AVIATION TECHNOLOGY RESEARCH PLAN
ROTARY WING PROGRAM
DEVELOPMENT AND EVALUATION OF PERFORMANCE AND HANDLING QUALITIES CRITERA FOR LOW COST REAL TIME ROTORCRAFT SIMULATORS - PHASE I
METHODOLOGY DEVELOPMENT

J.V.R. Prasad
D.P. Schrage
W. Lewis
D. Wolfe
C.M. McKeithan
C. Gardner

Final Report
Prepared for
The Institute for Simulation and Training
University of Central Florida
Purchase Order No. 101700

SCHOOL OF AEROSPACE ENGINEERING
GEORGIA INSTITUTE OF TECHNOLOGY
ATLANTA, GEORGIA 30332
MARCH 1991

TABLE OF CONTENTS

SUMMARY

The value of rotorcraft simulators through increased safety, reduced operating/training costs and enhanced mission training has been well documented in the past twenty years. Due to the increased emphasis in rotorcraft simulation, the Federal Aviation Administration (FAA) has launched a program to establish certification standards for rotorcraft simulators. This program is aimed at updating both rotorcraft simulator standards and the methods of simulator validation through objective and subjective tests. In addition, the Defence Advanced Research Project Agency (DARPA) and the U.S. Army Program Manager for Training Devices (PM-TRADE) are developing an aviation technology research plan to investigate fidelity issues in aviation simulators and networking. No established methodology and acceptance criteria currently exist for the performance and handling qualities assessment of rotorcraft simulators. In order to establish certification criteria, a planned research effort to quantify the system capabilities of "selective fidelity" simulators is required. This report addresses the initial step toward that goal: the establishment of a methodology for defining the performance and handling qualities acceptance criteria for selective fidelity, real time rotorcraft simulators. Within this framework the simulator is then classified based on the required task. The simulator is evaluated by separating the various subsystems (visual, motion, etc.) and applying corresponding fidelity constants based on the specific task. This methodology not only provides an assessment technique, but also provides a technique to determine the required levels of subsystem fidelity for a specific task. This approach provides a helpful tool for eliminating system suboptimization and for identifying critical research areas.

BACKGROUND

General

In recent years, greater emphasis is being placed on the use of simulation as a design tool for new aircraft development. This is exemplified by the US Government's insistence that the Army's Light Helicopter (LH) design concept should be validated through simulation before a prototype is built. Also, the FAA has initiated a program to establish an update of rotorcraft simulator standards and the methods of validation through both objective and subjective tests. The results of this program will be used by the FAA in its qualification of rotorcraft simulators. Experience has shown that a piloted simulation is essential for safety before first flight. In addition, the extremely high cost of operating actual aircraft makes the use of simulators much more attractive as training tools. With the emphasis by the Defence Department on networking of low cost simulators to enhance warfighting capability, DARPA and PM-TRADE have initiated an aviation technology research plan aimed at investigating fidelity issues in low cost aviation simulators and networking. The Georgia Tech Center of Excellence for Rotary Wing Aircraft Technology (CERWAT) has been working with the Institute for Simulation and Training (IST) at the University of Central Florida in formulating a methodology for implementation of these plans. An annotated bibliography from the literature search conducted as part of this study is included as Appendix A.

Simulators are routinely used for pilot training and initial evaluation of pilots. In addition, the use of simulation as a design tool for new aircraft development is being well recognized in the rotorcraft community [1]. The major advantage of the use of simulation as a design tool is that one can evaluate a configuration very early in the design process to see how it functions and what changes may be required. The major human-factor problems as well as performance problems can be corrected in the simulator before the first prototype is built resulting in substantial savings of both time and money.

There is a definitive need for a planned research effort to understand the quantitative and qualitative performance and system capabilities of "selective fidelity" simulators [2]. Selective fidelity is defined to mean both the objective and perceptual fidelity aspects of rotorcraft simulation. Perceptual fidelity provides a psychological/physiological viewpoint and it is the degree to which the trainee subjectively perceives the simulator to reproduce it real-life counterpart aircraft in flight [3]. Objective fidelity provides an engineering viewpoint and it is the degree to which a simulator would be observed to reproduce its real-life counterpart aircraft in flight [3]. The quantitative aspects include time delays (both internal and external to the simulator), matching aerodynamic characteristics between simulator and actual vehicle (where data are available), static stability matching between simulator and flight vehicle, dynamic stability matching studies, and simulated performance evaluations using FAA and Naval Air Test Center (NATC) evaluation criteria. The qualitative aspects deal with pilot acceptance of the rotorcraft simulation for its intended use through functional and subjective tests. The use of the accepted Cooper-Harper scale provides this bridge for evaluation of fixed-wing aircraft. A similar bridge is sought for rotorcraft simulators. The ultimate goal of such a planned effort is to establish simulator flying qualities and criteria for pilot acceptance of the simulation. Such criteria can be used to reduce simulator "pilot tailoring" by a Fleet Project Team and allow simulators to be built as true engineering models of yet to be built aircraft.

In developing a methodology our task becomes twofold; define rotorcraft simulators in terms of fidelity and then apply data collection techniques to evaluate performance and handling qualities. With respect to fidelity, the current thrust of minimizing training costs focuses attention on the question, "What is the required level of fidelity?" As a general rule, procurement of new simulation devices or updating existing models consisted of fulfilling a wish list. If a state-of-the-art system was desired, state-of-the-art subsystems were procured and integrated. It would not be inconceivable to have a high fidelity visual and motion system coupled with a somewhat simplistic math model. After investing millions of dollars into the system, the pilot comments were still unfavorable, i.e., " a very nice procedural trainer, but it just doesn't fly like the aircraft." In this case the system integrator has suboptimized the system. Unfortunately, there lacks a quantitative methodology for defining a required level of fidelity for a given simulation task. A methodology for assessing selective fidelity simulators would provide the systems integrator with acceptance criteria and aid in preventing system suboptimization by defining required subsystem fidelity for a specific task. This paper proposes to approach this problem by defining a task specific simulator classification system based on fidelity. With respect to applying data collection techniques for evaluating handling qualities, ADS33, the emerging standard in helicopter handling qualities [4], coupled with the U.S. Army Light Helicopter (LH) Demonstration/Validation Phase test results will be used to define:

1. quantitative evaluation criteria: In general, data collection focuses on quantifiable items such as bandwidth, minimum and peak rates, and damping ratios that are useful in defining acceptable tolerances between actual flight data and simulation data.

2. qualitative evaluation criteria: In general, a rating scale system for a specified set of tasks is outlined for pilot acceptance of the simulation.

Simulation Tasks and Fidelity

As depicted in Figure 1, the fidelity requirement for any simulation device is inherently dependent upon the given simulation task. The procedure illustrated in Figure 1 is a top down design decision process where the user simulation requirements (customer voice) are the identified need. The problem is defined in terms of a simulator task and selection of the type of device required. Value objectives are established as feasibility constraints and a task dependent weighting factor. Feasible simulation alternatives are generated in terms of fidelity requirements. By evaluating the alternatives against the established value objectives a decision can be made to determine which simulator can be used for a specific simulator task. This decision making process is summarized in Table 1. The flow in Figure 1 is similar to the recommended approach in Reference [3].

Table 1. Methodology for simulator fidelity selection based on user simulation requirements.

Decision Making Process	Methodology
Identify the need	Identify user simulation requirements
Define the problem	Define simulation task
Establish value objectives	Establish minimum fidelity characteristics (Quantitative evaluation criteria)
	Establish a simulation task dependent weighting factor (Qualitative Evaluation criteria)
Generate feasible alternatives	Generate feasible simulation devices based on simulation classification model
Evaluate alternatives	Evaluate feasible simulation devices against value objectives
Make decision	Choose the best simulation device for the identified user simulation requirement

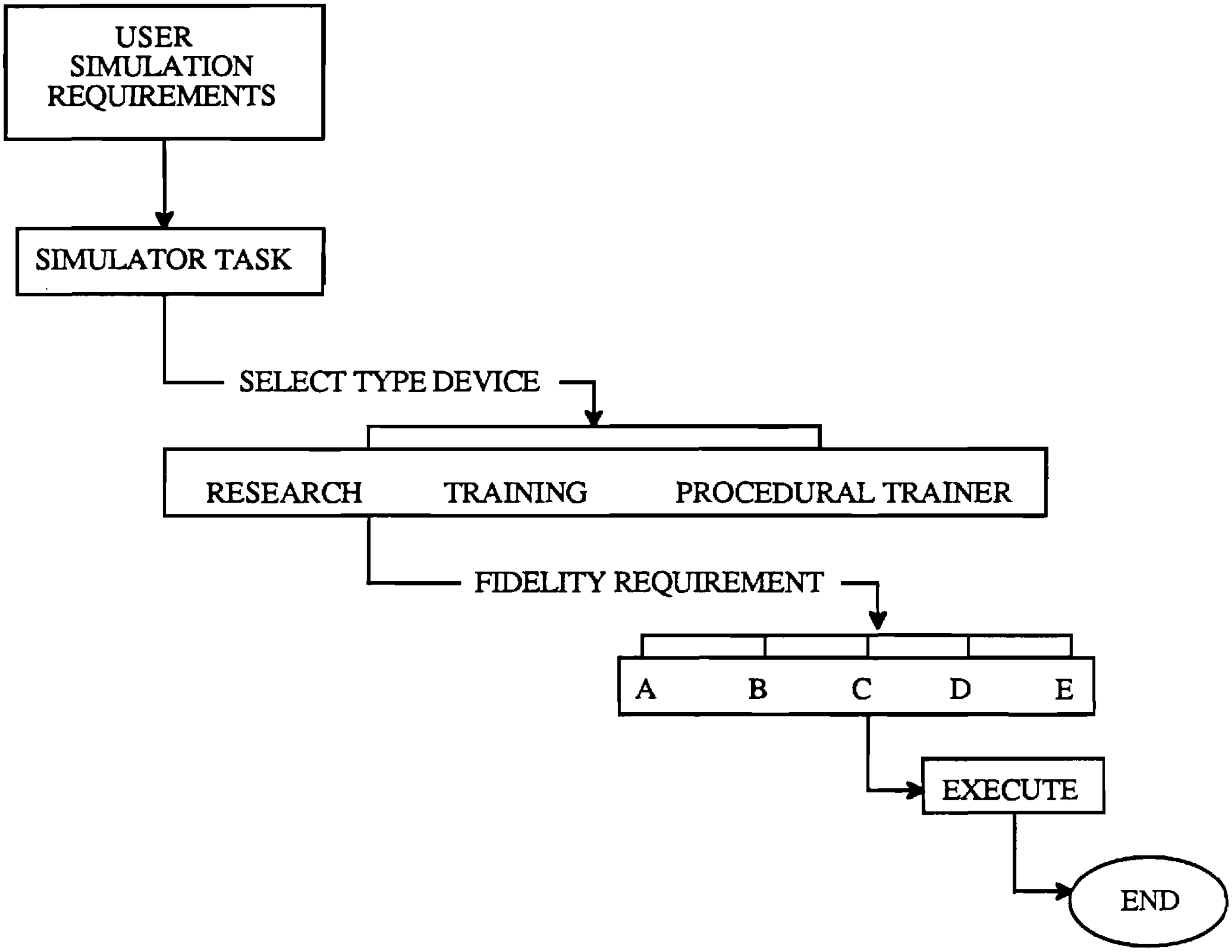


Figure 1. Fidelity dependence on type task

The requirements for simulators in the civil and military fields has expanded greatly throughout the past decade. Along with that growth, the variety of simulation tasks has also increased. Tasks can be categorized as;

(1) non real time research analysis
(2) part task simulation
(3) part mission
(4) full mission
(5) interactive mission scenario (networked, multiple nodes)

Encompassing these tasks, simulation devices can be broadly categorized into three types; research, training, and procedural trainers. Among these categories of simulators, there is a tradeoff between equipment and environmental cue fidelity required [3]. Equipment cues provide a duplication of the appearance and feel of the operational aircraft. Environmental cues provide a duplication of the environment and motion through the environment. For example, procedural trainers require high level of equipment cue fidelity and research simulators require high level of environmental cue fidelity. Within these broad simulation types, the levels of fidelity for a type device can vary greatly. For example, using cockpit crew coordination as our simulation task, a work station can be defined as a relatively low fidelity research simulator. Yet, another simulator of the same type, such as the Crew Station Research and Development Facility (CSRDF) located at NASA-Ames, certainly has a higher level of fidelity for the same task. Thus, for a specified task, the user must be able to determine fidelity requirements. Failure to properly determine these requirements can result in:

(1) unsatisfactory results due to lack of fidelity
(2) satisfactory results but at premium cost
(suboptimization)

Consequently, it is desirable to classify a simulation device in terms of its fidelity. This allows a user with defined task specific fidelity requirements to select a simulator of appropriate fidelity and eliminate the above problems. The Federal Aviation Administration (FAA), for example, qualifies airplane training simulators in terms of objective fidelity. Simulator classification by fidelity sets a basis from which the user community can identify the specific simulation device that is optimized for their needs.

FAA Simulator Qualification

The current FAA approach for the qualification of airplane simulators is embodied in FAA AC-120-40B [5]. A similar approach is being planned by the FAA for qualification of rotorcraft simulators. The FAA approach designates simulators into four categories, A through D, based on increasing levels of objective fidelity. Simulator standards, objective validation tests, and functional and subjective tests are then defined for each category. For airplanes, the standards, validation tests, and functional and subjective tests have been fairly well accepted by industry through a series of workshops. Rotorcraft simulators do not have such well defined standards due to the unique capabilities and complexities of the air vehicle and existing simulation technology. Development of the rotorcraft criteria will require extensive research and development.

SIMULATOR CLASSIFICATION MODEL

General

Unlike the FAA approach to simulator classification, this methodology quantitatively classifies a given type simulation device in terms of objective fidelity and a simulation task dependent weighting vector (TDWV). Each TDWV consists of a weighting parameter per fidelity characteristic, i.e.

$$[\ \mathrm{TDWV}\ ] = [\ K_{\mathrm{cockpit}}\ \ K_{\mathrm{audio}}\ \ K_{\mathrm{motion}}\ \cdots\ K_{\mathrm{visual}}\ ]$$

For example, an air-to-air combat task requires a significant weighting parameter for the visual characteristic, whereas, the instrument training task would not require as large a weighting parameter for the visual characteristic. Clearly, in general terms, the weighting vector will always be dependent upon the simulation task to be performed. The fidelity of the simulation device is assessed by rating each component of the system. For the purposes of this methodology, a simulation device is described in terms of ten subsystems, with each subsystem having varying degrees of sophistication and thus, fidelity.

Subsystems and Fidelity

In surveying current simulation devices and existing technologies, there are generally ten subsystems which adequately describe a given simulation device:

(a) Cockpit
(b) Audio
(c) Motion
(d) Control System
(e) Math model
(f) Environment
(g) Ground handling
(h) Mission equipment
(i) System Latency
(j) Visual

In each subsystem, it is possible to associate a level of objective fidelity with the degree of equipment/software sophistication. For example, a motion system that employs six degrees of freedom can be associated with high fidelity, while a fixed base system can be associated with low fidelity. This association between fidelity and the subsystems defines *fidelity characteristics*. Subsequently, listed below are the fidelity characteristics (rank order; none to high) of the simulator subsystems that span the spectrum of fidelity.

