/VM)-1.)-(Z-F)*S*T)/(S*S)
            RETURN
C      3. TERMINUS
          END
       SUBROUTINE VELFOR
      + (F        ,FA       ,FB       ,FC       ,FORMX
      + ,K        ,NGR      ,VEL      ,VFMAX    ,VG )
C      ---------------------------------------------------
C      MAXIMUM VELOCITY OVERCOMING A GIVEN RESISTANCE
C      ---------------------------------------------------
C      1. VARIABLE DECLARATION
          REAL       FA       (20,3)
          REAL       FB       (20,3)
          REAL       FC       (20,3)
          REAL       FORMX    (3)
          REAL       VFMAX    (3)
          REAL       VG       (20,3,3)
C      2. ALGORITHM
          DO 10 INDEX=1,NGR
          NG=NGR+1-INDEX
          DSQ=FB(NG,K)**2-4.*FC(NG,K)*(FA(NG,K)-F)
          IF(DSQ) 1,2,4
C         DISCRIMINANT, NEGATIVE
1         IF(FC(NG,K)) 10,10,20
C         DISCRIMINANT, ZERO
2         IF(FC(NG,K)) 10,3,20
C         DISCRIMINANT, POSITIVE
3         IF(FA(NG,K)-F) 10,20,20
4         IF(FC(NG,K)) 11,5,11
5         R=-(   ( FA(NG,K)-F )/FB(NG,K)   )
          IF( VG(NG,1,K)-R) 7,6,6
6         IF(FB(NG,K)) 10,20,20
7         IF(VG(NG,3,K)-R) 9,9,8
8         IF(FB(NG,K)) 30,10,10
9         IF( FB(NG,K)) 20,10,10
11        IF(FB(NG,K)) 12,12,13
12        R2=(-FB(NG,K)+SQRT(DSQ))/(2.*FC(NG,K))
          R1=(FA(NG,K)-F)/(R2*FC(NG,K))
          GO TO 14
13        R1=(-FB(NG,K)-SQRT(DSQ))/(2.*FC(NG,K))
          R2=(FA(NG,K)-F)/(R1*FC(NG,K))
14        RL=AMIN1(R1,R2)
          RH=AMAX1(R1,R2)
          IF(FC(NG,K)) 18,18,15
15        IF(VG(NG,1,K)-RL) 16,16,17
16        IF(VG(NG,3,K)-RL) 20,20,40
17        IF( VG(NG,1,K).GT.RL.AND.VG(NG,1,K).LT.RH) GO TO 10
          GO TO 20
18        IF( VG(NG,1,K)-RL) 10,10,19
19        IF(VG(NG,1,K)-RH) 21,21,10
21        IF(VG(NG,3,K)-RH) 20,20,50
10        CONTINUE
```

```
      IF( FORMX(K)-F) 23,22,22
22    VEL=VFMAX(K)
      GO TO 60
23    VEL=0.
      GO TO 60
20    VEL=VG(NG,3,K)
      GO TO 60
30    VEL=R
      GO TO 60
40    VEL=RL
      GO TO 60
50    VEL=RH
      GO TO 60
C   3. TERMINUS
60    CONTINUE
      END
    REAL FUNCTION MPH (ARG)
C   ---------------
C   UNITS CONVERSION
C   ---------------
C   1. VARIABLE DECLARATION
      REAL ARG
C   2. ALGORITHM
C     A. IN/SEC TO MPH
        MPH = ARG / 17.6
        RETURN
C   3. TERMINUS
      END
C
C
    SUBROUTINE PLTSET(
   +  NPTS,   VGV,   NGR,   ATF,   BTF,   CTF,   IPOWER,
   +  POWER,  TOPSPD,LUN1)
C
C   --------------------------------------
C   PRINTER PLOT OF POWER TRAIN DATA AND
C   QUADRATIC CURVE FITTED TO THE DATA
C   --------------------------------------
C
    REAL POWER  (2,201)
    REAL ATF    (20)
    REAL BTF    (20)
    REAL CTF    (20)
    REAL VGV    (20,5)
    REAL D1     (2,400)
    REAL D2     (2,400)
C
C   NPTS     = NUMBER OF POINTS TO PLOT
C   VGV      = GEAR MIN AND MAX SPEEDS
C   NGR      = MAX NUMBER OF GEARS IN VEHICLE
C   ATF      = A COEF. OF QUADRATIC EQUATION
C   BTF      = B COEF. OF QUADRATIC EQUATION
```

```
C      CTF      = C COEF. OF QUADRATIC EQUATION
C      IPOWER   = NUMBER OF TRACTIVE FORCE POINTS
C      POWER    = TRACTIVE FORCE POINTS FOR MPH
C      TOPSPD   = TOP SPEED OF VEHICLE IN .25 MPH INCREMENTS
C      LUN1     = LOGICAL UNIT 1
C
C    ROUND OFF POWER ARRAY BY NEAREST .25 MPH
C
       CALL RESCAL(IPOWER,POWER)
       CALL FIXER(NPTS,IPOWER,POWER,D1)
C
C    RECONSTRUCT QUADRATIC CURVE DATA USING COEF.
C
       CALL CURPLT(NPTS,VGV,NGR,ATF,BTF,CTF,D2)
C
C    PLOT THE DATA ON PRINTER
C
       CALL PNTPLT(D1,D2,NPTS,TOPSPD,LUN1)
       RETURN
       END
C
C
       SUBROUTINE PNTPLT(
     +    DATA1,    DATA2    ,NPTS,    TOPSPD,    LUN1)
C
C      --------------------------
C      ROUTINE TO DO A PRINT PLOT
C      --------------------------
C
       DIMENSION DATA1(2,400),DATA2(2,400),IPOINT(120),AXH(13)
       DIMENSION LABLY(400),LINE(12)
C
C      DATA1   = FIRST CURCE DATA POINTS
C      DATA2   = SECOND CURVE DATA POINTS
C      TOPSPD    = MAXIMUM MILES PER HOUR IN PLOT
C      LUN1    = LOGICAL UNIT 1
C
       DATA ICASH/10H----------/
       DATA IBLANK/10H          /
       DATA LABLY(52),LABLY(53),LABLY(54),LABLY(55),LABLY(56),
     +      LABLY(57),LABLY(58),LABLY(59),LABLY(60),LABLY(61),
     +      LABLY(62),LABLY(63),LABLY(64),LABLY(65),LABLY(66),
     +      LABLY(67),LABLY(68)/
     +      1HS,1H ,1HP,1H ,1HE,1H ,1HE,1H ,1HD,1H ,1H ,1H ,
     +      1HM,1H ,1HP,1H ,1HH/
C
C    INITIALIZE PRINT LINE TC ALL SPACES
C
       DO 5 K=1,120
       IPOINT(K)=IBLANK
5      CONTINUE
       DO 7 N=1,51
```

```
7       LABLY(N)=IBLANK
        DO 8 N=69,NPTS
8       LABLY(N)=IBLANK
C
C    SET UP Y AXIS LABEL
C
        DO 55 N=1,12
55      LINE(N)=IDASH
C
C  RESCALE POINTS FOR .25 MPH INCREMENTS
C
        CALL RESCAL(NPTS,DATA1)
        CALL RESCAL(NPTS,DATA2)
C
C    COMPUTE MIN, MAX AND SCALE FACTORS FOR X AXIS
C
        CALL LIMITS(DATA1,DATA2,NPTS,BMIN,BMAX,SCALE)
C
C    PRINT X AXIS LABEL AND LABELED TIC MARKS
C
888     FORMAT(1H1,44X,20HTRACTIVE FORCE - LBS,/)
        WRITE(LUN1,888)
        STRT=BMIN
        DO 33 N=1,13
        AXH(N)=STRT
        STRT=STRT+((BMAX-BMIN)/12.)
33      CONTINUE
        WRITE(LUN1,777) (AXH(N),N=1,13)
777     FORMAT(1X,13(F8.0,2X))
        WRITE(LUN1,50) (LINE(N),N=1,12)
50      FORMAT(6X,12A10,1H+)
C
C    INITIALIZE SPEED, FLAGS AND INDEX VARIABLES
C
        VSPEED=0.
        IFLG=100
        INDEX1=1
        INDEX2=1
C
C    NOW BEGIN PLOTTING LOOP
C
   10 CONTINUE
C
C    BLANK PLOT STRING FOR NEW LINE
C
        DO 20 N=1,120
   20 IPOINT(N)=IBLANK
C
C    GET ALL DATA1 POINTS AT THIS SPEED INTO PRINT STRING
C
   30 IF (DATA1(1,INDEX1).NE.VSPEED) GOTO 40
      CALL SCAL(BMIN,SCALE,DATA1(2,INDEX1),1,IPOINT)
```

```
      INDEX1=INDEX1+1
      GOTO 30
C
C   NOW PUT DATA2 POINT FOR THIS SPEED INTO STRING
C
   40 CONTINUE
      CALL SCAL( BMIN,SCALE,DATA2(2,INDEX2),2,IPOINT)
      INDEX2=INDEX2+1
C
C   PRINT STRING, CHECK IFLG TO SEE IF NUMBER IS
C   NEEDED ON AXIS
C
      IF (IFLG.LT.20) GOTO 60
      WRITE( LUN1,1000) LABLY(INDEX2-1),VSPEED,(IPOINT(J),J=1,120)
 1000 FORMAT(1X,A1,F3.0,2H +,120A1)
      IFLG=1
      GOTO 70
   60 CONTINUE
      WRITE( LUN1,1010) LABLY(INDEX2-1),(IPOINT(J),J=1,120)
 1010 FORMAT(1X,A1,3X,2H I,120A1)
      IFLG=IFLG+1
C
C   BUMP VSPEED UP BY 0.25 MPH
C   IF NOT FINISHED, REPEAT FROM 10 WITH NEW VSPEED
C
   70 CONTINUE
      VSPEED=VSPEED+0.25
      IF (INDEX2.LE.NPTS) GOTO 10
      WRITE( LUN1,65) TOPSPD,(LINE(N),N=1,12)
65    FORMAT (1X,F4.1,1X,12A 10,1H+)
      WRITE( LUN1,444)
444   FORMAT (/,21X,6HLEGEND,/,21X,6H------,//,21X,17HX -    DATA POINTS
     +,//,21X,25H* -    FITTED CURVE POINTS,
     +//,21X,48HO -    COINCIDENT VALUES OF CURVE AND DATA POINTS)
      RETURN
      END
C
C
      SUBROUTINE SCAL(
     +   BMIN,    SCALE,    DATA,    IFLG,    IPOINT)
C
C     ----------------------------------
C     HORIZONTAL SCALE SUBROUTINE
C     ----------------------------------
C
      DIMENSION IPOINT(120)
      DATA IEX,ISTAR,IZERO,IBLANK /1HX,1H*,1H0,1H /
C
C     BMIN = MINIMUM VALUE FOR X AXIS
C     SCALE= SCALE FACTOR FROM "LIMITS"
C     DATA = VALUE OF DATA POINT
C     IFLG = PRINT FLAG
```