(a) Cockpit / Crew Station
- none
- simulated/generic type instruments
- partially simulated cockpit
- full up crew station

(b) Audio
- none
- significant cockpit sounds
- incidental sounds (precip., etc...)
- realistic

(c) Motion
- none
- 2DOF (pitch and roll)
- 3DOF (pitch, roll, and yaw)
- 6DOF

d) Control System
- no force feel
- constant force (spring/damper)
- partial duplication of actual force
- complete duplication

(e) Mathematical Model
- none
- 3 DOF
- 6 DOF
- 6 DOF with rotor

(f) Environmental
- clean air
- discrete gusts
- first order filtered turbulence
- rotationally sampled turbulence

(g) Ground Handling
- no gear
- rigid gear
- simplified gear model
- comprehensive

(h) Mission Equipment
- none
- communication only
- communication/navigation only
- complete

(i) System Latency
- non real time (off line)
- significant delay
- minimal delays
- real time

(j) Visual
- none

- field of view	dynamic range	detail
workstation	day	low
75°horiz/30°vert	dusk	medium
90°horiz/40°vert	haze/fog	high
wider	night	very high

Assigning a value to each fidelity characteristic of the simulation device allows us to quantify fidelity by forming the Simulator Fidelity Constants (SFC) vector. A logarithmic scale of 0 to 10 is used to assign a value to each of the fidelity characteristic; 0 corresponding to the case of a missing characteristic and 10 corresponding to the highest level of representation of a particular characteristic. For illustration purposes, a value of 5 is used for the case of partial representation of a characteristic and 8 is used for moderate representation. For example, the U.S. Army 2B38 UH-60 simulator has the following characteristics:

(a) Cockpit - full up crew station
(b) Audio - incidental sounds
(c) Motion - 6D
(d) Control System - complete duplication
(e) Math model - 6DOF w/simple rotor

(f) Environment - discrete gusts
(g) Ground handling - simple gear model
(h) Mission equipment - complete
(i) System Latency - real time
(j) Visual - 90^ohoriz/40^overt
full dynamic range
medium detail

With the above characteristics, the UH60 training simulator's fidelity constants vector is:

$$[\ SFC\ ]^T = [\ C_{cockpit}, C_{aud}, C_{mot}, C_{feel}, C_{math}, C_{env}, C_{grnd}, C_{mep}, C_{lat}, C_{vis}\ ]$$

$$= [\ 10\ 8\ \ 10\ \ 10\ \ 8\ \ 5\ \ 8\ \ 10\ \ 10\ \ 8\ ]$$

The definition used for the SFC implies that no coupling exists between the various fidelity characteristics. Anyone who has flown in a simulator with a high fidelity visual system employing infinity collimation knows this to be untrue. With a fixed base motion system, the aforementioned visual system will cause a perceived motion. The strength of the perceived motion will vary depending on the fidelity of the visual system. This example would indicate some degree of coupling between the visual and motion characteristics. This interdependence may be better represented by use of a coupling matrix, [C]. In order to account for the coupling between the various fidelity characteristics, we define a Perceived Simulator Fidelity Constants (PSFC) vector which is obtained by premultiplying the SFC with C.

$$\{PSFC\} = [C]*\{SFC\}$$

where $[C] = \begin{matrix} [\ c_{11} & c_{12} \cdots\cdots \\ c_{21} & c_{22} \cdots\cdots \\ \cdots\cdots & \cdots etc.] \end{matrix}$

The exact form of the coupling would need to be determined through research.

Minimum Fidelity Characteristics

For a given simulation task, minimum acceptable fidelity characteristics must be established in order to constrain the number of feasible simulation devices eligible to perform the task. For example, to conduct aircrew contact training, some form of visual system is a minimum requirement for the visual fidelity characteristic. Without a visual system, the device would be unable to adequately provide task training. Consequently, a minimum required Simulation Fidelity Constants (SFC_{min}) vector:

$$[\ SFC_{min}\ ]^T = \{\ \min[\ C_{cockpit}\ \ C_{audio}\ \ C_{motion} \cdots\ C_{visual}]\ \}$$

is utilized to establish the minimum acceptable fidelity characteristics for a given task. Similarly, a vector of maximum desirable fidelity characteristics (SFC_{max}) is established for a given task. Exemplifying this concept, the U.S. Army 2B24 instrument training simulator, although it has many high fidelity characteristics such as a 6DOF motion system, full up cockpit, and a complete mission package, is not eligible for consideration as a simulator for contact training because it lacks a visual system.

Simulator Classification

The function:

$$SIMRATING_{task(i)} = [\ TDWV_{task(i)}\ ]*[PSFC]$$

with the individual elements of [PSFC] constrained by:

$$[\ SFC_{min}\ ]^T = \{\ \min[\ C_{cockpit}\ \ C_{audio}\ \ C_{motion} \cdots\ C_{visual}]\ \}$$

permits classification of a type simulation device with respect to fidelity. Given a simulation task, SFC_{min} vector and SFC_{max} vectors are determined either subjectively or through extensive research. Once the weighting vector is known, a minimum and maximum $SIMRATING_{task(i)}$ is calculated. Given this range of values, the simulation devices can be classified in terms of fidelity for a specified task. The range of values is partitioned into five subranges, the lowest corresponding to poor fidelity and the highest corresponding to high fidelity.

As an example, suppose the given task is instrument training and the hypothetical SFC_{min} and TDWV have been determined to be:

$$[\ SFC_{min}\ ]^T = [\ 10\ \ 0\ \ 0\ 5\ 0\ 0\ 5\ 5\ 8\ 0\ ]$$

$$[\,TDWV\,] = [\,1\ \ .5\ \ .5\ \ 1\ \ .5\ \ .5\ \ .25\ \ .75\ \ 1\ \ 0\,]$$

Multiplying $[TDWV]*[SFC_{min}]$ we find the minimum $SIMRATING_{task(i)}$ to be 28. For the maximum $SIMRATING_{task(i)}$, we must multiply

$$[TDWV]*[\,SFC_{max}\,]$$

where the maximum fidelity constant matrix $[SFC_{max}]$ for the hypothetical case is assumed to be

$$[\,SFC_{max}\,] = [\,10\ \ 10\ \ ....\ 10\,].$$

Thus the maximum $SIMRATING_{task(i)}$ is calculated to be 60. Partitioning this range of values, we can now form a task specific (instrument training) classification for simulation devices based on fidelity. For this example:

FIDELITY	CLASSIFICATION	SIMRATING
Excellent	A	57 - 60
High	B	51 - 56
Medium	C	44 - 50
Low	D	36 - 43
Lowest	E	28 - 35

Neglecting coupling between various fidelity characteristics, i.e., taking [C] to be an identity matrix, the task dependent simulator rating for the U.S. Army 2B38 UH-60 simulator is 55 to perform instrument training task and it would receive B (high fidelity) classification based on the methodology proposed in this report.

METHODOLOGY EXTENSIONS

Fidelity Characteristics

Within the scope of this methodology framework, each of the fidelity characteristics was limited to four levels (none to high) with corresponding ratings of 0, 5, 8, and 10, in order to provide an equivalent weighting between characteristics. This general approach obviously cannot handle specifics of any single characteristic. An alternate approach to provide equivalent weighting between characteristics is to employ normalization for each characteristic. This approach would allow a greater degree of flexibility in assessing each characteristic. For example, while assessing the visual system the user could include the use of texture, infinity collimation, display types, etc., over a wider range of values. This enables the visual system characteristic to be well defined in terms of it's specific attributes.

QUANTITATIVE EVALUATION CRITERIA

General

The methodology at this point allows categorization of rotorcraft simulator in terms of fidelity for a specific task, but leaves unanswered the means of evaluating the performance and handling qualities of the rotorcraft simulator.

A simulator must be assessed in the areas critical to the accomplishment of the assigned mission task. These areas typically include longitudinal and lateral-directional responses, performance in takeoff, climb, cruise, descent, etc. Objective tests are used to quantitatively compare simulator and aircraft data to assure that they agree within some specified tolerance. ADS33 specifies an absolute standard for rotorcraft stability behavior. Requirements for handling qualities standards are quantitatively specified frequently in terms of frequency responses. Subsequently, characteristics of frequency response, such as bandwidth, damping ratios, overshoot, and time to peak become the tools of quantitative evaluation criteria. The methodology of ADS33 is applicable to simulation as well, except now these quantitative tools permit definition of tolerances between flight test and simulation data.

Performance

Historically, simulator performance has been evaluated in terms of the simulator's original design specification. This specification normally requires the simulator designer to meet the aircraft flight test data within specified tolerances. Paralleling the FAA's approach, performance testing will include the following flight regimes; hover, vertical and forward flight climb, level flight and autorotational descent. The method of performance testing will consist of classical test techniques as outlined in USNTPS-FTM-106 [6]. Tolerances between actual and simulated flight data are then established for each phase of flight based on simulator category. The tolerance for a category A simulator is thus the most restrictive while the tolerance for a category E simulator is the most relaxed. Figure 2 illustrates the relation between the level of tolerance and the simulator category. The level of tolerance, represented by the expanding circles, reflects an increasing +/- tolerance range with decreasing simulator fidelity classification.

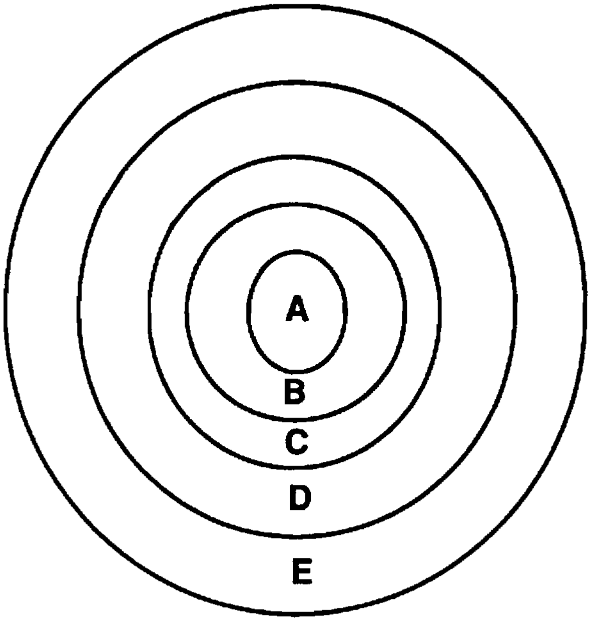


Figure 2. Tolerance Level and Simulator Category

Handling Qualities

While classical performance testing techniques were adequate for evaluation purposes, classic handling qualities testing techniques do not provide adequate information for assessing comparative simulator response. For the past eight years the U.S. Army, with participation from the other military services, the FAA and industry, has been developing a new approach to specifying flight handling qualities for rotorcraft. The existing military specification, MIL-H-8501A, was first published in the early 1950's and had one revision in the early 1960's. The new specification will eventually be designated as a military standard, however, for application to the U.S. Army LH Procurement, the designation ADS33 has been issued. The approach in this new specification is based on defining mission task elements (MTE's) and relating the visual cue environment (VCE) experienced in the aircraft to the level of stabilization required. Although the approach is currently being applied to qualifying rotorcraft, it will have substantial applicability to rotorcraft simulators. ADS33 provides clear quantitative requirements for classifying rotorcraft in terms of its handling qualities. A designation of levels (I, II, III) is utilized. These requirements are divided into three main categories; control system characteristics, hover and low speed, and forward flight. Applying this same standard to simulation, these categories now define evaluation criteria for simulation devices. Subsequently, a set of tolerance levels between flight and simulation data must be established for each simulator category as described in Figure 2. A set of flight test maneuvers based on mission task elements, are simulated to obtain quantitative and qualitative data. This quantitative data is then analyzed and a comparison with actual flight test data is conducted. The deviation between actual and simulated flight data then becomes the measure of acceptability. The proximity to the specified tolerance then validates the simulation device classification.

Pilot Acceptance Criteria

Pilot acceptance is a subjective evaluation. Subjective tests are designed to provide a basis for evaluating simulator capability to perform over a typical training period and to verify correct operation of the

simulator instruments and systems. With respect to ADS33, the flight maneuvers outlined in the previous paragraph serve as the vehicle for a subjective, qualitative evaluation. Based on mission task elements and the visual cue environment, this set of flight maneuvers allows the pilot to assess the perceived performance and handling quality characteristics of the simulator. These are then compared to the pilot's assessment of identical maneuvers in the aircraft. This set of flight maneuvers allows the pilot to explore the perceptual fidelity of the system facilitating a fair assessment. A Cooper Harper rating scale system is used for the evaluation.

METHODOLOGY EVALUATION RESEARCH PLAN

While the methodology discussed in the previous paragraphs provides a framework, it must be evaluated to determine the viability and usefulness. The methodology developed can be considered as a modified FAA and ADS33 approach although much of the structure has been based on work accomplished by the Advisory Group for Aerospace Research and Development (AGARD) [3]. The planned approach for establishing the math model fidelity characteristics is illustrated in Figure 3.

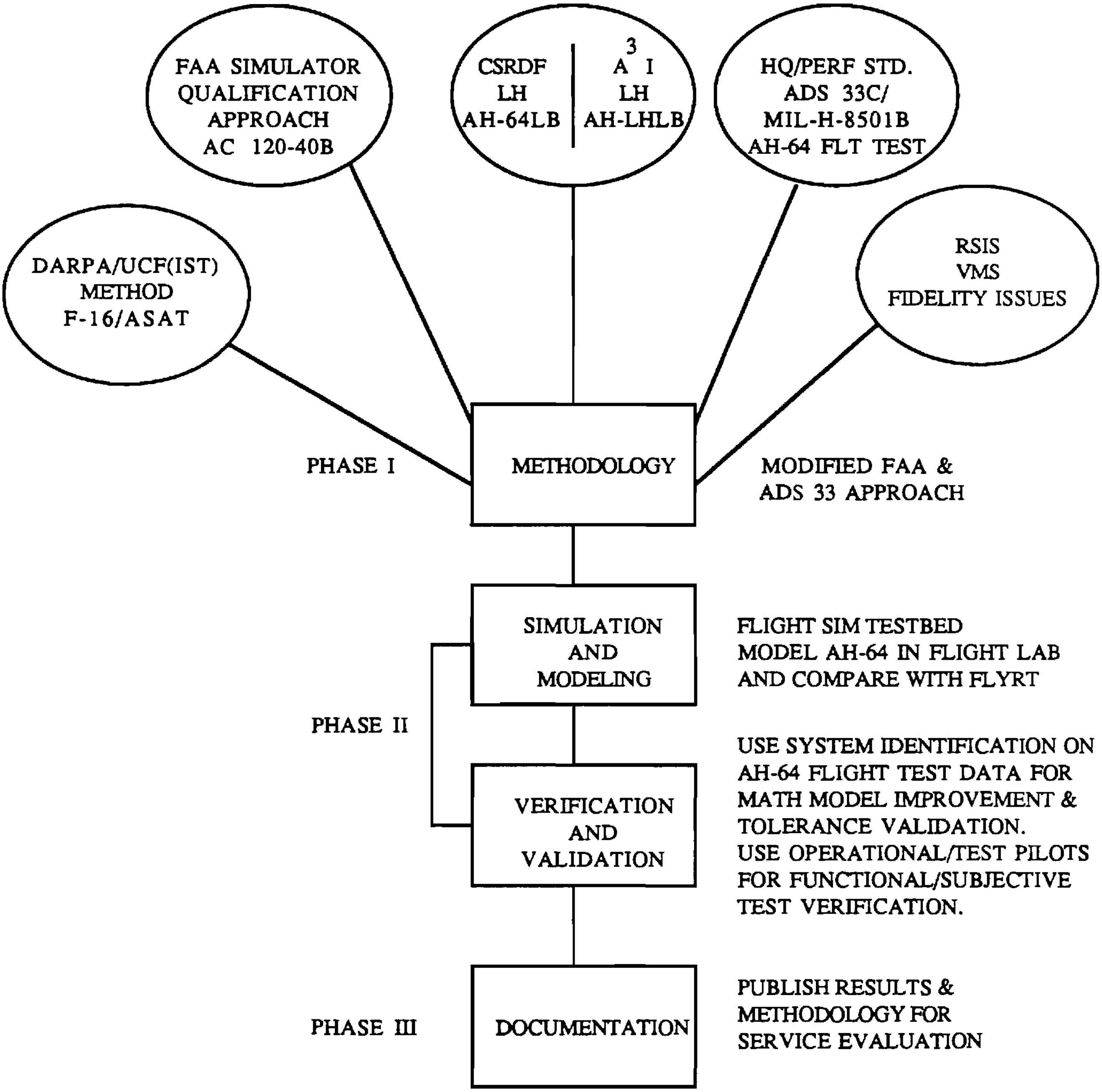


Figure 3. Research plan for establishing math model fidelity characteristics.

As illustrated in Figure 3, Phase II consists of a simulation and modeling effort as well as verification and validation. Under funding through the PM-TRADE/Army Research Institute (ARI), Georgia Tech has developed a real time rotorcraft flight simulation (FLIGHT SIM) laboratory. FLIGHT SIM combines the capabilities of CERWAT and the

NASA loaned Rotorcraft Digital Advanced Avionics System (RODAAS) with the real time parallel processing capabilities of FLIGHT LAB from the Advanced Rotorcraft Technology, Inc. (ART) to establish a university unique research and training evaluation facility. It is planned to use the Georgia Tech FLIGHT SIM as one of the testbeds for evaluating the methodology. The AH-64 Apache is being modeled in FLIGHT LAB and is being compared with the McDonnell Douglas Helicopter Company (MDHC) FLYRT simulation program for initial correlation. Plans are to use system identification methods on the AH-64 flight test data for math model improvement and parameter tolerance validation. Towards this goal, the proposed system identification technique is a two step method, sometimes referred to as Estimation Before Modeling (EBM). In the first step, the helicopter states and net airloads are reconstructed from flight data by a state estimation program. The second step involves modeling the helicopter dynamics so that the model accurately describes and predicts the measured airloads. Appendix B documents the parameter estimation computer program developed under the Phase I effort. U.S. Army operational test pilots who are at Georgia Tech CERWAT for advanced degrees will be used for functional/subjective test verification. Phase III of the planned program would provide published results and produce a methodology for service evaluation on different rotorcraft simulators. A detailed description of the Phase II effort is included as Appendix C. Appendix D details the plans for networking with other simulator sites.

CONCLUSIONS

This methodology offers the rotorcraft simulation community a unique tool for analyzing and tailoring simulation devices for specific requirements. Tieing fidelity directly to the simulation task, linkage is achieved through the simulator classification model. Concurrently, methods for evaluating quantitatively and qualitatively the performance and handling qualities of a rotorcraft simulation device are presented. These methods are consistent with current evaluation criteria. Additionally, this approach permits melding of the FAA certification methodology with the emerging rotorcraft handling qualities specification, ADS-33.

RECOMMENDATIONS

Based on this methodology, several aspects need to be addressed which will require further study. Some of the specific areas for further research include:

1. Quantitative method of defining the fidelity constants.

2. Determination of the task dependent weighting vectors for specified tasks based on simulator type.

3. Identifying the type and magnitude of the couplings for the various fidelity characteristics.

4. Determination of criteria for categorizing simulation types.

5. Validation of methodology using existing simulators.

ACKNOWLEDGEMENTS

This study was conducted under a sub-contract from the Institute for Simulation and Training, University of Central Florida with Brian Goldiez as the technical monitor. Also, valuable discussions with Dennis Folds of Georgia Tech Research Institute are gratefully acknowledged. Considerable cooperation was also received from the U.S. Army Aeroflightdynamics Directorate, particularly Terry Gossett and Doug Haller.

REFERENCES

1. Barber, J.J., "MDHC's Dual-Approach Philosophy to Simulation," Rotor & Wing International, Vol. 22, No. 3, March 1988, pp. 58-60.
2. Goldiez, B. and Uliano, K., "Aviation Technology Research Plan - Talking Paper," Institute for Simulation and Training, April 17, 1990.
3. "Fidelity of Simulation for Pilot Training," AGARD -AR-159.
4. Hoh, R.H. and et al., "Proposed Specification for Handling Qualities of Military Rotorcraft, Volume II - Background Information and User's Guide (BIUG)," USAAVSCOM TR 87-A-3, May 1988.
5. "Proposed Airplane Simulator Qualification," Federal Aviation Administration, AC No. 120-40B.
6. Higgins, L.B. and Hanks, M.L.,"Rotary Wing Performance," USNTPS-FTM-No. 106, 1987.

APPENDIX A

ANNOTATED BIBLIOGRAPHY

Hays, R. T. and Singer, M. J. Simulation Fidelity in Training System Design. New York, NY., Springer-Verlag, 1989

Presents an introduction to the concept of fidelity in training system design. Reviews selected training system research, summarizes the recent research which has examined the fidelity question directly, and provides a review of some of the current trends and future directions of fidelity in training system development.

Airplane Simulator Qualification. Federal Aviation Administration Advisory Circular (AC) no. 120-40B, date unknown.

An AC that provides a means of compliance with the Federal Aviation Regulations (FAR) regarding the evaluation and qualification of airplane simulators to be used in training programs or for airmen checking.

Rotorcraft Simulator, Visual and Motion System Evaluation. Federal Aviation Administration Advisory Circular (AC) no. 120-XX (DRAFT).

An AC that sets forth a means acceptable to the for the evaluation of rotorcraft simulators to be used in training programs or for certification of airmen under Title 14 Code of Federal Regulations (CFR).

Fidelity of Simulation For Pilot Training. Advisory Group for Aerospace Research and Development (AGARD) report no. 159 (AGARD-AR-159), London, United Kingdom, 1980.

Reviews the scope and effectiveness of current flight training in simulators, reviews the status of technologies and human behavior important to the fidelity of flight simulation, and identifies research objectives in the areas of simulation technologies and training that might lead to increased cost-effectiveness in simulator training.

Feasibility Study: Application of SIMNET Technology to the Weapon System Acquisition Process. United States Army Aviation

Research and Technology Activity (ARTA), NASA Ames Research Center, Moffett, CA, 1989.

Provides ARTA evaluation and recommendations on the feasibility of using SIMNET to assist in making DARPA prototyping decisions and OSD acquisition decisions. Issues reviewed include necessary characteristics of simulation systems, how to achieve fidelity, and the feasibility of modifying SIMNET.

Flight Simulation. Advisory Group for Aerospace Research and Development (AGARD) conference proceedings no. 408 (AGARD-CP-408), Cambridge, United Kingdom, 1985.

Provides an up-to-date description of state of the art technology and engineering for both ground-based and in-flight simulators, together with an indication of future possibilities. Places the roles of ground-based and in-flight simulators into context with one another and within the aerospace scene.

Anderson, G.R. A Method for Aircraft Simulation Verification and Validation Developed at the USAF Flight Simulation Facility (AFSF). USAF Flight Test Center, Edwards, CA, 1986.

Reviews the AFSF modular approach to simulation software development, subsequent verification through static and dynamic checks, and various validation methods.

Bray, R. S. Visual and Motion Cueing in Helicopter Simulation. Chapter 1, AGARD-CP-408, Cambridge, United Kingdom, 1986.

Compares the visual cues presented in simulation with those in actual flight in an attempt to identify deficiencies that contribute significantly to poor pilot acceptance. Reviews experimental results in an attempt to determine the effects of motion distortion on pilot performance of height-control tasks.

Ashkenas, I. L. Collected Flight and Simulation Comparisons and Considerations. Chapter 26, AGARD-CP-408, Cambridge, United Kingdom, 1986.

Illustrates diagnostic methods and tools useful in discovering and delineating significant qualitative and quantitative differences between simulation and flight. Reviews examples of both fixed and moving simulation successes and shortcomings.

UH60 Flight Simulator Evaluation. Aviation Engineering Flight Activity (AEFA) Project no. 86-23. Edwards Air Force Base, CA, 1989.

Final Report on tests conducted to evaluate the flight fidelity of the UH-60 Synthetic Flight Training System (SFTS) (Device 2B38). Lists deficiencies, major shortcomings, and enhancing characteristics. Compares simulation data with actual flight data using tolerances defined in the system design specification.

UH60 Flight Simulator Re-evaluation. Aviation Engineering Flight Activity (AEFA) Project no. 86-23-1. Edwards Air Force Base, CA, 1990.

Addendum report to project no. 86-23. A limited re-evaluation of the 2B38 simulator with modified software (CM-145) is reported. Fidelity is assessed primarily in those areas affected by the modifications.

Haller, D. K. AMC Aviation Simulation Plan; Vol. I: Management Plan. U.S. Army Aviation Research and Technology Activity (USAARTA), Aviation Systems Command (AVSCOM), U.S. Army Materials Command (AMC), 1988.

The first of three volumes describing the AMC plan for its overall activities involving development and applications of real-time man-in-the-loop aviation simulators.

Light Helicopter (LH) Program Demonstration and Validation (DEM/VAL) Handling Qualities Simulation Assessment. AEFA Project no. 90-03. Edwards Air Force Base, CA, 1990.

Test plan for an independent handling qualities assessment of the LH designs proposed by the two contractor teams, McDonnell Helicopter/Bell Helicopter Textron and Boeing Helicopters/Sikorsky Aircraft. Assessment plan includes; hands-on flight simulations, review of proposed designs, review of contractor's simulation plans and handling qualities simulation results, and examination and verification of simulator engineering models.

Hoh, Roger H. Advances in Flying Qualities: Concepts and Criteria for a Mission Oriented Flying Qualities Specification. Systems Technology Inc., Hawthorne, CA, date unknown.

Overview of efforts to upgrade flying qualities specifications to more directly reflect the requirements of the intended missions into the specifications. Discusses the use of time vs frequency domain criteria and presents an empirical method to combine the Cooper-Harper Ratings from each axis of control into an overall rating.

Padfield, G. D. The Application of System Identification Techniques to Rotorcraft Simulation Model Validation. Complete draft of chapter 8a, final report of AGARD WG-18 (Helicopter System Identification), London, United Kingdom, 1990.

Demonstrates the utility of system identification to validation of simulation models. Defines validation criteria in terms of three flight dynamic issues; trim, stability, and response.

Aronson, Moses. Validating Visual Cues in Flight Simulator Visual Displays. Proceedings of SPIE - The International Society for Optical Engineering, Bellingham, WA, 1987.

For the night carrier landing task, a comparison of pilot performance in an A-7 prototype part task trainer and actual flight is conducted. Results show that performances were similar, with differences between the flight simulator's flight characteristics and the aircraft have less of an effect than the pilots individual performances.

Barrette, R. E. Flight Simulator Visual Systems - An Overview. Proceedings - Society of Automotive Engineers, Warrendale, PA 1987.

A discussion of the various types of flight simulator visual systems in common use is presented. Limitations and capabilities are discussed, along with their impact on the simulation capabilities.

Norman, D. and Wooldridge, L. Simulator Features: The Neglected Aspect of Instructional Systems Development (ISD). Simulation Series, San Diego, CA, 1986.

Unfortunately, a simulator needed in the hands-on phase may be identified by the ISD process but without specifically defining its

characteristics. This report focuses on a conceptual basis for the specification and justification of simulator features.

Baum, D. R. Training Effectiveness as a Function of Training Device Fidelity. U. S. Army Research Institute Technical Report 593. Alexandria, VA, 1982.

Training effectiveness is defined and related to simulation device fidelity. Current research efforts are discussed and possible future applications explored.

ADDITIONAL REFERENCES

Crosby, J. Evaluation of a Low Fidelity Simulator for Instrument Training. Technical Report AFHRL-TR-78-22, Air Force Human Resources Lab, 1978.

Cox, J. A. Functional and Appearance Fidelity of Training Devices for Fixed Procedures Tasks. HumRRO Technical Report 65-4, Human Resources Research Organization, Alexandria, VA, 1965.

Grimsley, D. L. Acquisition, Retention, and Retraining: The Effects of High and Low Fidelity in Training Devices. HumRRO Technical `Report 69-1, Human Resources Research Organization, Alexandria, VA, 1969.

Grunwald, W. An Investigation of the Effectiveness of Training Devices with Varying Degrees of Fidelity. Unpublished doctoral dissertation, University of Oklahoma (microfilm no. 68-13, 559), 1968.

Hays, R. T. Simulator Fidelity: A Concept Paper. ARI Technical Report 490, U.S. Army Research Institute, Alexandria, VA, 1980.

APPENDIX B

Parameter Estimation Program Documentation

Charles Gardner
Georgia Institute of Technology

List of Symbols

e	residual
i	time index
j	parameter index
k	parameter index
N	number of parameters
M	number of measurements
NP	number of time points in data record
n1	number of positive residuals
n2	number of negative residuals
z	measurement
Z	approximate standardized normal distribution for residual runs
J	cost
$\hat{y}$	model response
GG^T	measurement noise covariance matrix
U	residual run distribution and matrix from SVD
X	parameter sensitivity coefficient
S	scaling matrix
V	matrix from SVD
W	singular value matrix
Λ	parameter error covariance matrix
CRB	Cramer-Rao bound
r	correlation coefficient
I	insensitivity
η	control
μ	mean
σ^2	variance
ξ	parameter vector
^	denotes an estimate

Introduction

One of the efforts under the subcontract from the University of Central Florida CERWAT Simulator contract entails applying system identification to global to global nonlinear flight simulation models. This work is motivated by the need to tune helicopter simulation models in order to closely represent the behavior of the actual helicopter. System identification offers techniques for tuning models to match flight data, and this study will attempt to establish whether these techniques can be effectively applied to nonlinear helicopter simulation models.

The system identification approach being used is a two step method, sometimes refered to as Estimation Before Modeling. In the first step, the helicopter states and net airloads are reconstructed from flight data by a state estimation program such as SMACK. The second step involves modeling the helicopter dynamics so that the model accurately describes and predicts the measured airloads. Developing a good model requires that a suitable model form be selected and that accurate model parameter estimates be obtained.

This report documents the FORTRAN subroutine which has been written to help perform the second, or modeling, step of the system identification method. The routine implements an output-error method, and is intended to be applied to nonlinear models (a nonlinear model is one which is nonlinear in the parameters). Linear models (linear in the parameters) can be handled as well, although regression analysis provides techniques which, in some respects, are more powerful. The Gauss-Newton algorithm (also called quasilinearization or modified Newton-Raphson) is used for minimization of a squared error cost function. Several features are provided in the subroutine to help assess the suitability of the model form and the accuracy of the estimates.

Background: Maximum Likelihood Estimators

It is assumed that the outcome of an experiment depends on some unknown parameters. A maximum likelihood estimator for the parameters is defined as one which maximizes the conditional probability of the outcome given the parameters, or

$$\hat{\xi} = \max_{\xi} P(z|\xi) \qquad (1)$$

Under the assumptions of uncorrelated and Gaussian process and measurement noise, maximum likelihood estimates have several important properties:

1) ML estimates are consistent:

$$\lim_{t \to \infty} P\{|\hat{\xi}(t)-\xi| \leq \varepsilon\} = 1 \qquad (2)$$

where ε is arbitrarily small.

2) ML estimates are asymptotically unbiased:

$$\lim_{t \to \infty} E\{\hat{\xi}(t)\} = \xi \qquad (3)$$

3) ML estimates are asymptotically efficient, which means the variance of the estimates asymptotically attains the Cramer-Rao lower bound.

Obviously, a maximum likelihood estimator has many desirable properties. Under certain circumstances, the ML estimator simplifies to an output error method. The output error method contains many good features, such as being easy to implement on a computer and providing information on the accuracy of the parameter estimates. It is customary in many identification problems to assume that these certain circumstances exist and that therefore an output error method is justified. In reality, this assumption is never strictly valid and the output error parameter estimates will not have the good properties of ML estimates. However, if these special circumstances are not violated too much, an output error method still can provide useful parameter estimates.

Output Error Method

The assumptions under which the maximum likelihood estimator reduces to an output error method are that no process noise (e.g., turbulence) exists and that the measurement noise consists of normally distributed sequences of independent random variables with zero mean and unity variance. This means that when these assumptions hold, an output error method provides consistent, asymptotically unbiased and efficient parameter estimates. Even when the assumptions are slightly violated, the output error method is often used because of its simplicity, with the understanding that the parameter estimates are not ML estimates. Thus, the area of aircraft model identification has in the past extensively employed the output error method to estimate model coefficients.

Case 1: Measurement Noise Covariance Matrix Known If the measurement noise covariance matrix is known, then the output error parameter estimates are those which minimize the sum of squared error cost function

$$J = \frac{1}{2} \sum_{i=1}^{NP} [z_i-\hat{y}_i]^T [GG^T]^{-1} [z_i-\hat{y}_i] \tag{4}$$

The term $[z_i-\hat{y}_i]$ is called the output error, thus motivating the name of the method. It is clear from this performance index that the minimizing parameter values give a system model which best approximates (in a least squares sense) the actual system response to the test input.