```
C
C   RETURNED :
C
C     IPOINT = FORMATED LINE OF PLOT CHARS
C
C   COMPUTE SCALE FACTOR
C
      IPT=INT(0.5+((DATA-BMIN)/SCALE))
C
C   DETEMINE PRINT CHARACTER
C
      IF(IPT.LE.0.OR.IPT.GT.120) RETURN
      IF(IPOINT(IPT).EQ.IBLANK) GOTO 10
      IPOINT(IPT)=IZERO
      RETURN
10    IPOINT(IPT)=ISTAR
      IF(IFLG.EQ.1) IPOINT(IPT)=IEX
      RETURN
      END
C
C
      SUBROUTINE RESCAL(
     +   NPTS,    DATA)
C
C     ---------------------------------------------
C     SUBROUTINE TO RESCALE FOR .25 MPH INCR.
C     ---------------------------------------------
C
      DIMENSION DATA(2,400)
C
C   NPTS    = NUMBER OF POINTS IN ARRAY
C   DATA    = ARRAY OF POINTS TO BE CHANGED
C
C   AINCR   = NUMBER OF DIVISIONS PER MILE / HOUR
      AINCR=4.
      DO 10 N=1,NPTS
      ITEMP=INT(.5+AINCR*DATA(1,N))
      DATA(1,N)=FLOAT(ITEMP)/AINCR
10    CONTINUE
      RETURN
      END
C
C
      SUBROUTINE LIMITS(
     +   DATA1,   DATA2,    NPTS,   BMIN,   BMAX,    SCALE)
C
C     ---------------------------------------------
C     LOCATE MINIMUM  MAXIMUM  + SCALE FACTORS
C     ---------------------------------------------
C
      DIMENSION DATA1(2,400),DATA2(2,400)
C
```

```
C       DATA1     = ARRAY OF FIRST DATA
C       DATA2     = ARRAY OF SECOND DATA
C       BMIN      = DATA MINIMUM VALUE
C       BMAX      = DATA MAXIMUM VALUE
C       SCALE     = SCALE FACTOR
C
        BMIN=9.9E99
        BMAX=-9.9E99
        DO 10 N=1,NPTS
        BMIN=AMIN1(BMIN,DATA1(2,N),DATA2(2,N))
        BMAX=AMAX1(BMAX,DATA1(2,N),DATA2(2,N))
10      CONTINUE
        SCALE=ABS(BMAX-BMIN)/120.
        RETURN
        END
C
C
        SUBROUTINE FIXER(
     +    NPTS,    IPOWER,    POWER,   D1)
C
C       ---------------------------------
C       SUBROUTINE TO RESCALE POWER ARRAY
C       ---------------------------------
C
        DIMENSION POWER(2,201),D1(2,400)
C
C       NPTS      = NUMBER OF POINTS IN PLOT
C       IPOWER    = NUMBER OF POINTS IN POWER ARRAY
C       POWER     = POWER DATA MPH AND TRACTIVE FORCE
C       D1        = NEW ARRAY OF POWER POINTS IN .25 MPH INCEMENTS
C
        DO 5 N=1,NPTS
5       D1(1,N)=-1.
        DO 10 N=1,IPOWER
        NN=N
C
C    CONVERT FROM INCH / SEC  TO MPH
C
        D1(1,NN) = POWER(1,N) / 17.6
        D1(2,NN) =POWER(2,N)
10      CONTINUE
        RETURN
        END
C
C
        SUBROUTINE CURPLT(
     +    NPTS,    VGV,    NGR,    ATF,    BTF,    CTF,    D2)
C
C       -------------------------------------
C       GENERATE CURVE FROM QUADRATIC EQUATION
C       -------------------------------------
C
```

```
      REAL ATF     (20)
      REAL BTF     (20)
      REAL CTF     (20)
      REAL VGV     (20,5)
      REAL D2      (2,400)
C
C     NPTS      = NUMBER OF POINT IN CURVE
C     VGV       = GRAR SHIFT SPEEDS
C     NGR       = NUMBER OF GEARS IN VEHICLE
C     ATF       = A COEF. OF QUADRATIC EQUATION
C     BTF       = B COEF. OF QUADRATIC EQUATION
C     CTF       = C COEF. OF QUADRATIC EQUATION
C
C   RETURNED;
C
C     D2      = DATA ARRAY OF CURVE POINTS
C
      VSPEED=-.25
      NG=1
      DO 10 N=1,NPTS
3     CONTINUE
      VSPEED=VSPEED+.25
5     CONTINUE
      IF(NG.GT.NGR) GOTO 10
      Y=ATF(NG) + BTF(NG)*VSPEED*17.6 + CTF(NG)*(VSPEED*17.6)**2
      IF(( VSPEED * 17.6 ) .LT. VGV(NG,5) ) GOTO 20
      NG=NG+1
      GOTO 5
20    CONTINUE
      D2(1,N)=VSPEED
      D2(2,N)=Y
10    CONTINUE
      RETURN
      END
```

APPENDIX B

VEHICLE INPUT FILES FOR PROGRAM NRMM

```
M60, TANK, COMBAT, FULL TRACKED, 105MM GUN
HEAVE NATURAL FREQUENCY 1.36 CY/SEC
$VEHICLE
NAMBLY=1,
NVEH(1)=0,
WGHT(1)=109000.,
IP(1)=1,
IB(1)=1,
REVM(1)=832.0,
TRAKWD(1)=28.,
GROUSH(1)=1.5,
NPAD(1)=1,
ASHOE(1)=194.,
TRAKLN(1)=167.,
NBOGIE(1)=12,
NFL(1)=1,
RW(1)=15.5,
WTE(1)=87.,
WT(1)=115.,
IAPG=1,
ACD=1.2,
CD=1.2,
PFA=92.,
HPNET=643.,
CID=1791.,
IDIESL=1.,
NCYL=12.,
NENG=1.,
QMAX=1682.,
IENGIN=13,
ENGINE(1,1)= 1200., 1610.,
             1300., 1645.,
             1400., 1670.,
             1500., 1682.,
             1600., 1680.,
             1700., 1675.,
             1800., 1655.,
             1900., 1630.,
             2000., 1600.,
             2100., 1560.,
             2200., 1515.,
             2300., 1470.,
             2400., 1420.,
ITCASE=1,
TCASE(1)= .862, .98,
ITRAN=1,
ITVAR=0,
TQIND=900.,
ICONV2=12,
CONV2(1,1)= 3.660, 0.0,
            3.125, 0.1,
            2.650, 0.2,
```

```
               2.228, 0.3,
               1.950, 0.4,
               1.670, 0.5,
               1.420, 0.6,
               1.220, 0.7,
               1.050, 0.8,
               0.980, 0.85,
               0.970, 0.90,
               0.970, 1.00,
 ICONV1=12,
 CONV1(1,1)=   1875., 0.0,
               1850., 0.1,
               1825., 0.2,
               1815., 0.3,
               1830., 0.4,
               1895., 0.5,
               1970., 0.6,
               2030., 0.7,
               2130., 0.8,
               2210., 0.85,
               2500., 0.9,
               2800., 1.0,
 LOCKUP=0,
 NGR=2,
 TRANS(1,1)=   3.497, 0.98,
               1.256, 0.98,
 FD(1)= 5.08, 0.98,
 XBRCOF=0.8,
 TL=167.,
 CL=15.,
 CGH=54.25,
 CGR=119.5,
 CGLAT=0.0,
 PBHT=45.0,
 EYEHGT=55.,
 WDTH=143.,
 WI=87.,
 WC=0.0,
 PBF=218000.,
 MAXL=1,
 MAXIPR=9,
 RMS(1)= 0.25,
         1.0,
         2.0,
         3.0,
         4.0,
         5.0,
         6.0,
         7.0,
         8.0,
 VRIDE(1,1)= 35.25,
             34.00,
```

```
                21.13,
                14.1Ø,
                11.ØØ,
                1Ø.75,
                1Ø.5Ø,
                1Ø.25,
                1Ø.ØØ,
 NHVALS=7,
 VOOB(1)= 1ØØ.,
          1ØØ.,
           12.,
            6.,
            4.,
            4.,
            4.,
 HVALS(1)=  6.Ø,
            9.Ø,
           1Ø.Ø,
           12.Ø,
           15.Ø,
           2Ø.Ø,
           4Ø.Ø,
 NSVALS=9,
 SVALS(1)=   1.,
             5.,
            1Ø.,
            25.,
            5Ø.,
           1ØØ.,
           2ØØ.,
           4ØØ.,
           6ØØ.,
 VOOBS(1)=  .Ø8,
            .39,
            .77,
           1.93,
           3.86,
           7.73,
          15.45,
          3Ø.91,
          46.36,
$END
 NOHGT
     3
  NANG
     8
 NWDTH
     3
```

CLRMIN INCHES	FOOMAX POUNDS	FOO POUNDS	HOVALS INCHES	AVALS RADIANS	WVALS INCHES
37.Ø3	8948.5	372.1	3.15	1.95	5.88
24.42	27Ø76.2	1842.Ø	15.75	1.95	5.88

6.57	89773.8	5211.1	33.46	1.95	5.88
37.03	8948.5	394.3	3.15	2.48	5.88
24.38	24473.2	1604.8	15.75	2.48	5.88
6.72	50134.8	3800.0	33.46	2.48	5.88
37.03	8948.5	399.0	3.15	2.69	5.88
24.56	18969.2	1390.5	15.75	2.69	5.88
11.43	32415.7	3016.3	33.46	2.69	5.88
36.90	8456.0	386.8	3.15	2.86	5.88
24.30	17646.6	1259.3	15.75	2.86	5.88
20.43	30044.5	2787.9	33.46	2.86	5.88
38.22	8281.7	707.0	3.15	3.42	5.88
21.27	18699.8	2246.3	15.75	3.42	5.88
2.87	30044.5	2696.0	33.46	3.42	5.88
39.64	4124.4	224.7	3.15	3.60	5.88
31.01	13744.6	1544.0	15.75	3.60	5.88
-1.30	30816.3	2642.9	33.46	3.60	5.88
40.00	3757.7	174.5	3.15	3.80	5.88
36.03	13166.8	982.9	15.75	3.80	5.88
20.01	31678.1	2626.5	33.46	3.80	5.88
40.00	1612.7	30.6	3.15	4.33	5.88
39.54	4149.3	145.9	15.75	4.33	5.88
37.79	5566.1	-125.5	33.46	4.33	5.88
37.13	9272.2	484.4	3.15	1.95	29.88
24.26	12489.2	-316.4	15.75	1.95	29.88
6.57	79647.8	4974.4	33.46	1.95	29.88
37.13	9272.2	500.0	3.15	2.48	29.88
24.22	20072.6	862.5	15.75	2.48	29.88
6.62	51346.5	4342.5	33.46	2.48	29.88
37.13	9272.2	516.7	3.15	2.69	29.88
24.36	20378.0	1717.0	15.75	2.69	29.88
11.70	34087.7	3769.5	33.46	2.69	29.88
36.99	8456.0	527.7	3.15	2.86	29.88
24.57	15926.4	1465.5	15.75	2.86	29.88
20.55	30044.5	3131.9	33.46	2.86	29.88
37.17	8448.1	629.9	3.15	3.42	29.88
14.79	18895.7	1864.3	15.75	3.42	29.88
2.92	30044.5	3040.6	33.46	3.42	29.88
36.88	7208.2	-219.2	3.15	3.60	29.88
22.08	31861.0	2261.9	15.75	3.60	29.88
-11.56	34784.1	3152.8	33.46	3.60	29.88
36.71	9361.9	1001.2	3.15	3.80	29.88
27.21	20061.7	1637.8	15.75	3.80	29.88
3.49	48386.8	4522.6	33.46	3.80	29.88
38.68	5964.9	196.1	3.15	4.33	29.88
37.04	7279.0	-102.6	15.75	4.33	29.88
35.01	12253.2	759.8	33.46	4.33	29.88
37.17	9272.2	231.1	3.15	1.95	141.60
24.77	20814.9	1040.4	15.75	1.95	141.60
6.59	79704.9	4401.1	33.46	1.95	141.60
37.17	9272.2	236.3	3.15	2.48	141.60
24.44	35968.2	1861.0	15.75	2.48	141.60
6.62	52815.6	3648.1	33.46	2.48	141.60

37.17	9272.2	241.8	3.15	2.69	141.60
24.40	27603.5	1707.9	15.75	2.69	141.60
11.59	34088.9	3306.2	33.46	2.69	141.60
36.93	8456.0	429.9	3.15	2.86	141.60
24.46	18740.7	1827.2	15.75	2.86	141.60
20.55	30044.5	3062.1	33.46	2.86	141.60
34.03	8295.3	471.2	3.15	3.42	141.60
22.76	19012.2	2295.4	15.75	3.42	141.60
20.46	30044.5	3493.0	33.46	3.42	141.60
34.12	9326.8	741.4	3.15	3.60	141.60
16.75	32341.8	2497.8	15.75	3.60	141.60
9.38	34368.4	4266.5	33.46	3.60	141.60
33.89	9787.3	452.9	3.15	3.80	141.60
12.40	38383.1	2027.9	15.75	3.80	141.60
-1.83	48928.4	3741.5	33.46	3.80	141.60
33.91	8474.2	608.2	3.15	4.33	141.60
10.90	18269.4	955.9	15.75	4.33	141.60
-23.03	79892.1	5167.6	33.46	4.33	141.60