Case 2: Measurement Noise Covariance Matrix Unknown In aircraft model identification, GG^T is seldom known. However, it can be estimated by treating its elements as additional parameters. The cost function when GG^T is unknown is

$$J = \frac{1}{2} \sum_{i=1}^{NP} [z_i-\hat{y}_i]^T [GG^T]^{-1} [z_i-\hat{y}_i] + \ln|GG^T| \tag{5}$$

Even when GG^T is known and fixed, equation 5 can be used instead of equation 4 since the last term in 5 will simply be a constant and will not affect the parameter estimates. The measurement noise covariance matrix GG^T is estimated from the sample covariance of the residuals

$$GG^T = \frac{1}{NP} \sum_{i=1}^{NP} [z_i-\hat{y}_i][z_i-\hat{y}_i]^T \tag{6}$$

Equation 6 is a maximum likelihood estimator for GG^T when all the model parameters are known and only GG^T is being estimated.

The output error algorithm for the case when GG^T is unknown involves successively updating the model parameters for fixed GG^T, and then updating GG^T based on the new model parameters. This procedure performs well and has become standard in aircraft identification, where the noise characteristics usually need to be estimated.

In practice, GG^T is restricted to be diagonal. This simplification is justified, as the problem formulation assumes independent measurement noise sequences, which implies zero correlation between different measurement sequences. Another justification is that it makes little sense to penalize products of different measurement sequences, which would be the result of including off-diagonal terms in GG^T.

Gauss-Newton Method

The Gauss-Newton algorithm is one way to minimize the cost function of equation 5. By taking advantage of the special form of the cost function, this algorithm achieves good performance with low computational effort and has emerged as a popular nonlinear parameter estimation tool. The Gauss-Newton method is a quasi-Newton method (a Newton-Raphson method which uses an approximation to the Hessian). Its utility arises from the fact that the Hessian need not be known analytically nor calculated numerically.

The Newton-Raphson parameter update is

$$\Delta\xi = -\left(\nabla_\xi^2 J\right)^{-1} \nabla_\xi^T J \tag{7}$$

The gradient and Hessian of the cost function (5) are

$$\nabla_\xi J = -\sum_{i=1}^{NP} \left[z-\hat{y}\right]_i^T \, [GG^T]^{-1} \, \nabla_\xi \hat{y}_i \tag{8}$$

$$\nabla_\xi^2 J = \sum_{i=1}^{NP} \nabla_\xi \hat{y}_i^T \, [GG^T]^{-1} \, \nabla_\xi \hat{y}_i + \sum_{i=1}^{NP} [z-\hat{y}]_i^T \, [GG^T]^{-1} \, \nabla_\xi^2 \hat{y}_i \tag{9}$$

The Gauss approximation to the Hessian ignores the second term in equation 9:

$$\nabla_\xi^2 J \cong \sum_{i=1}^{NP} \nabla_\xi \hat{y}_i^T \, [GG^T]^{-1} \, \nabla_\xi \hat{y}_i \tag{10}$$

This Hessian approximation involves only the gradient of the model response with respect to the parameters, and avoids requiring any second gradient information. Another benefit of using the Gauss approximation is that if GG^T is positive semidefinite (as it always is), then the Hessian approximation will also be positive semidefinite. This property is desirable because it helps Gauss-Newton avoid the behavior which Newton-Raphson sometimes exhibits of moving in the wrong direction when the Hessian is not positive semidefinite.

Model Structure Determination

Choosing a model structure entails determining which terms are necessary in the model to adequately describe the measured data and to adequately predict other responses of the system. In linear estimation problems, stepwise regression offers a systematic and relatively straightforward way to determine a good model structure. In this method, candidate terms are added one at a time to the model based on their statistical importance in describing the measurements. The model is thus built up from the most important candidate terms, and the unimportant candidate terms are neglected. In nonlinear estimation problems, there is no analogous procedure, and choosing a suitable model form remains a difficult task. In general, picking a nonlinear model structure relies mainly on engineering judgement and trial and error. Fortunately, most helicopter simulation models are primarily analytic representations of the helicopter instead of purely empirical representations. We therefore already have a good idea of the form of a suitable model.

Overparameterization of the Model One of the most common errors in system identification is using a model structure which is too complex. The use of models which are too detailed may in part be motivated by the fact that the model's fit to the data always improves as model complexity increases. However, the model's predictive capability does not always increase with model complexity. In fact, the model's predictive capability decreases when the model becomes too complex. Obviously, then, using only the value of the cost function as a measure of model goodness is a mistake, as a low cost function value does not imply good predictive capability.

A model whose predictive capability has been degraded as a result of the model's complexity is said to be overparameterized. Overparameterized models will frequently exhibit parameter identifiability problems. The best way to avoid overparameterization is to investigate simple models first, while considering the model's ability the match the data, the model's predictive capability, and the accuracy of the parameter estimates. Only those parameters which can be accurately estimated from the data should enter the model.

Physical Basis for the Model Sometimes a model structure will be unable to describe the data not because it is too complex or too simple, but because the model does not represent the essential characteristics of the data. For example, if temperature data from a cooling experiment roughly falling along an exponential curve is modeled using a quadratic model, the fit may be poor for all parameter values. This example illustrates the need for the model structure to reflect the properties of the physical system. Even if a high order polynomial model is used to fit the exponential data very closely, the essence of the cooling process will not be captured and the resulting model may be a poor predictor.

Analysis of Residuals One technique to determine if the model captures all the phenomena present in the data is to examine the residual sequence, which is the difference between the measurement and the model predicted response:

$$e_i = z_i - \hat{y}_i \tag{11}$$

If the model accurately fits the data, the residuals will be the measurement noise and will therefore form an uncorrelated sequence. If the model does not fit the data well, the residuals will be a correlated sequence. Hence, one way to determine whether the model accurately describes the data is to check the residuals. If the residual sequence looks like white noise, then the model has explained all the nonrandom information in the data. But if the residual sequence forms a pattern, then probably this pattern represents behavior the model has not captured.

A simple test for correlated residuals is based on the analysis of runs and the number of positive and negative residuals. The number of runs in a residual sequence is the number of residual sign changes plus one. Given the number of positive and the number of negative residuals (n_1 and n_2), the mean of the discrete distribution of U (expected number of runs) is

$$\mu = \frac{2n_1n_2}{(n_1+n_2)} + 1 \tag{12}$$

and the variance of U is

$$\sigma^2 = \frac{2n_1n_2(2n_1n_2-n_1-n_2)}{(n_1+n_2)^2(n_1+n_2-1)} \tag{13}$$

If n_1 and n_2 are large (greater than about 10) and the number of observations is much greater than the number of parameters (as it almost always is), then the distribution of U is approximately normal with mean μ and variance σ^2:

$$U \sim N(\mu, \sigma^2) \tag{14}$$

If a random variable Z is defined as

$$Z = \frac{(U-\mu+\frac{1}{2})}{\sigma} \tag{15}$$

where the 1/2 is a discrete distribution correction factor, then Z is approximately normally distributed with mean 0 and variance 1:

$$Z \sim N(0,1) \tag{16}$$

Hence Z has the standardized normal distribution. Now the probability of obtaining u or fewer runs is

$$P(U \leq u) = P(Z \leq z) \tag{17}$$

where z is calculated according to equation 15 and $P(Z \leq z)$ can be looked up in a standardized normal distribution table. If the probability of obtaining the observed u or fewer runs is very low (maybe <.05), this indicates that an unusually low number of runs have occured. Thus we conclude that the arrangement of residual signs is probably not random and that the residuals contain additional unmodeled information. If the probablility of obtaining the observed u or fewer runs is not low (maybe >.05), then a sufficient number of runs have occured to conclude that the residual sequence probably contains only noise.

Parameter Identifiability

It is not always true that all parameters in a model can be uniquely estimated from the measured data. Parameters which cannot be uniquely estimated are said to be unidentifiable. Situations which cause parameters to be unidentifiable are listed below.

Linear Dependence Among Sensitivity Coefficients When two or more of the parameters in the model affect the model response in a similar manner, these parameters cannot be uniquely determined, although many times linear combinations of these parameters can be estimated. This situation exists when the parameter sensitivity coefficients are linearly dependent or nearly so over the range of the measurements, where the sensitivity coefficients are defined as

$$X_k = \frac{\partial \hat{y}}{\partial \xi_k} \tag{18}$$

As an example, suppose we try to use the model $y=(m_1+m_2)x+b$ to describe some data which falls in roughly a straight line. The parameters are m_1, m_2, and b. The sensitivity coefficients are $X_1=x$, $X_2=x$, and $X_3=1$. The first two coefficients are linearly dependent since $1*X_1-1*X_2=0$, so m_1 and m_2 cannot be uniquely estimated. However, the linear combination m_1+m_2 can be uniquely determined, and it will equal the slope of the best-fit straight line.

In practice, groups of parameters are often close to being linearly dependent without exactly satisfying the linear dependence criterion. It is important to note that minimization routines will still usually select values for the parameters in a linearly dependent group, but that these parameter values will not be unique. In a model where the parameters represent actual physical quantities, we usually insist on unique parameter estimates. Therefore, when linear dependence problems occur, the model form must be changed so that the parameter estimates are unique and meaningful.

Extraneous Parameters in the Model Parameters which are present in the model but which are not important in determining the model output will be unidentifiable. If the model output is insensitive to these extraneous parameters, then the cost function will also be insensitive to them, implying that the parameters may assume any value without affecting the cost. Since the minimization routine cannot determine meaningful and accurate estimates for extraneous parameters, these parameters must be removed from the model.

Insufficient Excitation of the Dynamics Parameters which are generally important in describing the behavior of a system may by unidentifiable simply because the test data contains insufficient information. This situation arises when some dynamic modes are not excited and are therefore not present in the data. The parameters which describe the unexcited modes then are unidentifiable from the data. For example, the pitch damping stability derivative, while very important in determining the short period motion of an airplane, may be unidentifiable from data taken during a phugoid motion.

Generally, the best test inputs excite the aircraft over a wide range of frequencies to ensure that all the important dynamic modes are present in the response. The 3211 input developed at DLR is one of the best generic inputs for time domain aircraft identification; the doublet may be acceptable, while the step and pulse are usually unacceptable.

Symptoms of Identifiability Problems

For complex nonlinear models, it may be impossible to know before attempting minimization if there are identifiability problems. However, during the minimization some information becomes available which can indicate identifiability problems. Listed below are some of the symptoms of unidentifiable parameters.

1) Slow convergence during minimization.

2) Large Cramer-Rao bounds on the parameter estimates.

3) Large insensitivities on the parameter estimates.

4) Near-singular Hessian matrix.

5) High conditional or unconditional correlation coefficients.

Solutions to Parameter Identifiability Problems

Model Structure Simplification One obvious solution to identifiability problems is to remove the unidentifiable parameters from the model. Reducing the number of parameters in this way can produce a model with better predictive capabilities and smaller parameter error variances.

Parameter Fixing Sometimes a parameter is unidentifiable from some test data even though the parameter is generally important in determining the system response. In this case, it may be desirable to keep the unidentifiable parameter in the model but to not estimate its value. This technique is called parameter fixing. Parameter fixing is useful if the test response is insensitive to the parameter and if an a priori parameter value is available. However, in general the value of the fixed parameter affects the other parameter estimates, so parameter fixing should be done cautiously.

Rank-Deficient Parameter Updates The Newton-Raphson parameter update is the solution for $\Delta\xi$ to the equation

$$\left(\nabla_{\xi}^{2}J\right)\Delta\xi = -\nabla_{\xi}^{T}J \tag{19}$$

The Hessian can be viewed as a linear transformation which maps $\Delta\xi$ to the transpose of the negative gradient of J. A unique solution for $\Delta\xi$ exists if and only if the Hessian has full rank. If the Hessian has less than full rank, then equation 19 does not have a unique solution for $\Delta\xi$. Since equation 19 must be solved for $\Delta\xi$ during each iteration of the minimization program, the Hessian must have full rank to yield meaningful parameter updates.

Suppose the Hessian has nullity one. Then the Hessian inverse does not exist and equation 19 cannot be solved for a unique $\Delta\xi$. There will be one basis vector for the null space (since the dimension of the null space is one), and this basis vector will represent the combination of parameters which cannot be identified uniquely. One strategy is then to fix the combination of unidentifiable parameters during the update. This means the dimension of the parameter space is reduced by one; the direction ignored in the update is given by the basis for the nullspace of the Hessian. It is important to understand that all the parameter values are updated, and what is fixed is certain combinations of parameters. This strategy is called the rank-deficient parameter update.

The implementation of the rank-deficient parameter update is straightforward. The columns of the Hessian matrix are first scaled (by the column norm, for example):

$$\nabla_{\xi}^{2} J_{\text{scaled}} = \nabla_{\xi}^{2} J * S \qquad (20)$$

$$S = \begin{bmatrix} \frac{1}{\left\|\nabla_{\xi}^{2} J_{k1}\right\|} & & 0 \\ & \ddots & \\ 0 & & \frac{1}{\left\|\nabla_{\xi}^{2} J_{kN}\right\|} \end{bmatrix} \qquad (21)$$

This scaling is a preconditioning step to ensure that near-singularities are not caused by variable scale effects. Singular value decomposition is then performed on the scaled Hessian:

$$\nabla_{\xi}^{2} J_{\text{scaled}} = UWV^{T} \qquad (22)$$

In equation 22, W is a diagonal matrix containing the singular values of the scaled Hessian. The columns of U with corresponding nonzero singular values form an orthonormal set of basis vectors for the range of the scaled Hessian. The columns of V with corresponding zero singular values form an orthonormal set of basis vectors for the null space. These basis vectors for the null space indicate which combinations of parameters are unidentifiable.

In practice, singular values will seldom be exactly zero. However, the Hessian need only be near singular to indicate identifiability problems. The Hessian condition number (ratio of largest to smallest singular values) tells how close the matrix is to singular. If the condition number is greater than some specified number (maybe 1x10^3 or 1x10^4), we consider the Hessian singular and compute a rank-deficient inverse instead of the full rank inverse.

The full rank inverse of the scaled Hessian is

$$\left(\nabla_{\xi}^{2} J_{\text{scaled}}\right)^{-1} = VW^{-1}U^{T} \tag{23}$$

The rank-deficient inverse is computed in the same way, but with the inverse of the small singular values set to zero in W^{-1}:

$$\frac{1}{w_{kk}} = 0 \tag{24}$$

Now the inverse scaling is applied to the scaled Hessian inverse to obtain the Hessian inverse:

$$\left(\nabla_{\xi}^{2} J\right)^{-1} = S*\left(\nabla_{\xi}^{2} J_{\text{scaled}}\right)^{-1} \tag{25}$$

And finally the parameters are updated according to equation 7.

Of course, the Hessian must still have full rank at the solution for a well-defined minimum to exist, and the rank-deficient parameter update does not change this requirement. So even though the rank-deficient updates provide parameter estimates, these estimates are usually not acceptable if the Hessian is near-singular at the solution simply because the estimates are not unique. The SVD rank-deficient update is useful because it constrains the parameter values from moving in directions which contain little or no information about the parameters. The full rank update has the undesirable property of taking the largest parameter steps in the direction of most parameter uncertainty, while the rank-deficient update avoids this behavior. The rank-deficient update should be viewed as a one tool to help determine the cause of unidentifiable parameters, and not as a solution to identifiability problems.

The singular value decomposition provides information which can indicate the cause of identifiability problems. If the Hessian is near singular, the basis vectors for the null space define combinations of parameters which are unidentifiable. Therefore, these basis vectors can be inspected to determine either which parameters have linearly dependent sensitivity coefficients or which parameters are not important in the describing the data.

Accuracy of the Estimates

Cramer-Rao Lower Bound One of the best measures of an estimate's accuracy is the Cramer-Rao lower bound. The Cramer-Rao bound gives a theoretical lower limit to the standard deviation of the parameter estimate. A low CRB means the parameter can potentially be estimated accurately, whereas a high CRB indicates the parameter cannot be estimated accurately. High Cramer-Rao bounds signal parameter identifiability problems, caused by either an inadequate model structure or inadequate data.

In practice, the Cramer-Rao bound is seldom realized because most estimators are not efficient (nor are they unbiased and consistent). Yet the Cramer-Rao bound still gives important information about the relative accuracy of the parameter estimates. A widely used rule-of-thumb (based on experience) in aircraft identification for estimating the actual standard deviation of the parameter estimates is to multiply the CRB by a factor of five to ten.

Computing an approximation to the CRB is simple in the Gauss-Newton method. The CRB is

$$\mathrm{CRB}(\xi_k) = \sqrt{\Lambda_{kk}} \tag{26}$$

where Λ is the covariance matrix of the final parameter estimates. This covariance matrix is approximated (at the solution) by

$$\Lambda \equiv \mathrm{Cov}(\xi) \equiv \left\{ \sum_{i=1}^{NP} \nabla_\xi \hat{y}_i^{\,T} \; (GG^T)^{-1} \; \nabla_\xi \hat{y}_i \right\}^{-1} \tag{27}$$

The right hand side of equation 27 is the inverse of the Gauss approximation to the Hessian. Therefore, at the solution, the parameter error covariance matrix is approximately equal to the Hessian inverse. This quantity is already calculated in the Gauss-Newton minimization routine, so estimating the Cramer-Rao bounds requires only the computation shown in equation 26.

Other Measures of Accuracy Other measures of parameter accuracy exist, but are usually not as useful as the CRB. One such measure is the parameter error correlation coefficient (both unconditional and conditional). Using equation 27 as the approximation to the parameter error covariance matrix, the correlations are:

$$\text{unconditional } r_{jk} = \frac{\Lambda_{jk}}{\sqrt{\Lambda_{jj}\Lambda_{kk}}} \qquad (28)$$

$$\text{conditional } r_{jk} = \frac{-\Lambda^{-1}_{jk}}{\sqrt{\Lambda^{-1}_{jj}\,\Lambda^{-1}_{kk}}} \qquad (29)$$

where $\Lambda^{-1}{}_{jk}$ denotes the ij element of Λ^{-1}. When either of the correlation coefficients is high (maybe >.90), significant correlation exists between two or three of the parameters (the sensitivity coefficients are nearly linearly dependent). These parameters will then be difficult to estimate accurately. The problem with using the correlation coefficients as measures of parameter accuracy is that they show correlations between, at most, three parameters. When correlations exist between more than three parameters, these correlations cannot be seen using the correlation coefficients.

Another measure of parameter accuracy is the parameter insensitivity, defined as

$$I_k = \frac{1}{\sqrt{\Lambda^{-1}_{kk}}} \qquad (30)$$

The insensitivity is an approximation to the standard deviation of the parameter estimate, ignoring inaccuracies in other parameter estimates and effects of correlation between parameters. Since much of the inaccuracy in the estimates results from precisely the effects the insensitivity ignores, the insensitivity is generally not useful. However, it will point out one important parameter identifiability problem. A large insensitivity indicates that the parameter has little effect on the system response to the test input. Inspecting the insensitivities can then immediately point out parameters which should be excluded from the model.

Detailed Program Description

The subroutine gauss_newton and its supporting subroutines implement the Gauss-Newton algorithm. The user must write the calling program, which reads the data, sets some constants, logicals, initial parameter vector, etc., and passes this information to gauss_newton. The user must also write a subroutine called model which produces the model response given the parameters and control time histories. These control time histories must be passed to model from the main program through a common block.

Main Program The main program is written by the user, and the following variables must be set or initialized in this program:

n	number of parameters in model
m	number of measurement variables
np	number of time points in concatenated measurement sequence
z	array containing flight data (mxnp)
p	initial parameter vector (nx1)
r	measurement noise variance vector (mx1). This vector need not be set if it is being automatically estimated by the program.
chgparam	maximum allowable relative change in parameters for convergence.
cond	matrix condition number for determining which singular values are small.
itermax	maximum allowable number of iterations
imnoise	logical = true if measurement noise covariance matrix is being updated automatically = false otherwise
rank	logical = true for rank deficient updates = false for full rank updates
outfile	name of output file
cont	array containing control time histories (5xnp)

If the measurements consist of several maneuvers, then the measurement time histories are concatenated in the matrix z. That is, each row of z will contain the time history of one measurement variable for all maneuvers.

The beginning of the main program must appear as:

```
      program main
      implicit double precision (a-h,o-z)
      common/control/cont
      parameter (ndim=30,mdim=10,npdim=500)
      logical imnoise,rank
      character*16 outfile
      dimension z(mdim,npdim),r(mdim),p(ndim),cont(5,npdim)
```

The dimensioning parameters may be changed, but only if they are also changed in all the subroutines.

Once the values listed above have been set, subroutine gauss_newton is called with the statement:

```
      call gauss_newton(n,m,np,z,p,r,chgparam,cond,itermax,
                        imnoise,rank,outfile)
```

Upon return from this subroutine, the program should stop. The user does not need to print any output, as subroutine gauss_newton writes to the output file during each iteration and upon convergence.

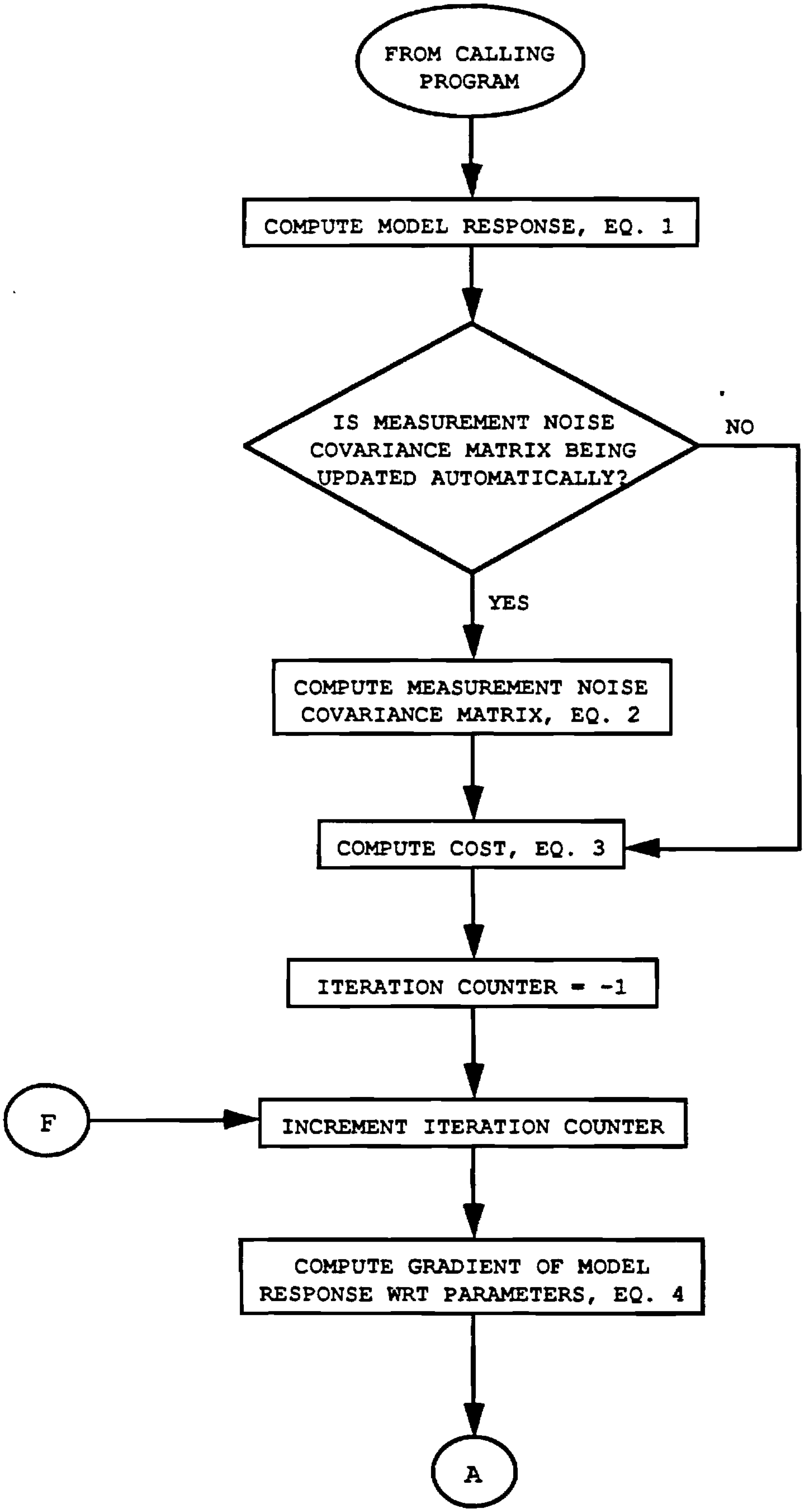
Subroutine Model This user-written subroutine generates the model response given the parameter vector and the control time histories. The first lines of the subroutine should be:

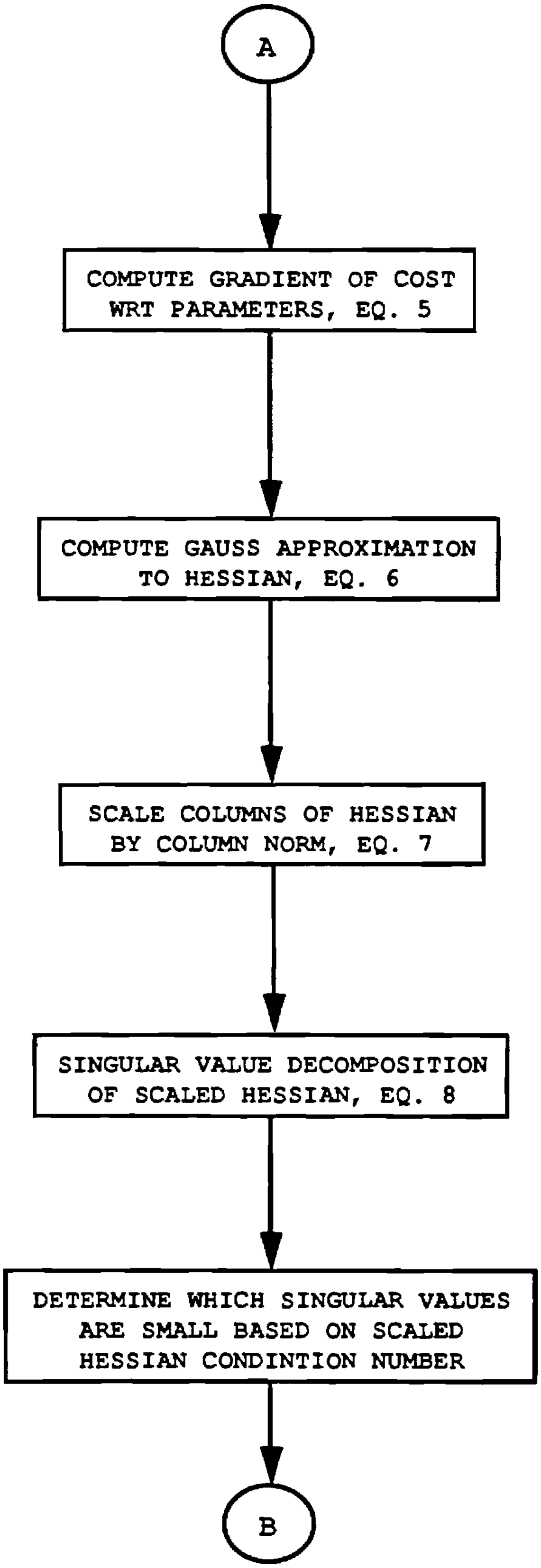
```
      subroutine model(p,yhat)
      implicit double precision (a-h,o-z)
      common/control/cont
      parameter(ndim=30,mdim=10,npdim=500)
      dimension p(ndim),yhat(mdim,npdim),cont(5,npdim)
```

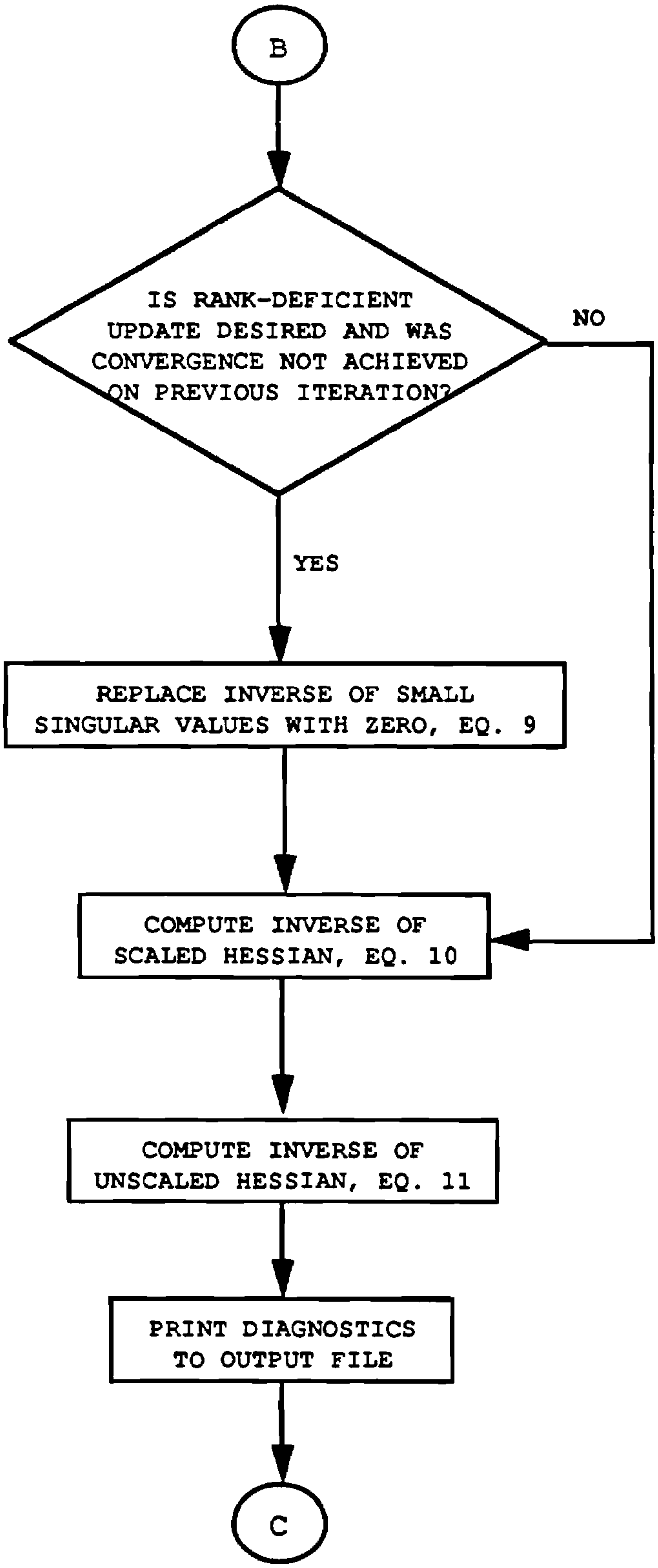
The model response must be loaded into the yhat array exactly as the measurement data appears in the z array. If several maneuvers are being used, yhat contains the concatenated response time histories.