```
M151 JEEP
NATO MOBILITY MODEL
 TEST VEHICLE
 $VEHICLE
  WI=45.8,
  LOCDIF=0,
  NAMBLY=2,
  NVEH(1)=1,
  NVEH(2)=1,
  WGHT(1)=1740.0,
  WGHT(2)=1460.0,
  IP(1)=1,
  IP(2)=1,
  IB(1)=1,
  IB(2)=1,
  RDIAM(1)=16.0,
  RDIAM(2)=16.0,
  RIMW(1)=4.5,
  RIMW(2)=4.5,
  ICONST(1)=1,
  ICONST(2)=1,
  TPLY(1)=6.0,
  TPLY(2)=6.0,
  REVM(1)=720.0,
  REVM(2)=720.0,
  DIAW(1)=30.8,
  DIAW(2)=30.8,
  SECTW(1)=7.15,
  SECTW(2)=7.15,
  SECTH(1)=7.40,
  SECTH(2)=7.40,
  TPSI(1,2)=15.0,
  TPSI(2,2)=15.0,
  TPSI(1,1)=15.0,
  TPSI(2,1)=15.0,
  TPSI(1,3)=25.0,
  TPSI(2,3)=25.0,
  DFLCT(1,1)=1.31,
  DFLCT(1,2)=1.31,
  DFLCT(2,1)=1.14,
  DFLCT(2,2)=1.14,
  DFLCT(1,3)=1.0,
  DFLCT(2,3)=1.0,
  NWHL(1)=2,
  NWHL(2)=2,
  ID(1)=0,
  ID(2)=0,
  CLRMIN(1)=11.4,
  CLRMIN(2)=11.4,
  WTE(1)=45.6,
  WTE(2)=45.6,
  WT(1)=53.0,
```

```
WT(2)=53.0,
NCHAIN(1)=0,
NCHAIN(2)=0,
HPNET=73.92,
ACD=1.2,
CD=1.2,
AXLSP(1)=85.,
AVGC=120.,
CID=141.5,
IDIESL=1.,
NCYL=4.,
NENG=1.,
QMAX=115.,
IT(1)=0,
IT(2)=0,
PFA=22.5,
IAPG=0,
IPOWER=17,
POWER(2,1)=2195.0,
POWER(2,2)=2185.0,
POWER(2,3)=2050.0,
POWER(2,4)=1815.0,
POWER(2,5)=1205.0,
POWER(2,6)=1180.0,
POWER(2,7)=1085.0,
POWER(2,8)=870.0,
POWER(2,9)=660.0,
POWER(2,10)=650.0,
POWER(2,11)=615.0,
POWER(2,12)=560.0,
POWER(2,13)=420.0,
POWER(2,14)=385.0,
POWER(2,15)=355.0,
POWER(2,16)=340.0,
POWER(2,17)=310.0,
POWER(1,1)=0.0,
POWER(1,2)=4.9,
POWER(1,3)=7.5,
POWER(1,4)=10.0,
POWER(1,5)=10.1,
POWER(1,6)=12.0,
POWER(1,7)=15.5,
POWER(1,8)=19.8,
POWER(1,9)=19.9,
POWER(1,10)=25.0,
POWER(1,11)=30.0,
POWER(1,12)=33.0,
POWER(1,13)=33.1,
POWER(1,14)=40.0,
POWER(1,15)=45.0,
POWER(1,16)=50.0,
POWER(1,17)=56.0,
```

```
ITRAN=0,
IENGIN=10,
ENGINE(1,1)=800.0,
ENGINE(1,2)=1200.0,
ENGINE(1,3)=1600.0,
ENGINE(1,4)=2000.0,
ENGINE(1,5)=2400.0,
ENGINE(1,6)=2800.0,
ENGINE(1,7)=3200.0,
ENGINE(1,8)=3600.0,
ENGINE(1,9)=4000.0,
ENGINE(1,10)=4400.0,
ENGINE(2,1)=115.0,
ENGINE(2,2)=115.0,
ENGINE(2,3)=115.0,
ENGINE(2,4)=115.0,
ENGINE(2,5)=112.0,
ENGINE(2,6)=108.0,
ENGINE(2,7)=103.0,
ENGINE(2,8)=96.0,
ENGINE(2,9)=88.0,
ENGINE(2,10)=80.0,
ITCASE=0,
TCASE(1)=1.0,
TCASE(2)=1.0,
NGR=4,
TRANS(1,1)=5.712,
TRANS(1,2)=3.179,
TRANS(1,3)=1.674,
TRANS(1,4)=1.000,
TRANS(2,1)=0.9,
TRANS(2,2)=0.9,
TRANS(2,3)=0.9,
TRANS(2,4)=0.9,
FD(1)=4.86,
FD(2)=0.9,
LOCKUP=0,
XBRCOF=0.7,
CL=9.1,
VAA=66.0,
VDA=37.0,
TL=85.0,
WDTH=64.0,
CGH=10.3,
CGR=42.0,
CGLAT=0.0,
PBHT=20.0,
EYEHGT=52.5,
WC=0.0,
PBF=3200.0,
MAXL=1,
MAXIPR=9,
```

```
RMS(1)=0.25,
RMS(2)=1.0,
RMS(3)=2.0,
RMS(4)=3.0,
RMS(5)=4.0,
RMS(6)=5.0,
RMS(7)=6.0,
RMS(8)=7.0,
RMS(9)=8.0,
VRIDE(1,1)=76.5,
VRIDE(2,1)=21.0,
VRIDE(3,1)=9.5,
VRIDE(4,1)=5.1,
VRIDE(5,1)=2.0,
VRIDE(6,1)=2.0,
VRIDE(7,1)=2.0,
VRIDE(8,1)=2.0,
VRIDE(9,1)=2.0,
NHVALS=7,
VOOB(1)=100.0,
VOOB(2)=20.0,
VOOB(3)=6.0,
VOOB(4)=4.0,
VOOB(5)=2.0,
VOOB(6)=1.0,
VOOB(7)=1.0,
HVALS(1)=0.01,
HVALS(2)=4.0,
HVALS(3)=8.0,
HVALS(4)=12.0,
HVALS(5)=16.0,
HVALS(6)=20.0,
HVALS(7)=40.0,
NSVALS=16,
VOOBS(1)=15.0,
VOOBS(2)=15.0,
VOOBS(3)=15.0,
VOOBS(4)=15.0,
VOOBS(5)=15.0,
VOOBS(6)=15.0,
VOOBS(7)=15.0,
VOOBS(8)=15.0,
VOOBS(9)=15.0,
VOOBS(10)=15.0,
VOOBS(11)=15.0,
VOOBS(12)=15.0,
VOOBS(13)=15.0,
VOOBS(14)=15.0,
VOOBS(15)=15.0,
VOOBS(16)=15.0,
SVALS(1)=1.0,
SVALS(2)=2.0,
```

```
 SVALS(3)=3.0,
 SVALS(4)=4.0,
 SVALS(5)=5.0,
 SVALS(6)=6.0,
 SVALS(7)=7.0,
 SVALS(8)=8.0,
 SVALS(9)=9.0,
 SVALS(10)=10.0,
 SVALS(11)=11.0,
 SVALS(12)=12.0,
 SVALS(13)=13.0,
 SVALS(14)=15.0,
 SVALS(15)=20.0,
 SVALS(16)=40.0,
$
 NOHGT
     3
  NANG
     8
 NWDTH
     3
```

CLRMIN	FOOMAX	FOO	HOVALS	AVALS	WVALS
INCHES	POUNDS	POUNDS	INCHES	RADIANS	INCHES
6.85	941.6	31.2	3.15	1.95	5.88
-3.75	2179.6	127.1	15.75	1.95	5.88
-21.21	2208.5	237.5	33.46	1.95	5.88
6.85	1015.5	35.6	3.15	2.48	5.88
-3.54	1061.2	118.7	15.75	2.48	5.88
-13.36	960.9	160.6	33.46	2.48	5.88
6.85	696.1	25.5	3.15	2.69	5.88
-2.31	696.7	124.9	15.75	2.69	5.88
-3.95	646.3	98.2	33.46	2.69	5.88
7.45	411.2	34.3	3.15	2.86	5.88
2.93	404.0	69.7	15.75	2.86	5.88
2.61	799.3	98.3	33.46	2.86	5.88
7.19	417.7	40.9	3.15	3.42	5.88
5.50	444.5	88.7	15.75	3.42	5.88
3.10	799.3	103.9	33.46	3.42	5.88
7.42	704.7	35.5	3.15	3.60	5.88
1.20	757.6	135.1	15.75	3.60	5.88
-4.83	839.1	135.3	33.46	3.60	5.88
8.20	662.5	16.3	3.15	3.80	5.88
.08	1170.4	180.3	15.75	3.80	5.88
-9.54	1301.5	240.0	33.46	3.80	5.88
9.65	344.3	4.8	3.15	4.33	5.88
5.79	1150.8	43.5	15.75	4.33	5.88
-.23	2378.0	146.0	33.46	4.33	5.88
6.85	592.1	-2.8	3.15	1.95	29.88
-3.75	2163.4	99.1	15.75	1.95	29.88
-21.46	2029.6	150.9	33.46	1.95	29.88
6.85	1015.5	29.3	3.15	2.48	29.88
-3.75	1052.4	98.4	15.75	2.48	29.88

-4.92	1110.3	109.8	33.46	2.48	29.88
6.85	698.1	24.7	3.15	2.69	29.88
.59	658.0	69.2	15.75	2.69	29.88
.52	837.9	116.9	33.46	2.69	29.88
7.45	411.2	28.8	3.15	2.86	29.88
4.86	443.4	50.1	15.75	2.86	29.88
4.75	799.3	105.0	33.46	2.86	29.88
7.29	417.6	31.1	3.15	3.42	29.88
5.40	444.5	57.0	15.75	3.42	29.88
4.92	799.3	108.6	33.46	3.42	29.88
6.83	708.6	39.9	3.15	3.60	29.88
.78	761.3	119.2	15.75	3.60	29.88
-2.82	842.2	137.0	33.46	3.60	29.88
6.70	991.4	34.9	3.15	3.80	29.88
-2.46	1178.4	145.1	15.75	3.80	29.88
-10.26	1318.0	195.9	33.46	3.80	29.88
6.68	575.1	4.9	3.15	4.33	29.88
-3.01	2401.8	157.0	15.75	4.33	29.88
-23.83	2551.4	228.7	33.46	4.33	29.88
6.85	541.3	-6.0	3.15	1.95	141.60
-.50	2428.4	87.4	15.75	1.95	141.60
-11.40	2556.1	128.8	33.46	1.95	141.00
6.85	1093.9	18.1	3.15	2.48	141.60
2.04	1170.6	68.6	15.75	2.48	141.60
-.73	1304.9	145.9	33.46	2.48	141.60
6.85	707.5	16.9	3.15	2.69	141.60
4.40	758.7	75.1	15.75	2.69	141.60
3.83	837.9	132.5	33.46	2.69	141.60
7.45	416.8	17.0	3.15	2.86	141.60
6.75	443.4	65.4	15.75	2.86	141.60
6.88	799.3	103.0	33.46	2.86	141.60
7.67	417.2	19.1	3.15	3.42	141.60
7.28	388.0	65.9	15.75	3.42	141.60
6.85	799.3	106.8	33.46	3.42	141.60
6.84	707.1	20.1	3.15	3.60	141.60
4.25	760.1	78.2	15.75	3.60	141.60
3.88	839.7	135.9	33.46	3.60	141.60
7.08	1094.0	18.6	3.15	3.80	141.60
2.04	1168.7	83.3	15.75	3.80	141.60
-.60	1312.2	164.2	33.46	3.80	141.60
6.80	1131.4	30.3	3.15	4.33	141.60
-.03	2397.2	80.3	15.75	4.33	141.60
-15.46	2549.8	147.3	33.46	4.33	141.60

APPENDIX C

TERRAIN INPUT FILES FOR PROGRAM NRMM

```
 1 1 1 1 11 1111111111111111
 2 1 5 5 51 1111111111111111
 3 1 6 6 61 1111111111111111
 4 1 8 8 81 1111111111111111
 5 11010101 1111111111111111
 6 1 1 1 12 1111111111111111
 7 1 1 1 13 1111111111111111
 8 1 1 1 14 1111111111111111
 9 1 1 1 16 1111111111111111
10 1 1 1 17 1111111111111111
11 2 1 1 11 1111111111111111
12 2 5 5 51 1111111111111111
13 2 6 6 61 1111111111111111
14 2 8 8 81 1111111111111111
15 21010101 1111111111111111
16 2 1 1 12 1111111111111111
17 2 1 1 13 1111111111111111
18 2 1 1 14 1111111111111111
19 2 1 1 16 1111111111111111
20 2 1 1 17 1111111111111111
21 1 1 1 11 1111111111111111
22 1 1 1 11 1111113111111111
23 1 1 1 11 1111115111111111
24 1 1 1 11 1111117111111111
25 1 1 1 11 1111119111111111
26 1 1 1 11 1111111666665551
27 1 1 1 11 1111111887655561
28 1 1 1 11 1111111666666661
29 1 1 1 11 1111111888876551
30 1 1 1 11 1111111888887651
31 1 1 1 1111433311666665551
32 1 5 5 5111433311887655551
33 1 5 5 5111433311666666661
34 1 1 1 1111433211887655551
35 1 5 5 5111433211666665551
36 1 1 1 1111135211111111111
37 1 1 1 1111135221111111111
38 1 1 1 1111335221111111111
39 1 1 1 1111535221111111111
40 1 1 1 1111735221111111111
41 1 1 1 1113437221111111111
42 1 1 1 1113437321111111111
43 1 1 1 1113437821111111111
44 1 1 1 11 1111111111111112
45 1 1 1 11 1111111111111113
46 1 1 1 11 1111111111111116
47 1 5 5 51 1111111111111112
48 1 5 5 51 1111111111111113
49 1 5 5 51 1111111111111116
50 1 7 7 71 1111111111111118
```

1	1	1	4	300	300	300	300	3	1000	1000	1000	1000	1	70	0.1000
2	2	1	4	300	300	300	300	7	500	500	500	500	2	40	0.1100
3	2	1	4	300	300	300	300	7	500	500	500	500	2	55	0.1200
4	2	1	1	300	300	300	300	3	500	500	500	500	2	25	0.1300
5	3	1	4	300	300	300	300	3	250	250	250	250	8	55	0.1400
6	3	1	4	300	300	300	300	15	500	500	500	500	8	55	0.1500
7	3	1	4	300	300	300	300	3	500	500	500	500	25	40	0.1600
8	3	1	4	250	190	80	80	3	250	250	250	250	8	40	0.1700
9	4	1	4	250	190	36	36	3	250	250	250	250	8	55	0.1800
10	4	1	4	200	200	200	200	15	250	250	250	250	8	55	0.1900
11	4	1	4	200	200	80	80	15	250	250	250	250	8	55	0.2000
12	4	1	4	250	200	200	200	3	250	250	250	250	25	40	0.2100
13	4	1	4	250	200	200	200	3	79	79	79	79	10	30	0.2200
14	4	1	4	250	200	200	200	3	250	250	250	250	10	9	0.2300

APPENDIX D

SAMPLE OUTPUT OF PROGRAM NRMM

This Appendix contains output generated by executing NRMM using the vehicle and terrain input files of Appendices B and C. In all cases, the control variable DETAIL was set to 2. This results in output of the control, scenario and vehicle data through NAMELIST directed WRITE's. This is followed by the speed-made-good as described in Section II.E. The control variable SEARCH was set to 0.

For the Areal terrain, the scenario variables MAP and ISEASN were set to 71 and 3 respectively. For the Road terrain, the scenario inputs were MAP = 11, ISEASN = 3 and MONTH = 1. The default values were used for all other control and scenario variables.

These outputs are presented as examples. The terrain files are artificial. They were made up to systematically exercise portions of the Model.

SAMPLE OUTPUT OF PROGRAM NRMM - VEHICLE: M60A1, TERRAIN: CKK

```
$CONTRL
DETAIL   = 2,
KSCEN    = 1,
KVEH     = 1,
KII1     = 0,
KII2     = 0,
KII3     = 0,
KII4     = 0,
KII5     = 0,
KII6     = 0,
KII7     = 0,
KII8     = 0,
KII9     = 0,
KII10    = 0,
KII11    = 0,
KII12    = 0,
KII13    = 0,
KII14    = 0,
KII15    = 0,
KII16    = 0,
KII17    = 0,
KMAP     = 0,
KTPP     = 0,
KIV1     = 0,
KIV2     = 0,
KIV3     = 0,
KIV4     = 0,
KIV5     = 0,
KIV6     = 0,
KIV7     = 0,
KIV8     = 0,
KIV9     = 0,
KIV10    = 0,
KIV11    = 0,
KIV12    = 0,
KIV13    = 0,
KIV14    = 0,
KIV15    = 0,
KIV16    = 0,
KIV17    = 0,
KIV18    = 0,
KIV19    = 0,
KIV20    = 0,
KIV21    = 0,
NTUX     = 1,
SEARCH   = 0,
$END
```

```
$SCENAR
COHES    = .5E-01,
DCLMAX   = .5E+00,
GAMMA    = .2E+00,
IOVER    = 9,
ISEASN   = 3,
ISURF    = 1,
ISNOW    = 0,
LAC      = 1,
MAP      = 71,
MAPG     = 1,
MONTH    = 1,
NOPP     = 0,
NSLIP    = 0,
NTRAV    = 3,
NTUX     = 1,
PHI      = .21E+02,
REACT    = .5E+00,
RDFOG    = .1E+04,
SFTYPC   = .9E+02,
VBRAKE   = .5E+01,
VISMNV   = .2E+01,
VLIM     = .55E+02,
ZSNOW    = .3E+01,
$END
```

```
$VEHICLE
ACD      = .12E+01,
ASHOE    = .194E+03, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
AVGC     = 0.0,
AXLSP    = 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
CD       = .12E+01,
CGH      = .5425E+02,
CGLAT    = 0.0,
CGR      = .1195E+03,
CID      = .1791E+04,
CL       = .15E+02,
CLRMIN   = 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
CONV1    = .1875E+04, 0.0, .185E+04, .1E+00, .1825E+04, .2E+00,
.1815E+04,      .3E+00, .183E+04, .4E+00, .1895E+04,
         .5E+00, .197E+04, .6E+00, .203E+04, .7E+00, .213E+04, .8E+00,
     .221E+04, .85E+00, .25E+04, .9E+00,
         .28E+04, .1E+01, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0,     0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
         0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
CONV2    = .366E+01, 0.0, .3125E+01, .1E+00, .265E+01, .2E+00, .2228E+01,
     .3E+00, .195E+01, .4E+00, .167E+01,
         .5E+00, .142E+01, .6E+00, .122E+01, .7E+00, .105E+01, .8E+00,
     .98E+00, .85E+00, .97E+00, .9E+00,
         .97E+00, .1E+01, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0,     0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
         0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
DFLCT    = 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
         0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
         0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
DIAW     = 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
DRAFT    = 0.0,
ENGINE   = .12E+04, .161E+04, .13E+04, .1645E+04, .14E+04, .167E+04,
.15E+04,     .1682E+04, .16E+04, .168E+04, .17E+04,
         .1675E+04, .18E+04, .1655E+04, .19E+04, .163E+04, .2E+04,
.16E+04,     .21E+04, .156E+04, .22E+04, .1515E+04,
         .23E+04, .147E+04, .24E+04, .142E+04, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0,     0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
         0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
         0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
         0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
EYEHGT   = .55E+02,
FD       = .508E+01, .98E+00,
```