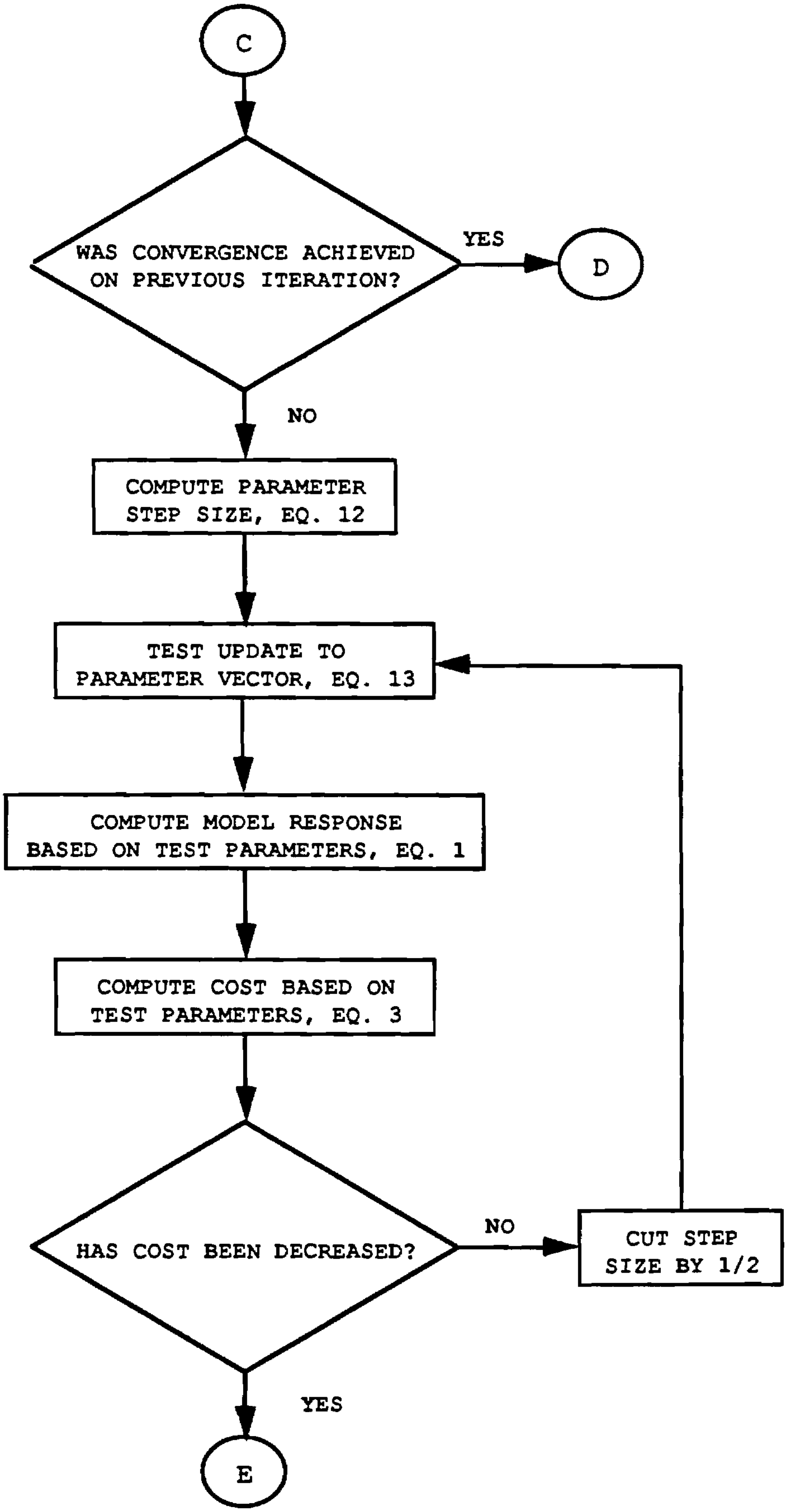
Subroutine Gauss_Newton This routine estimates the model parameters according to the Gauss-Newton algorithm. The following flow chart and list of equations describe the subroutine's operation.

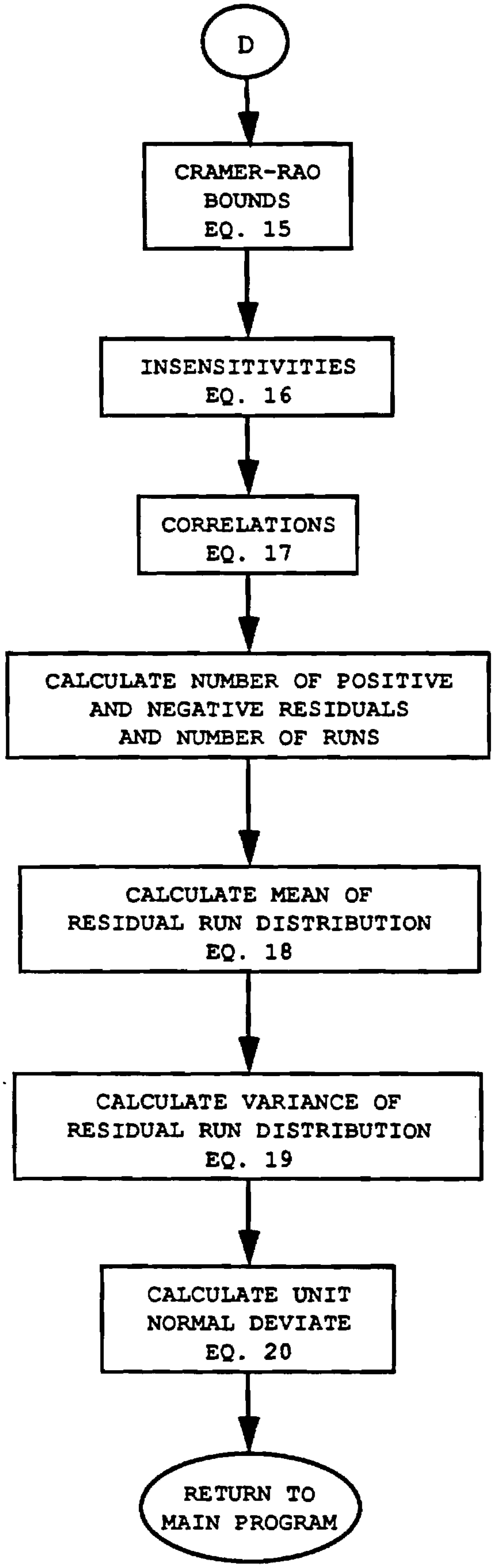
Subroutine Gauss-Newton Flow Chart

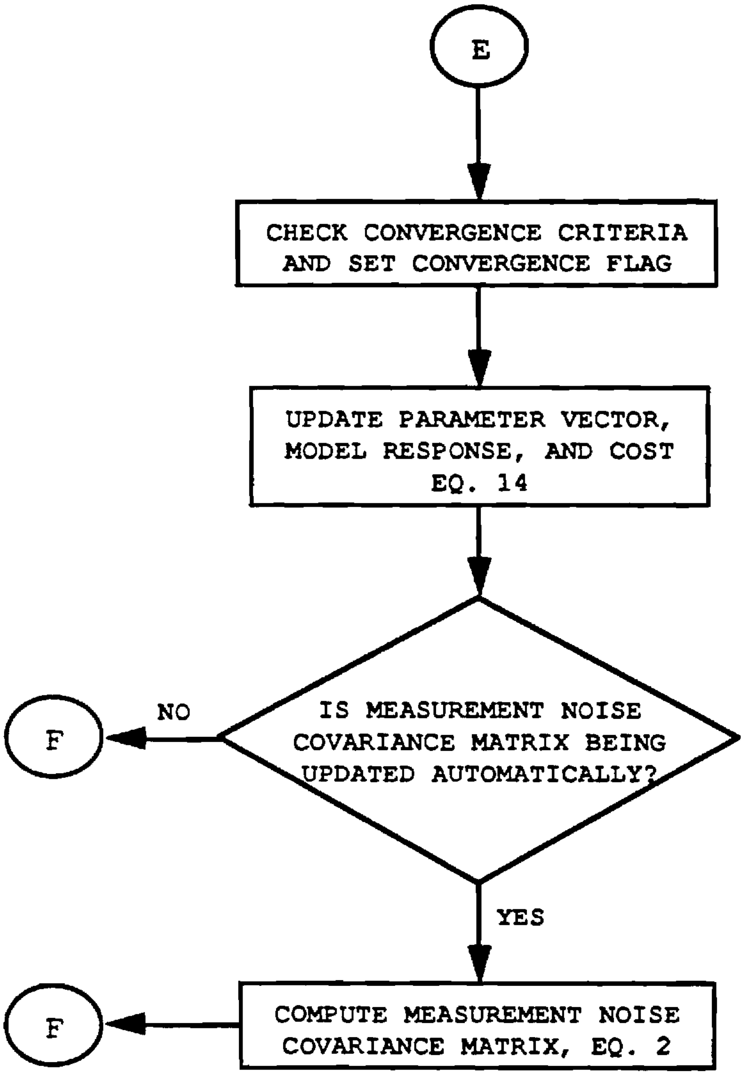

FROM CALLING PROGRAM
COMPUTE MODEL RESPONSE, EQ. 1
IS MEASUREMENT NOISE COVARIANCE MATRIX BEING UPDATED AUTOMATICALLY?
NO
YES
COMPUTE MEASUREMENT NOISE COVARIANCE MATRIX, EQ. 2
COMPUTE COST, EQ. 3
ITERATION COUNTER = -1
F
INCREMENT ITERATION COUNTER
COMPUTE GRADIENT OF MODEL RESPONSE WRT PARAMETERS, EQ. 4
A


A
COMPUTE GRADIENT OF COST WRT PARAMETERS, EQ. 5
COMPUTE GAUSS APPROXIMATION TO HESSIAN, EQ. 6
SCALE COLUMNS OF HESSIAN BY COLUMN NORM, EQ. 7
SINGULAR VALUE DECOMPOSITION OF SCALED HESSIAN, EQ. 8
DETERMINE WHICH SINGULAR VALUES ARE SMALL BASED ON SCALED HESSIAN CONDINTION NUMBER
B


B
IS RANK-DEFICIENT UPDATE DESIRED AND WAS CONVERGENCE NOT ACHIEVED ON PREVIOUS ITERATION?
NO
YES
REPLACE INVERSE OF SMALL SINGULAR VALUES WITH ZERO, EQ. 9
COMPUTE INVERSE OF SCALED HESSIAN, EQ. 10
COMPUTE INVERSE OF UNSCALED HESSIAN, EQ. 11
PRINT DIAGNOSTICS TO OUTPUT FILE
C


C
WAS CONVERGENCE ACHIEVED ON PREVIOUS ITERATION?
YES
D
NO
COMPUTE PARAMETER STEP SIZE, EQ. 12
TEST UPDATE TO PARAMETER VECTOR, EQ. 13
COMPUTE MODEL RESPONSE BASED ON TEST PARAMETERS, EQ. 1
COMPUTE COST BASED ON TEST PARAMETERS, EQ. 3
HAS COST BEEN DECREASED?
NO
CUT STEP SIZE BY 1/2
YES
E


D
CRAMER-RAO BOUNDS EQ. 15
INSENSITIVITIES EQ. 16
CORRELATIONS EQ. 17
CALCULATE NUMBER OF POSITIVE AND NEGATIVE RESIDUALS AND NUMBER OF RUNS
CALCULATE MEAN OF RESIDUAL RUN DISTRIBUTION EQ. 18
CALCULATE VARIANCE OF RESIDUAL RUN DISTRIBUTION EQ. 19
CALCULATE UNIT NORMAL DEVIATE EQ. 20
RETURN TO MAIN PROGRAM


E
CHECK CONVERGENCE CRITERIA AND SET CONVERGENCE FLAG
UPDATE PARAMETER VECTOR, MODEL RESPONSE, AND COST EQ. 14
IS MEASUREMENT NOISE COVARIANCE MATRIX BEING UPDATED AUTOMATICALLY?
NO
F
YES
COMPUTE MEASUREMENT NOISE COVARIANCE MATRIX, EQ. 2
F

Equations Implemented in Subroutine Gauss-Newton

Equation 1 (user supplied):

$$\hat{y} = f(\xi, \eta)$$

Equation 2:

$$GG^{T} = \mathrm{diag}\left[\frac{1}{NP}\sum_{i=1}^{NP}\left[z-\hat{y}\right]_i\left[z-\hat{y}\right]_i^{T}\right]$$

where

$$\left[z-\hat{y}\right]_i = \begin{bmatrix} z_1-\hat{y}_1 \\ \cdot \\ \cdot \\ z_M-\hat{y}_M \end{bmatrix}_i$$

Equation 3:

$$J(\xi) = \frac{1}{2}\sum_{i=1}^{NP}\left\{\left[z-\hat{y}\right]_i^{T}\left(GG^{T}\right)^{-1}\left[z-\hat{y}\right]_i\right\} + \frac{1}{2}NP*\ln\left|GG^{T}\right|$$

Equation 4:

$$\frac{\partial\hat{y}_i}{\partial\xi_k} = \frac{\hat{y}_i(\xi_k+\Delta\xi_k) - \hat{y}_i(\xi_k)}{\Delta\xi_k}$$

so

$$\nabla_\xi \hat{y}_i = \begin{bmatrix} \frac{\partial \hat{y}_1}{\partial \xi_1} & \cdots & \frac{\partial \hat{y}_1}{\partial \xi_N} \\ & \ddots & \\ & & \frac{\partial \hat{y}_M}{\partial \xi_N} \end{bmatrix}_i$$

Equation 5:

$$\nabla_\xi J = -\sum_{i=1}^{NP} \left[z-\hat{y}\right]_i^T \left(GG^T\right)^{-1} \nabla_\xi \hat{y}_i$$

so

$$\nabla_\xi J = \begin{bmatrix} \frac{\partial J}{\partial \xi_1} & \cdots & \frac{\partial J}{\partial \xi_N} \end{bmatrix}$$

Equation 6:

$$\nabla_\xi^2 J \cong \sum_{i=1}^{NP} \nabla_\xi \hat{y}_i^T \left(GG^T\right)^{-1} \nabla_\xi \hat{y}_i$$

so

$$\nabla_\xi^2 J = \begin{bmatrix} \frac{\partial^2 J}{\partial \xi_1^2} & \cdots & \frac{\partial^2 J}{\partial \xi_1 \partial \xi_N} \\ & \ddots & \\ & & \frac{\partial^2 J}{\partial \xi_N^2} \end{bmatrix}$$

Equation 7:

$$\nabla_{\xi}^{2} J_{scaled} = \nabla_{\xi}^{2} J * S$$

where

$$S = \begin{bmatrix} \frac{1}{\left\| \nabla_{\xi}^{2} J_{k1} \right\|} & & 0 \\ & \ddots & \\ 0 & & \frac{1}{\left\| \nabla_{\xi}^{2} J_{kN} \right\|} \end{bmatrix}$$

Equation 8:

$$\nabla_{\xi}^{2} J_{scaled} = UWV^{T}$$

Equation 9:

$$\frac{1}{w_{kk}} = 0$$

Equation 10:

$$\left(\nabla_{\xi}^{2} J_{scaled} \right)^{-1} = VW^{-1}U^{T}$$

Equation 11:

$$\left(\nabla_{\xi}^{2} J \right)^{-1} = S * \left(\nabla_{\xi}^{2} J_{scaled} \right)^{-1}$$

Equation 12:

$$\Delta\xi = -\left(\nabla_{\xi}^{2} J \right)^{-1} \nabla_{\xi}^{T} J$$

Equation 13:

$$\xi_{test} = \xi + \Delta\xi$$

Equation 14:

$$\xi = \xi_{test}$$

$$J = J_{test}$$

$$\hat{y} = \hat{y}_{test}$$

Equation 15:

$$CRB_k = \sqrt{\left(\left(\nabla_\xi^2 J\right)^{-1}\right)_{kk}}$$

Equation 16:

$$I_k = \frac{1}{\sqrt{\left(\nabla_\xi^2 J\right)_{kk}}}$$

Equation 17:

$$\text{unconditional } r_{jk} = \frac{\left(\left(\nabla_\xi^2 J\right)^{-1}\right)_{jk}}{\sqrt{\left(\left(\nabla_\xi^2 J\right)^{-1}\right)_{jj}\left(\left(\nabla_\xi^2 J\right)^{-1}\right)_{kk}}}$$

$$\text{conditional } r_{jk} = \frac{-\left(\nabla_\xi^2 J\right)_{jk}}{\sqrt{\left(\nabla_\xi^2 J\right)_{jj}\left(\nabla_\xi^2 J\right)_{kk}}}$$

Equation 18:

$$\mu = \frac{2n_1n_2}{(n_1+n_2)} + 1$$

Equation 19:

$$\sigma^2 = \frac{2n_1n_2(2n_1n_2-n_1-n_2)}{(n_1+n_2)^2(n_1+n_2-1)}$$

Equation 20:

$$Z = \frac{(U-\mu+\frac{1}{2})}{\sigma}$$

Example Problem This example is taken from Draper and Smith. The problem is to model the level of available chlorine in a product as a function of time. Data consists of measured available chlorine fractions at different times after manufacture. The proposed model is

$$y = \alpha + (.49 - \alpha) e^{-\beta(x-8)}$$

where y is the available chlorine fraction, x is the time in weeks after manufacture, and α and β are the parameters which need to be estimated.