```
FORDD    = Ø.Ø,
GROUSH   = .15E+Ø1, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø, Ø.Ø,  Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
HPNET    = .643E+Ø3,
HVALS    = Ø.Ø, .9E+Ø1, .1E+Ø2, .12E+Ø2, .15E+Ø2, .2E+Ø2, .4E+Ø2, Ø.Ø,
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
           Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
IAPG     = 1,
IB       = 1, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø,
ID       = Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø,
IDIESL   = .1E+Ø1 ,
IENGIN   = 13,
ICONST   = Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø,
ICONV1   = 12,
ICONV2   = 12,
IP       = 1, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø,
IPOWER   = Ø,
IT       = Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø,
ITCASE   = 1,
ITRAN    = 1,
ITVAR    = Ø,
LOCDIF   = Ø,
LOCKUP   = Ø,
MAXIPR   = 9,
MAXL     = 1,
NAMBLY   = 1,
NBOGIE   = 12, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø,
NCHAIN   = Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø,
NCYL     = .12E+Ø2,
NENG     = .1E+Ø1 ,
NFL      = 1, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø,
NGR      = 2,
NHVALS   = 7,
NPAD     = 1, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø,
NSVALS   = 9,
NVEH     = Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø,
NWHL     = Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø,
NWR      = Ø,
PBF      = .218E+Ø6,
PBHT     = .45E+Ø2,
PFA      = .92E+Ø2,
POWER    = Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
           Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
           Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
           Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
           Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
           Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
```

```
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
          Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
          Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
          Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
          Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
          Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
          Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
          Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
          Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
          Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
          Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
          Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
          Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
          Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
          Ø.Ø, Ø.Ø, Ø.Ø,
QMAX     = .1682E+Ø4,
RDIAM    = Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
REVM     = .832E+Ø3, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
RTMW     = Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
RMS      = .25E+ØØ, .1E+Ø1, .2E+Ø1, .3E+Ø1, .4E+Ø1, .5E+Ø1, .6E+Ø1,
.7E+Ø1,        .8E+Ø1, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
          Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
RW       = .155E+Ø2, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
SAE      = Ø.Ø,
SAI      = Ø.Ø,
SECTH    = Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
SECTW    = Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
SVALS    = .1E+Ø1, .5E+Ø1, .1E+Ø2, .25E+Ø2, .5E+Ø2, .1E+Ø3, .2E+Ø3,
.4E+Ø3,        .6E+Ø3, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
          Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
TCASE    = .862E+ØØ, .98E+ØØ,
TL       = .167E+Ø3,
TPLY     = Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
```

```
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
TPSI     = 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
           0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
           0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
TOIND    = .9E+03,
TRAKLN   = .167E+03, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
TRAKWD   = .28E+02, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
TRANS    = .3497E+01, .98E+00, .1256E+01, .98E+00, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
           0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
           0.0, 0.0,
VAA      = 0.0,
VDA      = 0.0,
VFS      = 0.0,
VOOB     = .1E+03, .1E+03, .12E+02, .6E+01, .4E+01, .4E+01, .4E+01, 0.0,
0.0,     0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
           0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
VOOBS    = .8E-01, .39E+00, .77E+00, .193E+01, .386E+01, .773E+01,
.1545E+02,    .3091E+02, .4636E+02, 0.0, 0.0, 0.0,
           0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0,
VRIDE    = .3525E+02, .34E+02, .2113E+02, .141E+02, .11E+02, .1075E+02,
      .105E+02, .1025E+02, .1E+02, 0.0, 0.0,
           0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
           0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
           0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
VSS      = 0.0,
VSSAXP   = 0.0,
WC       = 0.0,
WDAXP    = 0.0,
WDPTH    = 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
WDTH     = .143E+03,
WGHT     = .109E+06, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
WI       = .87E+02,
WRAT     = 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
WRFORD   = 0.0,
WT       = .115E+03, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
WTE      = .87E+02, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
WWAXP    = 0.0,
```

```
XBRCOF   = .8 E+00,
$END
```

SAMPLE OUTPUT OF PROGRAM NRMM - VEHICLE: M60A1, TERRAIN: CKK

NTU	ITUT	VMAX	UP	LEVEL	DOWN	VSEL	UP	LEVEL	DOWN	GRADE	AREA
1	1	27.07	27.07	27.07	27.07	27.07	27.07	27.07	27.07	1.00	0.0000
2	1	21.20	18.81	21.15	24.35	21.20	18.81	21.15	24.35	1.00	0.0000
3	1	15.11	13.63	15.16	16.86	15.11	13.63	15.16	16.86	1.00	0.0000
4	1	8.09	7.68	8.08	8.55	8.09	7.68	8.08	8.55	1.00	0.0000
5	1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.0000
6	1	23.86	19.29	27.07	27.07	23.86	19.29	27.07	27.07	3.50	0.0000
7	1	19.56	12.58	27.07	27.07	19.56	12.58	27.07	27.07	7.50	0.0000
8	1	15.54	8.39	27.07	27.07	15.54	8.39	27.07	27.07	15.00	0.0000
9	1	6.10	2.39	27.07	27.05	6.10	2.39	27.07	27.05	50.00	0.0000
10	1	2.17	.77	27.07	19.17	2.17	.77	27.07	19.17	65.00	0.0000
11	1	16.73	15.10	16.78	18.71	16.73	15.10	16.78	18.71	1.00	0.0000
12	1	9.20	8.63	9.22	9.82	9.20	8.63	9.22	9.82	1.00	0.0000
13	1	7.01	6.73	7.01	7.33	7.01	6.73	7.01	7.33	1.00	0.0000
14	1	5.48	5.28	5.49	5.71	5.48	5.28	5.49	5.71	1.00	0.0000
15	1	3.92	3.74	3.92	4.11	3.92	3.74	3.92	4.11	1.00	0.0000
16	1	16.31	11.53	16.78	26.54	16.31	11.53	16.78	26.54	3.50	0.0000
17	1	15.14	9.83	16.78	27.12	15.14	9.83	16.78	27.12	7.50	0.0000
18	1	12.27	6.76	16.78	27.12	12.27	6.76	16.78	27.12	15.00	0.0000
19	1	4.66	1.82	16.78	27.11	4.66	1.82	16.78	27.11	50.00	0.0000
20	1	2.47	.91	16.78	17.26	2.47	.91	16.78	17.26	65.00	0.0000
21	1	27.07	27.07	27.07	27.07	27.07	27.07	27.07	27.07	1.00	0.0000
22	1	21.13	21.13	21.13	21.13	21.13	21.13	21.13	21.13	1.00	0.0000
23	1	11.00	11.00	11.00	11.00	11.00	11.00	11.00	11.00	1.00	0.0000
24	1	10.50	10.50	10.50	10.50	10.50	10.50	10.50	10.50	1.00	0.0000
25	1	10.00	10.00	10.00	10.00	10.00	10.00	10.00	10.00	1.00	0.0000
26	1	14.54	13.12	14.60	16.23	9.26	9.26	9.26	9.26	1.00	0.0000
27	1	14.30	12.90	14.36	15.97	9.26	9.26	9.26	9.26	1.00	0.0000
28	1	9.98	9.84	9.97	10.14	0.00	0.00	0.00	0.00	1.00	0.0000
29	1	9.26	9.26	9.26	9.26	9.26	9.26	9.26	9.26	1.00	0.0000
30	1	5.37	5.17	5.37	5.59	5.37	5.17	5.37	5.59	1.00	0.0000
31	1	14.54	13.12	14.60	16.23	8.58	8.58	8.58	8.58	1.00	0.0000
32	1	10.74	10.11	10.36	11.92	8.58	8.58	8.58	8.58	1.00	0.0000
33	1	9.08	8.50	9.06	9.76	0.00	0.00	0.00	0.00	1.00	0.0000
34	1	14.30	12.90	14.36	15.97	9.24	9.24	9.24	9.24	1.00	0.0000
35	1	10.98	10.14	10.84	12.13	9.24	9.24	9.24	9.24	1.00	0.0000
36	1	27.07	27.07	27.07	27.07	27.07	27.07	27.07	27.07	1.00	0.0000
37	1	27.07	27.07	27.07	27.07	27.07	27.07	27.07	27.07	1.00	0.0000
38	1	10.45	10.28	10.44	10.63	10.45	10.28	10.44	10.63	1.00	0.0000
39	1	8.34	8.23	8.34	8.46	8.34	8.23	8.34	8.46	1.00	0.0000
40	1	8.09	7.99	8.09	8.20	8.09	7.99	8.09	8.20	1.00	0.0000
41	1	8.54	8.42	8.54	8.66	8.54	8.42	8.54	8.66	1.00	0.0000
42	1	5.76	5.72	5.75	5.81	5.76	5.72	5.75	5.81	1.00	0.0000
43	1	3.61	3.61	3.61	3.61	3.61	3.61	3.61	3.61	1.00	0.0000
44	1	27.07	27.07	27.07	27.07	27.07	27.07	27.07	27.07	1.00	0.0000
45	1	23.48	23.48	23.48	23.48	23.48	23.48	23.48	23.48	1.00	0.0000
46	1	10.90	10.90	10.90	10.90	10.90	10.90	10.90	10.90	1.00	0.0000
47	1	21.20	18.81	21.15	24.35	21.20	18.81	21.15	24.35	1.00	0.0000
48	1	20.97	18.81	21.15	23.48	20.97	18.81	21.15	23.48	1.00	0.0000
49	1	10.90	10.90	10.90	10.90	10.90	10.90	10.90	10.90	1.00	0.0000
50	1	6.04	6.04	6.04	6.04	6.04	6.04	6.04	6.04	1.00	0.0000

```
$CONTRL
DETAIL  = 2,
KSCEN   = 1,
KVEH    = 1,
KII1    = 0,
KII2    = 0,
KII3    = 0,
KII4    = 0,
KII5    = 0,
KII6    = 0,
KII7    = 0,
KII8    = 0,
KII9    = 0,
KII10   = 0,
KII11   = 0,
KII12   = 0,
KII13   = 0,
KII14   = 0,
KII15   = 0,
KII16   = 0,
KII17   = 0,
KMAP    = 0,
KTPP    = 0,
KIV1    = 0,
KIV2    = 0,
KIV3    = 0,
KIV4    = 0,
KIV5    = 0,
KIV6    = 0,
KIV7    = 0,
KIV8    = 0,
KIV9    = 0,
KIV10   = 0,
KIV11   = 0,
KIV12   = 0,
KIV13   = 0,
KIV14   = 0,
KIV15   = 0,
KIV16   = 0,
KIV17   = 0,
KIV18   = 0,
KIV19   = 0,
KIV20   = 0,
KIV21   = 0,
NTUX    = 1,
SEARCH  = 0,
$END
```

```
$SCENAR
COHES    = .5E-01,
DCLMAX   = .5E+00,
GAMMA    = .2E+00,
IOVER    = 9,
ISEASN   = 3,
ISURF    = 1,
ISNOW    = 0,
LAC      = 1,
MAP      = 11,
MAPG     = 1,
MONTH    = 1,
NOPP     = 0,
NSLIP    = 0,
NTRAV    = 3,
NTUX     = 1,
PHI      = .21E+02,
REACT    = .5E+00,
RDFOG    = .1E+04,
SFTYPC   = .9E+02,
VBRAKE   = .5E+01,
VISMNV   = .2E+01,
VLIM     = .55E+02,
ZSNOW    = .3E+01,
$END
```

```
$VEHICLE
ACD      = .12E+01,
ASHOE    = .194E+03, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
AVGC     = 0.0,
AXLSP    = 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
CD       = .12E+01,
CGH      = .5425E+02,
CGLAT    = 0.0,
CGR      = .1195E+03,
CID      = .1791E+04,
CL       = .15E+02,
CLRMIN   = 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
CONV1    = .1875E+04, 0.0, .185E+04, .1E+00, .1825E+04, .2E+00,
.1815E+04,        .3E+00, .183E+04, .4E+00, .1895E+04,
         .5E+00, .197E+04, .6E+00, .203E+04, .7E+00, .213E+04, .8E+00,
      .221E+04, .85E+00, .25E+04, .9E+00,
         .28E+04, .1E+01, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0,     0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
         0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
CONV2    = .366E+01, 0.0, .3125E+01, .1E+00, .265E+01, .2E+00, .2228E+01,
      .3E+00, .195E+01, .4E+00, .167E+01,
         .5E+00, .142E+01, .6E+00, .122E+01, .7E+00, .105E+01, .8E+00,
      .98E+00, .85E+00, .97E+00, .9E+00,
         .97E+00, .1E+01, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0,     0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
         0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
DFLCT    = 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
         0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
         0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
DIAW     = 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
DRAFT    = 0.0,
ENGINE   = .12E+04, .161E+04, .13E+04, .1645E+04, .14E+04, .167E+04,
.15E+04,      .1682E+04, .16E+04, .168E+04, .17E+04,
         .1675E+04, .18E+04, .1655E+04, .19E+04, .163E+04, .2E+04,
.16E+04,      .21E+04, .156E+04, .22E+04, .1515E+04,
         .23E+04, .147E+04, .24E+04, .142E+04, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0,     0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
         0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
         0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
         0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
EYEHGT   = .55E+02,
FD       = .508E+01, .98E+00,
```

```
FORDD    = 0.0,
GROUSH   = .15E+01, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0,  0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
HPNET    = .643E+03,
HVALS    = 0.0, .9E+01, .1E+02, .12E+02, .15E+02, .2E+02, .4E+02, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
           0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
IAPG     = 1,
IB       = 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
ID       = 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
IDIESL   = .1E+01,
IENGIN   = 13,
ICONST   = 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
ICONV1   = 12,
ICONV2   = 12,
IP       = 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
IPOWER   = 0,
IT       = 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
ITCASE   = 1,
ITRAN    = 1,
ITVAR    = 0,
LOCDIF   = 0,
LOCKUP   = 0,
MAXIPK   = 9,
MAXL     = 1,
NAMBLY   = 1,
NBOGIE   = 12, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
NCHAIN   = 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
NCYL     = .12E+02,
NENG     = .1E+01,
NFL      = 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
NGR      = 2,
NHVALS   = 7,
NPAD     = 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
NSVALS   = 9,
NVEH     = 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
NWHL     = 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
NWR      = 0,
PBF      = .218E+06,
PBHT     = .45E+02,
PFA      = .92E+02,
POWER    = 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
           0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
           0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
           0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
           0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
           0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
```

```
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
          0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
          0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
          0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
          0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
          0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
          0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
          0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
          0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
          0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
          0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
          0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
          0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
          0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
          0.0, 0.0, 0.0,
QMAX    = .1682E+04,
RDIAM   = 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
REVM    = .832E+03, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
RTMW    = 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
RMS     = .25E+00, .1E+01, .2E+01, .3E+01, .4E+01, .5E+01, .6E+01,
.7E+01,     .8E+01, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
          0.0, 0.0, 0.0, 0.0, 0.0,
RW      = .155E+02, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
SAE     = 0.0,
SAI     = 0.0,
SECTH   = 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
SECTW   = 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
SVALS   = .1E+01, .5E+01, .1E+02, .25E+02, .5E+02, .1E+03, .2E+03,
.4E+03,     .6E+03, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
          0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
TCASE   = .862E+00, .98E+00,
TL      = .167E+03,
TPLY    = 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
```

```
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
TPSI     = Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
           Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
           Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
TQIND    = .9E+Ø3,
TRAKLN   = .167E+Ø3, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
TRAKWD   = .28E+Ø2, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
TRANS    = .3497E+Ø1, .98E+ØØ, .1256E+Ø1, .98E+ØØ, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
           Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
           Ø.Ø, Ø.Ø,
VAA      = Ø.Ø,
VDA      = Ø.Ø,
VFS      = Ø.Ø,
VOOB     = .1E+Ø3, .1E+Ø3, .12E+Ø2, .6E+Ø1, .4E+Ø1, .4E+Ø1, .4E+Ø1, Ø.Ø,
Ø.Ø,     Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
           Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
VUOBS    = .8E-Ø1, .39E+ØØ, .77E+ØØ, .193E+Ø1, .386E+Ø1, .773E+Ø1,
.1545E+Ø2,    .3Ø91E+Ø2, .4636E+Ø2, Ø.Ø, Ø.Ø, Ø.Ø,
           Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø,
VRIDE    = .3525E+Ø2, .34E+Ø2, .2113E+Ø2, .141E+Ø2, .11E+Ø2, .1Ø75E+Ø2,
        .1Ø5E+Ø2, .1Ø25E+Ø2, .1E+Ø2, Ø.Ø, Ø.Ø,
           Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
           Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
           Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
VSS      = Ø.Ø,
VSSAXP   = Ø.Ø,
WC       = Ø.Ø,
WDAXP    = Ø.Ø,
WDPTH    = Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
WDTH     = .143E+Ø3,
WGHT     = .129E+Ø6, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
WI       = .87E+Ø2,
WRAT     = Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
WRFORD   = Ø.Ø,
WT       = .115E+Ø3, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
WTE      = .87E+Ø2, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
WWAXP    = Ø.Ø,
```

```
XBRCOF   = .8 E+ØØ,
$END
```

SAMPLE OUTPUT OF PROGRAM NRMM - VEHICLE: M60A1, TERRAIN: CKKRD

NTU	ITUT	VSEL	UP	DOWN	GRADE	DISTANCE
1	11	24.84	22.64	27.50	3.00	.1000
2	12	19.12	14.65	27.50	7.00	.1100
3	12	19.12	14.65	27.50	7.00	.1200
4	12	7.30	6.47	8.37	3.00	.1300
5	13	18.45	13.88	27.50	3.00	.1400
6	13	11.47	7.25	27.50	15.00	.1500
7	13	15.53	13.88	17.62	3.00	.1600
8	13	18.45	13.88	27.50	3.00	.1700
9	14	11.12	9.54	13.35	3.00	.1800
10	14	12.71	8.28	27.34	15.00	.1900
11	14	11.68	7.42	27.33	15.00	.2000
12	14	17.62	17.62	17.62	3.00	.2100
13	14	22.85	19.63	27.34	3.00	.2200
14	14	3.94	3.49	4.53	3.00	.2300

```
$CONTRL
DETAIL  = 2,
KSCEN   = 1,
KVEH    = 1,
KII1    = 0,
KII2    = 0,
KII3    = 0,
KII4    = 0,
KII5    = 0,
KII6    = 0,
KII7    = 0,
KII8    = 0,
KII9    = 0,
KII10   = 0,
KII11   = 0,
KII12   = 0,
KII13   = 0,
KII14   = 0,
KII15   = 0,
KII16   = 0,
KII17   = 0,
KMAP    = 0,
KTPP    = 0,
KIV1    = 0,
KIV2    = 0,
KIV3    = 0,
KIV4    = 0,
KIV5    = 0,
KIV6    = 0,
KIV7    = 0,
KIV8    = 0,
KIV9    = 0,
KIV10   = 0,
KIV11   = 0,
KIV12   = 0,
KIV13   = 0,
KIV14   = 0,
KIV15   = 0,
KIV16   = 0,
KIV17   = 0,
KIV18   = 0,
KIV19   = 0,
KIV20   = 0,
KIV21   = 0,
NTUX    = 1,
SEARCH  = 0,
$END
```

```
$SCENAR
COHES    = .5E-01,
DCLMAX   = .5E+00,
GAMMA    = .2E+00,
IOVER    = 9,
ISEASN   = 3,
ISURF    = 1,
ISNOW    = 0,
LAC      = 1,
MAP      = 71,
MAPG     = 1,
MONTH    = 1,
NOPP     = 0,
NSLIP    = 0,
NTRAV    = 3,
NTUX     = 1,
PHI      = .21E+02,
REACT    = .5E+00,
RDFOG    = .1E+04,
SFTYPC   = .9E+02,
VBRAKE   = .5E+01,
VISMNV   = .2E+01,
VLIM     = .55E+02,
ZSNOW    = .3E+01,
$END
```

```
$VEHICLE
ACD     = .12E+01,
ASHOE   = 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
AVGC    = .12E+03,
AXLSP   = .85E+02, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0,  0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
CD      = .12E+01,
CGH     = .103E+02,
CGLAT   = 0.0,
CGR     = .42E+02,
CID     = .1415E+03,
CL      = .91E+01,
CLRMIN  = .114E+02, .114E+02, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
          0.0,
CONV1   = 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
          0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
          0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
CONV2   = 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
          0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
          0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
DFLCT   = .131E+01, .114E+01, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
          0.0, .131E+01, .114E+01, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
          0.0, 0.0, .1E+01, .1E+01, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0,     0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
          0.0, 0.0, 0.0,
DIAW    = .308E+02, .308E+02, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
          0.0,
DRAFT   = 0.0,
ENGINE  = .8E+03, .115E+03, .12E+04, .115E+03, .16E+04, .115E+03,
.2E+04,         .115E+03, .24E+04, .112E+03, .28E+04,
          .108E+03, .32E+04, .103E+03, .36E+04, .96E+02, .4E+04,
.88E+02,        .44E+04, .8E+02, 0.0, 0.0, 0.0, 0.0,
          0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
          0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
          0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
          0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0,
EYEHGT  = .525E+02,
FD      = .486E+01, .9E+00,
FORDD   = 0.0,
```