The main program and the subroutine model for this problem are given on the following pages. The independent variable and measurement time histories are read from a data file and stored in the matrices cont and z. The data file is also reproduced below. Lastly, the output file is given. The parameter estimates agree with Draper and Smith to four decimal places. Examination of the estimates' accuracies and the residual runs indicate that the model form and parameter estimates are good.

```
      program test

      implicit double precision (a-h,o-z)
      parameter(ndim=30,mdim=10,npdim=500)
      common/control/cont
      dimension z(mdim,npdim),r(mdim),p(ndim),cont(5,npdim)
      logical rank,imnoise
      character*16 outfile

      n=2
      m=1
      np=44
      imnoise=.false.
      rank=.false.
      cond=1.0d+03
      itermax=20
      chgparam=.01d+00
      r(1)=1.0d+00
      p(1)=3.0d-01
      p(2)=5.0d-02
      outfile='testout.dat'
      open(unit=4,file='infile.dat',status='old')
      do 10 i=1,np
      read(4,*)cont(1,i),z(1,i)
0     continue
      close(unit=4)

      call gauss_newton(n,m,np,z,p,r,chgparam,cond,itermax,
     $                  imnoise,rank,outfile)

      stop
      end

      subroutine model(p,yhat)

      implicit double precision (a-h,o-z)
      common/control/cont
      parameter(ndim=30,mdim=10,npdim=500)
      dimension p(ndim),yhat(mdim,npdim),cont(5,npdim)

      do 10 i=1,44
      yhat(1,i)=p(1)+(.49d+00-p(1))*dexp(-p(2)*(cont(1,i)-8.0d+00))
0     continue

      return
      end
```

8	.49
8	.49
10	.48
10	.47
10	.48
10	.47
12	.46
12	.46
12	.45
12	.43
14	.45
14	.43
14	.43
16	.44
16	.43
16	.43
18	.46
18	.45
20	.42
20	.42
20	.43
22	.41
22	.41
22	.40
24	.42
24	.40
24	.40
26	.41
26	.40
26	.41
28	.41
28	.40
30	.40
30	.40
30	.38
32	.41
32	.40
34	.40
36	.41
36	.38
38	.40
38	.40
40	.39
42	.39

```
**************** ITERATION  0 ****************
COST = 0.20207D-01
NORM OF COST GRADIENT = 0.13869D+01
NUMBER OF SMALL SINGULAR VALUES OF HESSIAN =  0
BASIS VECTORS (ROW VECTORS) FOR RANGE SPACE =
  -0.41432  0.91013
  -0.91013 -0.41432
DIAGONALS OF MEASUREMENT NOISE COVARIANCE MATRIX
  0.1000D+01
PARAMETER VECTOR
  0.30000D+00 0.50000D-01


**************** ITERATION  1 ****************
COST = 0.15417D-01
NORM OF COST GRADIENT = 0.73658D+00
NUMBER OF SMALL SINGULAR VALUES OF HESSIAN =  0
BASIS VECTORS (ROW VECTORS) FOR RANGE SPACE =
  -0.92841  0.37155
  -0.37155 -0.92841
DIAGONALS OF MEASUREMENT NOISE COVARIANCE MATRIX
  0.1000D+01
PARAMETER VECTOR
  0.41805D+00 0.80288D-01


**************** ITERATION  2 ****************
COST = 0.25276D-02
NORM OF COST GRADIENT = 0.20691D-01
NUMBER OF SMALL SINGULAR VALUES OF HESSIAN =  0
BASIS VECTORS (ROW VECTORS) FOR RANGE SPACE =
  -0.94955  0.31361
  -0.31361 -0.94955
DIAGONALS OF MEASUREMENT NOISE COVARIANCE MATRIX
  0.1000D+01
PARAMETER VECTOR
  0.39183D+00 0.10963D+00


**************** ITERATION  3 ****************
COST = 0.25012D-02
NORM OF COST GRADIENT = 0.44114D-02
NUMBER OF SMALL SINGULAR VALUES OF HESSIAN =  0
BASIS VECTORS (ROW VECTORS) FOR RANGE SPACE =
  -0.93520  0.35413
  -0.35413 -0.93520
DIAGONALS OF MEASUREMENT NOISE COVARIANCE MATRIX
  0.1000D+01
PARAMETER VECTOR
  0.39034D+00 0.10166D+00


!!!!!!! CONVERGENCE BY CRITERIA #1 !!!!!!!!!


**************** ITERATION  4 ****************
COST = 0.25008D-02
NORM OF COST GRADIENT = 0.56463D-06
```

```
NUMBER OF SMALL SINGULAR VALUES OF HESSIAN =  0
BASIS VECTORS (ROW VECTORS) FOR RANGE SPACE =
  -0.93491  0.35489
  -0.35489 -0.93491
DIAGONALS OF MEASUREMENT NOISE COVARIANCE MATRIX
  0.1000D+01
PARAMETER VECTOR
  0.39014D+00 0.10163D+00
PARAMETER INSENSITIVITIES
  0.21271D+00 0.56331D+00
CRAMER-RAO LOWER BOUNDS
  0.46229D+00 0.12243D+01
UNCONDITIONAL CORRELATIONS
  0.10000D+01
  0.88786D+00 0.10000D+01
CONDITIONAL CORRELATIONS
  -.10000D+01
  0.88786D+00 -.10000D+01
RESIDUAL RUNS
MEASUREMENT #  1
  NUMBER OF POSITIVE RESIDUALS =   23
  NUMBER OF NEGATIVE RESIDUALS =   21
  EXPECTED NUMBER OF RUNS = 0.22955D+02
  ACTUAL NUMBER OF RUNS =   22
  UNIT NORMAL DEVIATE = -.13897D+00
```

Bibliography

1. Beck, J.V. and K.J. Arnold, Parameter Estimation in Engineering and Science, John Wiley and Sons, 1977.

2. Draper, N.R. and H. Smith, Applied Regression Analysis, John Wiley and Sons, 1966.

3. Iliff, K.W. and L.W. Taylor, Jr., *Determination of Stability and Control Derivatives From Flight Data Using a Newton-Raphson Minimization Technique*, NASA TN D-6579, 1972.

4. Klein, V. and J.G. Batterson and P.C. Murphy, *Determination of Airplane Model Structure From Flight Data by Using Modified Stepwise Regression*, NASA TP1916, 1981.

5. Maine, R.E. and K.W. Iliff, *Identification of Dynamic Systems*, NASA RP1138, 1985.

6. *Methods for Aircraft State and Parameter Identification*, AGARD CP-172, 1975.

7. *Parameter Estimation Techniques and Applications in Aircraft Flight Testing*, NASA TN D-7647, 1974.

8. *Parameter Identification*, AGARD LS-104, 1979.

9. Press, W.H. et al, Numerical Recipes (FORTRAN Version), Cambridge University Press, 1989.

10. Stepner, D.E. and R.K. Mehra, *Maximum Likelihood Identification and Optimal Input Design for Identifying Aircraft Sftability and Control Derivatives*, NASA CR2200, 1973.