```
GROUSH   = 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
HPNET    = .7392E+02,
HVALS    = .1E-01 , .4E+01, .8E+01, .12E+02, .16E+02, .2E+02, .4E+02, 0.0,
0.0,     0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
         0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
IAPG     = 0,
IB       = 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
ID       = 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
IDIESL   = .1E+01,
IENGIN   = 10,
ICONST   = 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
ICONV1   = 0,
ICONV2   = 0,
IP       = 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
IPOWER   = 17,
IT       = 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
ITCASE   = 0,
ITRAN    = 0,
ITVAR    = 0,
LOCDIF   = 0,
LOCKUP   = 0,
MAXIPR   = 9,
MAXL     = 1,
NAMBLY   = 2,
NBOGIE   = 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
NCHAIN   = 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
NCYL     = .4E+01,
NENG     = .1E+01,
NFL      = 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
NGR      = 4,
NHVALS   = 7,
NPAD     = 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
NSVALS   = 16,
NVEH     = 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
NWHL     = 2, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
NWR      = 0,
PBF      = .32E+04,
PBHT     = .2E+02,
PFA      = .225E+02,
POWER    = 0.0, .2195E+04, .49E+01, .2185E+04, .75E+01, .205E+04, .1E+02,
       .1815E+04, .101E+02, .1205E+04, .12E+02,
         .118E+04, .155E+02, .1085E+04, .198E+02, .87E+03, .199E+02,
.66E+03,  .25E+02, .65E+03, .3E+02, .615E+03,
         .33E+02, .56E+03, .331E+02, .42E+03, .4E+02, .385E+03,
.45E+02,        .355E+03, .5E+02, .34E+03, .56E+02,
         .31E+03, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0,  0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
         0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
         0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
```

```
          0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
          0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
          0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
          0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
          0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
          0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
          0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
          0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
          0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
          0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
          0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
          0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
          0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
          0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
          0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0,
QMAX    = .115E+03,
RDIAM   = .16E+02, .16E+02, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0,      0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
          0.0,
REVM    = .72E+03, .72E+03, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0,      0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
          0.0,
RIMW    = .45E+01, .45E+01, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0,      0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
          0.0,
RMS     = .25E+00, .1E+01, .2E+01, .3E+01, .4E+01, .5E+01, .6E+01,
.7E+01,        .8E+01, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
          0.0, 0.0, 0.0, 0.0, 0.0,
RW      = 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
SAE     = 0.0,
SAI     = 0.0,
SECTH   = .74E+01, .74E+01, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0,      0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
          0.0,
SECTW   = .715E+01, .715E+01, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
```

```
            0.0,
SVALS   = .1E+01, .2E+01, .3E+01, .4E+01, .5E+01, .6E+01, .7E+01,
.8E+01,        .9E+01, .1E+02, .11E+02, .12E+02, .13E+02,
            .15E+02, .2E+02, .4E+02, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0,
TCASE   = .1E+01, .1E+01,
TL      = .85E+02,
TPLY    = .6E+01, .6E+01, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0,       0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
            0.0,
TPSI    = .15E+02, .15E+02, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0,     0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
            0.0, .15E+02, .15E+02, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0,     0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
            0.0, 0.0, .25E+02, .25E+02, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0,     0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
            0.0, 0.0, 0.0,
TQIND   = 0.0,
TRAKLN  = 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
TRAKWD  = 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
TRANS   = .5712E+01, .9E+00, .3179E+01, .9E+00, .1674E+01, .9E+00,
.1E+01,        .9E+00, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
            0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
            0.0, 0.0, 0.0, 0.0, 0.0,
VAA     = .66E+02,
VDA     = .37E+02,
VFS     = 0.0,
VOOB    = .1E+03, .2E+02, .6E+01, .4E+01, .2E+01, .1E+01, .1E+01, 0.0,
0.0,      0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
            0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
VOOBS   = .15E+02, .15E+02, .15E+02, .15E+02, .15E+02, .15E+02, .15E+02,
       .15E+02, .15E+02, .15E+02, .15E+02,
            .15E+02, .15E+02, .15E+02, .15E+02, .15E+02, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
VRIDE   = .765E+02, .21E+02, .95E+01, .51E+01, .2E+01, .2E+01, .2E+01,
.2E+01,  .2E+01, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
            0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
            0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
            0.0, 0.0, 0.0,
VSS     = 0.0,
VSSAXP  = 0.0,
WC      = 0.0,
WDAXP   = 0.0,
WDPTH   = 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
WDTH    = .64E+02,
WGHT    = .174E+04, .146E+04, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
```

```
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
          0.0,
WI      = .458E+02,
WRAT    = 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
WRFORD  = 0.0,
WT      = .53E+02, .53E+02, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0,   0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
          0.0,
WTE     = .456E+02, .456E+02, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
          0.0,
WWAXP   = 0.0,
XBRCOF  = .7E+00,
$END
```

SAMPLE OUTPUT OF PROGRAM NRMM - VEHICLE: M151, TERRAIN: CKK

NTU	ITUT	VMAX	UP	LEVEL	DOWN	VSEL	UP	LEVEL	DOWN	GRADE	AREA
1	1	37.15	37.15	37.15	37.15	37.15	37.15	37.15	37.15	1.00	0.0000
2	1	37.15	37.15	37.15	37.15	37.15	37.15	37.15	37.15	1.00	0.0000
3	1	37.15	37.15	37.15	37.15	37.15	37.15	37.15	37.15	1.00	0.0000
4	1	33.26	30.03	33.66	36.79	33.26	30.03	33.66	36.79	1.00	0.0000
5	1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.0000
6	1	37.15	37.15	37.15	37.15	37.15	37.15	37.15	37.15	3.50	0.0000
7	1	37.15	37.15	37.15	37.15	37.15	37.15	37.15	37.15	7.50	0.0000
8	1	30.91	23.13	37.15	37.15	30.91	23.13	37.15	37.15	15.00	0.0000
9	1	18.99	11.26	37.15	23.66	18.99	11.26	37.15	23.66	50.00	0.0000
10	1	0.00	0.00	37.15	10.26	0.00	0.00	37.15	10.26	65.00	0.0000
11	1	37.15	37.15	37.15	37.15	37.15	37.15	37.15	37.15	1.00	0.0000
12	1	17.64	16.67	17.71	18.67	17.64	16.67	17.71	18.67	1.00	0.0000
13	1	12.72	12.67	12.72	12.76	12.72	12.67	12.72	12.76	1.00	0.0000
14	1	8.90	8.45	8.92	9.35	8.90	8.45	8.92	9.35	1.00	0.0000
15	1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.0000
16	1	33.80	28.64	37.15	37.15	33.80	28.64	37.15	37.15	3.50	0.0000
17	1	31.04	23.36	37.15	37.15	31.04	23.36	37.15	37.15	7.50	0.0000
18	1	29.41	20.76	37.15	37.15	29.41	20.76	37.15	37.15	15.00	0.0000
19	1	0.00	0.00	37.15	27.50	0.00	0.00	37.15	27.50	50.00	0.0000
20	1	0.00	0.00	37.15	17.57	0.00	0.00	37.15	17.57	65.00	0.0000
21	1	37.15	37.15	37.15	37.15	37.15	37.15	37.15	37.15	1.00	0.0000
22	1	9.50	9.50	9.50	9.50	9.50	9.50	9.50	9.50	1.00	0.0000
23	1	2.00	2.00	2.00	2.00	2.00	2.00	2.00	2.00	1.00	0.0000
24	1	2.00	2.00	2.00	2.00	2.00	2.00	2.00	2.00	1.00	0.0000
25	1	2.00	2.00	2.00	2.00	2.00	2.00	2.00	2.00	1.00	0.0000
26	1	24.73	24.73	24.73	24.73	24.73	24.73	24.73	24.73	1.00	0.0000
27	1	24.73	24.73	24.73	24.73	24.73	24.73	24.73	24.73	1.00	0.0000
28	1	18.54	18.54	18.54	18.54	18.54	18.54	18.54	18.54	1.00	0.0000
29	1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.0000
30	1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.0000
31	1	24.52	24.52	24.52	24.52	24.52	24.52	24.52	24.52	1.00	0.0000
32	1	22.96	22.83	22.95	23.11	22.96	22.83	22.95	23.11	1.00	0.0000
33	1	18.36	18.36	18.36	18.36	18.36	18.36	18.36	18.36	1.00	0.0000
34	1	24.72	24.72	24.72	24.72	24.72	24.72	24.72	24.72	1.00	0.0000
35	1	24.54	24.19	24.72	24.72	24.54	24.19	24.72	24.72	1.00	0.0000
36	1	37.15	37.15	37.15	37.15	37.15	37.15	37.15	37.15	1.00	0.0000
37	1	37.14	37.14	37.14	37.14	37.14	37.14	37.14	37.14	1.00	0.0000
38	1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.0000
39	1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.0000
40	1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.0000
41	1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.0000
42	1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.0000
43	1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.0000
44	1	34.68	34.68	34.68	34.68	34.68	34.68	34.68	34.68	1.00	0.0000
45	1	22.83	22.83	22.83	22.83	22.83	22.83	22.83	22.83	1.00	0.0000
46	1	10.57	10.57	10.57	10.57	10.57	10.57	10.57	10.57	1.00	0.0000
47	1	34.68	34.68	34.68	34.68	34.68	34.68	34.68	34.68	1.00	0.0000
48	1	22.83	22.83	22.83	22.83	22.83	22.83	22.83	22.83	1.00	0.0000
49	1	10.57	10.57	10.57	10.57	10.57	10.57	10.57	10.57	1.00	0.0000
50	1	5.84	5.84	5.84	5.84	5.84	5.84	5.84	5.84	1.00	0.0000

SAMPLE OUTPUT OF PROGRAM NRMM - VEHICLE: M151, TERRAIN: CKKRD

```
$CONTRL
DETAIL  = 2,
KSCEN   = 1,
KVEH    = 1,
KII1    = 0,
KII2    = 0,
KII3    = 0,
KII4    = 0,
KII5    = 0,
KII6    = 0,
KII7    = 0,
KII8    = 0,
KII9    = 0,
KII10   = 0,
KII11   = 0,
KII12   = 0,
KII13   = 0,
KII14   = 0,
KII15   = 0,
KII16   = 0,
KII17   = 0,
KMAP    = 0,
KTPP    = 0,
KIV1    = 0,
KIV2    = 0,
KIV3    = 0,
KIV4    = 0,
KIV5    = 0,
KIV6    = 0,
KIV7    = 0,
KIV8    = 0,
KIV9    = 0,
KIV10   = 0,
KIV11   = 0,
KIV12   = 0,
KIV13   = 0,
KIV14   = 0,
KIV15   = 0,
KIV16   = 0,
KIV17   = 0,
KIV18   = 0,
KIV19   = 0,
KIV20   = 0,
KIV21   = 0,
NTUX    = 1,
SEARCH  = 0,
$END
$SCENAR
COHES   = .5E-01,
DCLMAX  = .5E+00,
GAMMA   = .2E+00,
IOVER   = 9,
```

```
ISEASN   = 3,
ISURF    = 1,
ISNGW    = 0,
LAC      = 1,
MAP      = 11,
MAPG     = 1,
MONTH    = 1,
NOPP     = 0,
NSLIP    = 0,
NTRAV    = 3,
NTUX     = 1,
PHI      = .21E+02,
REACT    = .5E+00,
RDFOG    = .1E+04,
SFTYPC   = .9E+02,
VBRAKE   = .5E+01,
VISMNV   = .2E+01,
VLIM     = .55E+02,
ZSNCW    = .3E+01,
$END
$VEHICLE
ACD      = .12E+01,
ASHOE    = 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
AVGC     = .12E+03,
AXLSP    = .85E+02, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0,  0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
CD       = .12E+01,
CGH      = .103E+02,
CGLAT    = 0.0,
CGR      = .42E+02,
CID      = .1415E+03,
CL       = .91E+01,
CLRMIN   = .114E+02, .114E+02, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
           0.0,
CONV1    = 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
           0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
           0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
CONV2    = 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
           0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
           0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
DFLCT    = .131E+01, .114E+01, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
           0.0, .131E+01, .114E+01, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
           0.0, 0.0, .1E+01, .1E+01, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0,       0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
```

```
               0.0, 0.0, 0.0,
DIAW     = .308E+02, .308E+02, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
               0.0,
CRAFT    = 0.0,
ENGINE   = .8E+03, .115E+03, .12E+04, .115E+03, .16E+04, .115E+03,
.2E+04,         .115E+03, .24E+04, .112E+03, .28E+04,
               .108E+03, .32E+04, .103E+03, .36E+04, .98E+02, .4E+04,
.88E+02,        .44E+04, .8E+02, 0.0, 0.0, 0.0, 0.0,
               0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
               0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
               0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
               0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0,
EYEHGT   = .525E+02,
FD       = .486E+01, .9E+00,
FORDD    = 0.0,
GROUSH   = 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
HPNET    = .7392E+02,
HVALS    = .1E-01, .4E+01, .8E+01, .12E+02, .16E+02, .2E+02, .4E+02, 0.0,
0.0,    0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
               0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
IAPG     = 0,
IB       = 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
ID       = 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
IDIESL   = .1E+01,
IENGIN   = 10,
ICONST   = 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
ICONV1   = 0,
ICONV2   = 0,
IP       = 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
IPOWER   = 17,
IT       = 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
ITCASE   = 0,
ITRAN    = 0,
ITVAR    = 0,
LOCDIF   = 0,
LOCKUP   = 0,
MAXIPK   = 9,
MAXL     = 1,
NAMBLY   = 2,
NBOGIE   = 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
NCHAIN   = 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
NCYL     = .4E+01,
NENG     = .1E+01,
NFL      = 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
NGR      = 4,
NHVALS   = 7,
```

```
NPAD     = Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø,
NSVALS   = 16,
NVEH     = 1, 1, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø,
NWHL     = 2, 2, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø, Ø,
NWR      = Ø,
PBF      = .32E+Ø4,
PBHT     = .2E+Ø2,
PFA      = .225E+Ø2,
POWER    = Ø.Ø, .2195E+Ø4, .49E+Ø1, .2185E+Ø4, .75E+Ø1, .2Ø5E+Ø4, .1E+Ø2,
        .1815E+Ø4, .1Ø1E+Ø2, .12Ø5E+Ø4, .12E+Ø2,
           .118E+Ø4, .155E+Ø2, .1Ø85E+Ø4, .198E+Ø2, .87E+Ø3, .199E+Ø2,
.66E+Ø3,   .25E+Ø2, .65E+Ø3, .3E+Ø2, .615E+Ø3,
           .33E+Ø2, .56E+Ø3, .331E+Ø2, .42E+Ø3, .4E+Ø2, .385E+Ø3,
.45E+Ø2,        .355E+Ø3, .5E+Ø2, .34E+Ø3, .56E+Ø2,
           .31E+Ø3, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø, Ø.Ø,  Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
           Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
           Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
           Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
           Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
           Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
           Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
           Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
           Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
           Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
           Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
           Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
           Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
           Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
           Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
           Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
           Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
           Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
Ø.Ø,
QMAX     = .115E+Ø3,
RDIAM    = .16E+Ø2, .16E+Ø2, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø, Ø.Ø,
```

```
0.0,     0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
            0.0,
REVM     = .72E+03, .72E+03, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0,     0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
            0.0,
RIMW     = .45E+01, .45E+01, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0,     0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
            0.0,
RMS      = .25E+00, .1E+01, .2E+01, .3E+01, .4E+01, .5E+01, .6E+01,
.7E+01,       .8E+01, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
            0.0, 0.0, 0.0, 0.0, 0.0,
RW       = 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
SAE      = 0.0,
SAI      = 0.0,
SECTH    = .74E+01, .74E+01, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0,     0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
            0.0,
SECTW    = .715E+01, .715E+01, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
            0.0,
SVALS    = .1E+01, .2E+01, .3E+01, .4E+01, .5E+01, .6E+01, .7E+01,
.8E+01,        .9E+01, .1E+02, .11E+02, .12E+02, .13E+02,
            .15E+02, .2E+02, .4E+02, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0,
TCASE    = .1E+01, .1E+01,
TL       = .85E+02,
TPLY     = .6E+01, .6E+01, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0,      0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
            0.0,
TPSI     = .15E+02, .15E+02, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0,     0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
            0.0, .15E+02, .15E+02, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0,     0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
            0.0, 0.0, .25E+02, .25E+02, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0,     0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
            0.0, 0.0, 0.0,
TQIND    = 0.0,
TRAKLN   = 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
TRAKWD   = 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
TRANS    = .5712E+01, .9E+00, .3179E+01, .9E+00, .1674E+01, .9E+00,
.1E+01,        .9E+00, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
            0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
            0.0, 0.0, 0.0, 0.0, 0.0,
VAA      = .66E+02,
VDA      = .37E+02,
VFS      = 0.0,
VOOB     = .1E+03, .2E+02, .65E+01, .4E+01, .2E+01, .1E+01, .1E+01, 0.0,
0.0,      0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
```

```
              0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
VOOBS    = .15E+02, .15E+02, .15E+02, .15E+02, .15E+02, .15E+02, .15E+02,
        .15E+02, .15E+02, .15E+02, .15E+02,
              .15E+02, .15E+02, .15E+02, .15E+02, .15E+02, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
VRIDE    = .765E+02, .21E+02, .95E+01, .51E+01, .2E+01, .2E+01, .2E+01,
.2E+01,  .2E+01, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
              0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
              0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
              0.0, 0.0, 0.0,
VSS      = 0.0,
VSSAXP   = 0.0,
WC       = 0.0,
WDAXP    = 0.0,
WDPTH    = 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
WDTH     = .64E+02,
WGHT     = .174E+04, .146E+04, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
              0.0,
WI       = .458E+02,
WRAT     = 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
WRFCRD   = 0.0,
WT       = .53E+02, .53E+02, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0,   0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
              0.0,
WTE      = .456E+02, .456E+02, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
              0.0,
WWAXP    = 0.0,
XBRCOF   = .7E+00,
$END
 NTU LTUT  VSEL    UP  DOWN GRADE DISTANCE
   1   11 54.94 54.87 55.00  3.00     .1000
   2   12 40.00 40.00 40.00  7.00     .1100
   3   12 49.62 45.19 55.00  7.00     .1200
   4   12 25.00 25.00 25.00  3.00     .1300
   5   13 35.80 35.80 35.80  3.00     .1400
   6   13 32.24 29.32 35.80 15.00     .1500
   7   13  7.30  7.30  7.30  3.00     .1600
   8   13 35.80 35.80 35.80  3.00     .1700
   9   14 35.39 34.99 35.80  3.00     .1800
  10   14 28.22 23.29 35.80 15.00     .1900
  11   14 28.00 22.99 35.80 15.00     .2000
  12   14  7.30  7.30  7.30  3.00     .2100
  13   14 21.00 21.00 21.00  3.00     .2200
  14   14  9.00  9.00  9.00  3.00     .2300
```

APPENDIX E

POSSIBLE STOPS IN PROGRAM NRMM

The program NRMM can terminate execution prior to normal completion for several reasons. In most such cases, an octal number is available to indicate at which of the STOPs the program halted.

NUMBER	LOCATION	REASON FOR STOP
1	SCN	User erroneously specified control variable DETAIL.
2	VEH	Towed track elements indicated in vehicle data are not permitted.
3	VPP	Printer plot of tractive effort vs. speed has been produced because either user specified DETAIL = 5 level of output or the tractive effort vs. speed curve fit error has been exceeded.
4	LINEAR	Interpolation routine LINEAR requires powertrain data to be inserted in ascending order of magnitude.
5	LINEAR	A calculated point is outside the bounds of the array to be interpolated. (Check powertrain data for errors.)
6	IV3	Terrain soil value, IST, does not conform to the soil types addressed in the Model. (Check terrain data.)
7	IV3	For tracked vehicles, slipperiness scenario variable NSLIP is outside the range used (0-6). (Check scenario data.)
10	IV3	Same as 7 except for non-CH soil type.
11	IV3	Same as 7 except for wheeled vehicles.
12	TFORCF	Soil type variable IST has erroneous value. Relationships for modifying the drawbar pull curve at 20% slip vs. soil type are available only for fine grained, coarse grained and muskeg soils. (Trace passage of variable IST and its value through the program.)

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER 12503	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) NATO REFERENCE MOBILITY MODEL, EDITION I USERS GUIDE, VOLUME I		5. TYPE OF REPORT & PERIOD COVERED Final
		6. PERFORMING ORG. REPORT NUMBER SIT-DL-79-9-2058
7. AUTHOR(s) M. Peter Jurkat, Peter M. Brady, Jr., Peter W. Haley		8. CONTRACT OR GRANT NUMBER(s) DAAK30-77-C-0027
9. PERFORMING ORGANIZATION NAME AND ADDRESS Stevens Institute of Technology, Davidson Lab, Castle Point Station, Hoboken, NJ 07030		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS US Army Tank-Automotive Rsch & Dev Command, DRDTA-ZSA, Warren, MI 48090		12. REPORT DATE OCT 1979
		13. NUMBER OF PAGES 390
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

Includes: Operational Modules; App A: Program Listing; App B: Vehicle Input Files; App C: Terrain Input Files; App D: Sample Output of Program; App E: Possible STOP's In Program.

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Mobility
Mobility Modeling
Computerized Simulation
Vehicle Performance
Terrain

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

Instructions in the organization and use of the computer programs which implement the Initial NATO Reference Model (INRMM) are presented. A brief description of the mathematical equations and computing algorithms which predict the speed of a vehicle over a variety of terrain, the input data required, and the outputs generated is included. Some aid to the interpretation of various output variables is given.

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

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)