Subroutine Gauss_Newton Program Listing

```
      subroutine gauss_newton(n,m,np,z,p,r,chgparam,cond,itermax,
     $                        imnoise,rank,outfile)
C*************************************************************************C
C    Output error routine to estimate parameter values in a model based   C
C    on test data.  Minimization uses the gauss-newton method with        C
C    automatic step size cutting and optional rank-deficient parameter    C
C    updates.  The measurement noise covariance matrix can be fixed or    C
C    automatically updated in an axial iteration scheme.  Insensitivities,C
C    Cramer-Rao lower bounds, unconditional and conditional correlations  C
C    on final parameter estimates are given. Residual run statistics are  C
C    given.                                                               C
C*************************************************************************C
C
C INTEGERS
C n             number of parameters
C m             number of measurements
C np            number of time points in each measurement record
C ndim          parameter for dimensioning arrays---n dimension
C mdim          parameter for dimensioning arrays---m dimension
C npdim         parameter for dimensioning arrays---np dimension
C iter          iteration counter
C itermax       maximum number of iterations allowed
C iconvcrit     number corresponding to convergence criterion satisfied
C ismallcount   number of small SV's
C n1            number of negative residuals
C n2            number of positive residuals
C irun          number of residual runs
C
C REALS
C p             parameter vector (nx1)
C ptest         test parameter vector (nx1)
C dp            parameter step size vector (nx1)
C r             measurement noise cov matrix, diagonal (mxm)
C rj            cost
C rjtest        cost based on test parameter values
C z             measured response (mxnp)
C yhat          model response (mxnp)
C yhattest      model response based on test parameter values
C yhatplus      model response based on perturbed parameter vector
C delyhat       derivative of model response wrt parameters (mxnxnp)
C del1j         cost gradient wrt parameters (1xn)
C del2j         2nd gradient of cost wrt parameters (nxn)
C del1jnorm     norm of cost gradient
C del2jnorm     scaled del2j
C del2jinv      inverse of del2j
C del2jnorminv  inverse of del2jnorm
C s             diagonal scaling matrix to precondition Hessian (nxn)
C w             diagonal matrix (nxn) of singular values
C winv          inverse of w
C u             matrix (nxn) from SVD
C v             matrix (nxn) from SVD
C cond          matrix condition # for determining which SV's are small
C sens          parameter insensitivities (nx1)
C crb           Cramer-Rao bounds on final parameter estimates (nx1)
C ucorr         unconditional correlations (nxn)
C ccorr         conditional correlations (nxn)
C outfile       name of output file
C wmax          value of largest singular value
```

```
C chgparam     max allowable relative change in parameters for convergence
C ave          expected value of residual run distribution
C var          variance of residual run distribution
C znormal      unit normal deviate for residual run distribution
C det          determinate of measurement noise cov matrix
C delp         parameter perturbation
C pdum         parameter vector containing one perturbed parameter
C
C LOGICALS
C imnoise      logical=true for automatic measurement noise cov matrix update
C                     =false for fixed measurement noise covariance matrix
C rank         logical=true if rank deficient updates desired
C conv         logical=true if convergence has been achieved
C                     =false if convergence has not been reached
C wflag        logical vector (nx1)=true if corresponding SV is small
C                                  =false otherwise
C runflag      logical used in calculating number of residual runs

      implicit double precision (a-h,o-z)
      parameter(ndim=30,mdim=10,npdim=500)
      dimension z(mdim,npdim),yhat(mdim,npdim),yhattest(mdim,npdim),
     $   delyhat(mdim,ndim,npdim),del1j(ndim),del2j(ndim,ndim),
     $   del2jnorm(ndim,ndim),del2jinv(ndim,ndim),
     $   del2jnorminv(ndim,ndim),r(mdim),w(ndim),winv(ndim),
     $   u(ndim,ndim),v(ndim,ndim),p(ndim),ptest(ndim),crb(ndim),
     $   dp(ndim),s(ndim),ucorr(ndim,ndim),ccorr(ndim,ndim),sens(ndim)
      logical wflag(ndim),conv,rank,imnoise,runflag
      character*16 outfile

      conv=.false.

C open output file
      open(unit=3,file=outfile,status='unknown')

C model response with starting parameter values
      call model(p,yhat)

C calculate R matrix if it is being updated automatically
      if(imnoise) call mnoise(m,np,z,yhat,r)

C calculate cost
      call cost(m,np,z,yhat,r,rj)

C initialize iteration counter
      iter=-1

C ....................... begin minimization loop .......................C

 100  continue
      iter=iter+1

C gradient of model response wrt parameters
      call gradyhat(n,m,np,p,yhat,delyhat)

C gradient of cost wrt parameters
      call gradcost(n,m,np,z,yhat,r,delyhat,del1j)

C 2nd gradient of cost wrt parameters (gauss approx. to hessian)
```

```
C and hessian with columns scaled by column norm.
      call hessian(n,m,np,delyhat,r,del2j,del2jnorm,s)

C SVD of scaled hessian
      do 10 i=1,n
      do 10 j=1,n
      u(i,j)=del2jnorm(i,j)
 10   continue
      call svdcmp(u,n,n,w,v)

C check spread in singular values to determine if scaled hessian is near
C singular and which singular values are small.
      call smallsv(n,w,cond,wflag,wmax,ismallcount)

C Inverse of scaled and unscaled hessian. Full rank inverse is computed
C if no singular values are small or if rank=.false. or if conv=.true.
C When convergence has been reached, the full rank inverse is always
C computed so that the Cramer-Rao bounds and unconditional correlations
C can be determined. If hessian is very close to singular, full rank inverse
C cannot be computed --- the program will print a divide by zero error and bomb.
      call inverse(n,conv,rank,wflag,w,v,u,s,del2jnorminv,del2jinv)

C write diagnostics to output file
      call diagnostic(n,m,del1j,conv,iconvcrit,iter,rj,
     $                ismallcount,wflag,wmax,w,u,v,r,p)

C if convergence was reached on last iteration, exit loop
      if(conv) go to 200

C parameter step size and test update to parameter vector
      do 20 i=1,n
      dp(i)=0.0d+00
      do 20 j=1,n
      dp(i)=dp(i)-del2jinv(i,j)*del1j(j)
 20   continue

C test update to parameter vector
 300  do 40 i=1,n
      ptest(i)=p(i)+dp(i)
 40   continue

C model response and cost based on test parameters
      call model(ptest,yhattest)
      call cost(m,np,z,yhattest,r,rjtest)

C cut step size by 1/2 if cost was not reduced
      if(rjtest.gt.rj) then
        do 50 i=1,n
        dp(i)=.5d+00*dp(i)
 50     continue
        go to 300
      endif

C check convergence criteria
      call convergence(n,iter,itermax,ptest,p,chgparam,conv,
     $                 iconvcrit)

C cost has been reduced --- update parameter vector,model response,cost
```

```
      do 60 i=1,n
      p(i)=ptest(i)
 60   continue

      do 70 i=1,m
      do 70 j=1,np
      yhat(i,j)=yhattest(i,j)
 70   continue

      rj=rjtest

C automatic measurement noise covariance matrix update, if desired
      if(imnoise) call mnoise(m,np,z,yhat,r)

      go to 100

C .................. end minimization iteration loop ....................C

 200  continue

C parameter insensitivities, Cramer-Rao lower bounds, correlations
      do 80 i=1,n
      sens(i)=1.0d+00/dsqrt(del2j(i,i))
      crb(i)=dsqrt(del2jinv(i,i))
      do 80 j=1,i
      ucorr(i,j)=del2jinv(i,j)/
     $           dsqrt(del2jinv(i,i)*del2jinv(j,j))
      ccorr(i,j)=-del2j(i,j)/dsqrt(del2j(i,i)*del2j(j,j))
 80   continue

C write insensitivities, CRB, correlations to output file
      write(3,11)
      write(3,15)(sens(i),i=1,n)
      write(3,12)
      write(3,15)(crb(i),i=1,n)
      write(3,13)
      do 90 i=1,n
      write(3,15)(ucorr(i,j),j=1,i)
 90   continue
      write(3,14)
      do 110 i=1,n
      write(3,15)(ccorr(i,j),j=1,i)
 110  continue

C residual runs
      write(3,16)
C calculate # of positive residuals, # of negative residuals, # of runs
      do 130 i=1,m
      n1=0
      n2=0
      irun=1
      if((z(i,1)-yhat(i,1)).gt.0.0d+00) runflag=.true.
      if((z(i,1)-yhat(i,1)).lt.0.0d+00) runflag=.false.
      do 120 j=1,np
      if((z(i,j)-yhat(i,j)).gt.0.0d+00) then
        n1=n1+1
        if(.not.runflag) irun=irun+1
        runflag=.true.
      else if((z(i,j)-yhat(i,j)).lt.0.0d+00) then
        n2=n2+1
        if(runflag) irun=irun+1
        runflag=.false.
      endif
 120  continue
C mean, variance, unit normal deviate for runs
      rn1=dfloat(n1)
      rn2=dfloat(n2)
      dum1=2.*rn1*rn2
      ave=1.+dum1/(rn1+rn2)
      var=dum1*(dum1-rn1-rn2)/((rn1+rn2)*(rn1+rn2)*(rn1+rn2-1.))
      znormal=(dfloat(irun)-ave+.5)/dsqrt(var)
      write(3,17)i
      write(3,18)n1
      write(3,19)n2
      write(3,21)ave
      write(3,22)irun
      write(3,23)znormal
 130  continue

 11   format(1x,' PARAMETER INSENSITIVITIES')
 12   format(1x,' CRAMER-RAO LOWER BOUNDS')
 13   format(1x,' UNCONDITIONAL CORRELATIONS')
 14   format(1x,' CONDITIONAL CORRELATIONS')
 15   format(1x,'  ',30(1x,d11.5))
 16   format(1x,' RESIDUAL RUNS')
 17   format(1x,' MEASUREMENT # ',i2)
 18   format(1x,'   NUMBER OF POSITIVE RESIDUALS = ',i4)
 19   format(1x,'   NUMBER OF NEGATIVE RESIDUALS = ',i4)
 21   format(1x,'   EXPECTED NUMBER OF RUNS = ',d11.5)
 22   format(1x,'   ACTUAL NUMBER OF RUNS = ',i4)
 23   format(1x,'   UNIT NORMAL DEVIATE = ',d11.5)

      close(unit=3)

      return
      end

C***********************************************************************C
C             measurement noise covariance matrix update                C
C***********************************************************************C
      subroutine mnoise(m,np,z,yhat,r)
      implicit double precision (a-h,o-z)
      parameter(ndim=30,mdim=10,npdim=500)
      dimension z(mdim,npdim),yhat(mdim,npdim),r(mdim)

      do 10 i=1,m
      sum=0.0d+00
      do 20 j=1,np
      sum=sum+(z(i,j)-yhat(i,j))*(z(i,j)-yhat(i,j))
 20   continue
      r(i)=sum/dfloat(np)
 10   continue

      return
      end
```

```
C**********************************************************************C
C                           cost function                              C
C**********************************************************************C
      subroutine cost(m,np,z,yhat,r,rj)
      implicit double precision (a-h,o-z)
      parameter(ndim=30,mdim=10,npdim=500)
      dimension z(mdim,npdim),yhat(mdim,npdim),r(mdim)

C cost function
      det=1.0d+00
      rj=0.0d+00
      do 10 i=1,m
      det=det*r(i)
      do 10 j=1,np
      rj=rj+((z(i,j)-yhat(i,j))*(z(i,j)-yhat(i,j)))/r(i)
 10   continue
      rj=.5*(rj+dfloat(np)*dlog(det))

      return
      end

C**********************************************************************C
C                gradient of model response wrt parameters             C
C                        forward difference                            C
C**********************************************************************C
      subroutine gradyhat(n,m,np,p,yhat,delyhat)
      implicit double precision (a-h,o-z)
      parameter(ndim=30,mdim=10,npdim=500)
      dimension yhat(mdim,npdim),yhatplus(mdim,npdim),
     $          delyhat(mdim,ndim,npdim),p(ndim),pdum(ndim)

C relative precision of floating point arithmatic
C VAX double precision
C      data eps/2.8d-17/
C standard double precision
      data eps/2.2d-16/

      deps=dsqrt(eps)

C initialize dummy parameter vector
      do 10 i=1,n
      pdum(i)=p(i)
 10   continue

C forward difference
      do 20 i=1,n
      delp=deps*dmax1(dabs(pdum(i)),.1d+00)
      pdum(i)=pdum(i)+delp
      call model(pdum,yhatplus)
      do 30 j=1,m
      do 30 k=1,np
      delyhat(j,i,k)=(yhatplus(j,k)-yhat(j,k))/delp
 30   continue
      pdum(i)=p(i)
 20   continue

      return
      end
```

```
C******************************************************************************C
C                 gradient of cost wrt paramters                               C
C******************************************************************************C
      subroutine gradcost(n,m,np,z,yhat,r,delyhat,dellj)
      implicit double precision (a-h,o-z)
      parameter(ndim=30,mdim=10,npdim=500)
      dimension z(mdim,npdim),yhat(mdim,npdim),r(mdim),dellj(ndim),
     $           delyhat(mdim,ndim,npdim)

      do 10 i=1,n
      sum=0.0d+00
      do 20 j=1,m
      do 20 k=1,np
      sum=sum+(z(j,k)-yhat(j,k))*delyhat(j,i,k)/r(j)
 20   continue
      dellj(i)=-sum
 10   continue

      return
      end

C******************************************************************************C
C                     gauss approximation to hessian                           C
C           scale columns of hessian by column norm to minimize                C
C                scale effects on spread in singular values                    C
C******************************************************************************C
      subroutine hessian(n,m,np,delyhat,r,del2j,del2jnorm,s)
      implicit double precision (a-h,o-z)
      parameter(ndim=30,mdim=10,npdim=500)
      dimension delyhat(mdim,ndim,npdim),r(mdim),del2j(ndim,ndim),
     $           del2jnorm(ndim,ndim),s(ndim)

      do 10 i=1,n
      sum1=0.0d+00
      do 20 j=1,n
      sum2=0.0d+00
      do 30 l=1,np
      do 30 k=1,m
      sum2=sum2+delyhat(k,j,l)*delyhat(k,i,l)/r(k)
 30   continue
      del2j(j,i)=sum2
      sum1=sum1+sum2*sum2
 20   continue
      s(i)=1.0d+00/dsqrt(sum1)
      do 40 j=1,n
      del2jnorm(j,i)=del2j(j,i)*s(i)
 40   continue
 10   continue

      return
      end

C******************************************************************************C
C              determine which singular values are small                       C
C******************************************************************************C
      subroutine smallsv(n,w,cond,wflag,wmax,ismallcount)
      implicit double precision (a-h,o-z)
      parameter(ndim=30,mdim=10,npdim=500)
      dimension w(ndim)
      logical wflag(ndim)

C find maximum singular value
      wmax=0.0d+00
      do 10 i=1,n
      if(w(i).gt.wmax) wmax=w(i)
 10   continue

C find small SV's
      ismallcount=0
      do 20 i=1,n
      if(w(i)/wmax.lt.1.0d+00/cond) then
        wflag(i)=.true.
        ismallcount=ismallcount+1
      else
        wflag(i)=.false.
      endif
 20   continue

      return
      end

C***************************************************************************C
C                inverse of scaled and unscaled hessian                     C
C***************************************************************************C
      subroutine inverse(n,conv,rank,wflag,w,v,u,s,del2jnorminv,
     $                   del2jinv)
      implicit double precision (a-h,o-z)
      parameter(ndim=30,mdim=10,npdim=500)
      dimension w(ndim),v(ndim,ndim),u(ndim,ndim),winv(ndim),
     $          del2jinv(ndim,ndim),del2jnorminv(ndim,ndim),s(ndim)
      logical wflag(ndim),conv,rank

C form inverse of w matrix, zero small SV elements if desired
      do 10 i=1,n
      if((.not.conv).and.(rank).and.(wflag(i))) then
        winv(i)=0.0d+00
      else
        winv(i)=1.0d+00/w(i)
      endif
 10   continue

C invert scaled hessian and unscale it
      do 20 i=1,n
      do 20 j=1,n
      sum=0.0d+00
      do 30 k=1,n
      sum=sum+v(i,k)*winv(k)*u(j,k)
 30   continue
      del2jnorminv(i,j)=sum
      del2jinv(i,j)=s(i)*sum
 20   continue

      return
      end
```

```
C*************************************************************************C
C                write diagnostics to output file                         C
C*************************************************************************C
      subroutine diagnostic(n,m,dellj,conv,iconvcrit,iter,rj,
     $                      ismallcount,wflag,wmax,w,u,v,r,p)
      implicit double precision (a-h,o-z)
      parameter(ndim=30,mdim=10,npdim=500)
      dimension dellj(ndim),w(ndim),u(ndim,ndim),v(ndim,ndim),
     $          r(mdim),p(ndim)
      logical wflag(ndim),conv

C norm of cost gradient
      sum=0.0d+00
      do 100 i=1,n
      sum=sum+dellj(i)*dellj(i)
 100   continue
      delljnorm=dsqrt(sum)

      if(conv) write(3,5)iconvcrit
      write(3,10)iter
      write(3,20)rj
      write(3,30)delljnorm
      write(3,40)ismallcount
      if(ismallcount.gt.0) then
        do 200 i=1,n
        if(wflag(i)) then
          write(3,50)w(i)/wmax
          write(3,60)
          write(3,70)(v(j,i),j=1,n)
        endif
 200    continue
      endif
      write(3,80)
      do 300 i=1,n
      if(.not.wflag(i)) write(3,70)(u(j,i),j=1,n)
 300  continue
      write(3,90)
      write(3,110)(r(i),i=1,m)
      write(3,120)
      write(3,130)(p(i),i=1,n)

 5    format(//,1x,' !!!!!!! CONVERGENCE BY CRITERIA #',i1,' !!!!!!!!!')
 10   format(//,1x,' *************** ITERATION ',i2,' ***************')
 20   format(1x,' COST = ',d11.5)
 30   format(1x,' NORM OF COST GRADIENT = ',d11.5)
 40   format(1x,' NUMBER OF SMALL SINGULAR VALUES OF HESSIAN = ',i2)
 50   format(1x,'    RATIO SVsmall/SVmax = ',d11.5)
 60   format(1x,'    BASIS VECTOR FOR NULL SPACE = ')
 70   format(1x,'  ',30(1x,f8.5))
 80   format(1x,' BASIS VECTORS (ROW VECTORS) FOR RANGE SPACE = ')
 90   format(1x,' DIAGONALS OF MEASUREMENT NOISE COVARIANCE MATRIX')
 110  format(1X,'  ',10(1X,d10.4))
 120  format(1x,' PARAMETER VECTOR')
 130  format(1x,'  ',30(1x,d11.5))

      return
      end
```

```
C**************************************************************************C
C         check convergence criteria and set convergence flag              C
C**************************************************************************C
      subroutine convergence(n,iter,itermax,ptest,p,chgparam,conv,
     $                       iconvcrit)
      implicit double precision (a-h,o-z)
      parameter(ndim=30,mdim=10,npdim=500)
      dimension ptest(ndim),p(ndim)
      logical conv

C criterion #0: max iterations exceeded
      if(iter.eq.(itermax-1)) then
        conv=.true.
        iconvcrit=0
      endif

C criterion #1: all parameter values changed by less than a factor of
C chgparam during update
      do 10 i=1,n
      if(dabs((ptest(i)-p(i))/p(i)).gt.chgparam) go to 100
 10   continue
      conv=.true.
      iconvcrit=1

 100  return
      end

C**************************************************************************C
C                   singular value decomposition                           C
C                 algorithm from Numerical Recipes                         C
C**************************************************************************C
      subroutine svdcmp(a,m,n,w,v)
      implicit double precision (a-h,o-z)
      parameter(ndim=30,mdim=10,npdim=500)
      dimension a(ndim,ndim),w(ndim),v(ndim,ndim),rv1(ndim)

      g=0.0d+00
      scale=0.0d+00
      anorm=0.0d+00
      do 25 i=1,n
      l=i+1
      rv1(i)=scale*g
      g=0.0d+00
      s=0.0d+00
      scale=0.0d+00
      if(i.le.m) then
        do 11 k=i,m
        scale=scale+dabs(a(k,i))
 11     continue
        if(scale.ne.0.0d+00) then
          do 12 k=i,m
          a(k,i)=a(k,i)/scale
          s=s+a(k,i)*a(k,i)
 12       continue
          f=a(i,i)
          g=-dsign(dsqrt(s),f)
          h=f*g-s
          a(i,i)=f-g
          if(i.ne.n) then
            do 15 j=1,n
            s=0.0d+00
            do 13 k=i,m
            s=s+a(k,i)*a(k,j)
13          continue
            f=s/h
            do 14 k=i,m
            a(k,j)=a(k,j)+f*a(k,i)
14          continue
15          continue
          endif
          do 16 k=i,m
          a(k,i)=scale*a(k,i)
16        continue
        endif
      endif
      w(i)=scale*g
      g=0.0d+00
      s=0.0d+00
      scale=0.0d+00
      if((i.le.m).and.(i.ne.n)) then
        do 17 k=l,n
        scale=scale+dabs(a(i,k))
17      continue
        if(scale.ne.0.0d+00) then
          do 18 k=l,n
          a(i,k)=a(i,k)/scale
          s=s+a(i,k)*a(i,k)
18        continue
          f=a(i,l)
          g=-dsign(dsqrt(s),f)
          h=f*g-s
          a(i,l)=f-g
          do 19 k=l,n
          rvl(k)=a(i,k)/h
19        continue
          if(i.ne.m) then
            do 23 j=l,m
            s=0.0d+00
            do 21 k=l,n
            s=s+a(j,k)*a(i,k)
21          continue
            do 22 k=l,n
            a(j,k)=a(j,k)+s*rvl(k)
22          continue
23          continue
          endif
          do 24 k=l,n
          a(i,k)=scale*a(i,k)
24        continue
        endif
      endif
      anorm=dmaxl(anorm,(dabs(w(i))+dabs(rvl(i))))
25    continue

      do 32 i=n,1,-1
      if(i.lt.n) then
```

```
        if(g.ne.0.0d+00) then
          do 26 j=1,n
          v(j,i)=(a(i,j)/a(i,l))/g
26        continue
          do 29 j=1,n
          s=0.0d+00
          do 27 k=l,n
          s=s+a(i,k)*v(k,j)
27        continue
          do 28 k=l,n
          v(k,j)=v(k,j)+s*v(k,i)
28        continue
29        continue
        endif
        do 31 j=l,n
        v(i,j)=0.0d+00
        v(j,i)=0.0d+00
31      continue
      endif
      v(i,i)=1.0d+00
      g=rv1(i)
      l=i
32    continue

      do 39 i=n,1,-1
      l=i+1
      g=w(i)
      if(i.lt.n) then
        do 33 j=l,n
        a(i,j)=0.0d+00
33      continue
      endif
      if(g.ne.0.0d+00) then
        g=1.0d+00/g
        if(i.ne.n) then
          do 36 j=l,n
          s=0.0d+00
          do 34 k=l,m
          s=s+a(k,i)*a(k,j)
34        continue
          f=(s/a(i,i))*g
          do 35 k=i,m
          a(k,j)=a(k,j)+f*a(k,i)
35        continue
36        continue
        endif
        do 37 j=i,m
        a(j,i)=a(j,i)*g
37      continue
      else
        do 38 j=i,m
        a(j,i)=0.0d+00
38      continue
      endif
      a(i,i)=a(i,i)+1.0d+00
39    continue

      do 49 k=n,1,-1
      do 48 its=1,30
      do 41 l=k,1,-1
      nm=l-1
      if((dabs(rvl(l))+anorm).eq.anorm) go to 2
      if((dabs(w(nm))+anorm).eq.anorm) go to 1
41    continue
1     c=0.0d+00
      s=1.0d+00
      do 43 i=1,k
      f=s*rvl(i)
      rvl(i)=c*rvl(i)
      if((dabs(f)+anorm).eq.anorm) go to 2
      g=w(i)
      h=dsqrt(f*f+g*g)
      w(i)=h
      h=1.0d+00/h
      c=g*h
      s=-f*h
      do 42 j=1,m
      y=a(j,nm)
      z=a(j,i)
      a(j,nm)=y*c+z*s
      a(j,i)=-y*s+z*c
42    continue
43    continue
2     z=w(k)
      if(l.eq.k) then
        if(z.lt.0.0d+00) then
          w(k)=-z
          do 44 j=1,n
          v(j,k)=-v(j,k)
44        continue
        endif
        go to 3
      endif
      if(its.eq.30) print*,' No SV convergence in 30 iterations'
      x=w(l)
      nm=k-1
      y=w(nm)
      g=rvl(nm)
      h=rvl(k)
      f=((y-z)*(y+z)+(g-h)*(g+h))/(2.0d+00*h*y)
      g=dsqrt(f*f+1.0d+00)
      f=((x-z)*(x+z)+h*((y/(f+dsign(g,f)))-h))/x

      c=1.0d+00
      s=1.0d+00
      do 47 j=1,nm
      i=j+1
      g=rvl(i)
      y=w(i)
      h=s*g
      g=c*g
      z=dsqrt(f*f+h*h)
      rvl(j)=z
      c=f/z
      s=h/z
      f=x*c+g*s
      g=-x*s+g*c
      h=y*s
      y=y*c
      do 45 jj=1,n
      x=v(jj,j)
      z=v(jj,i)
      v(jj,j)=x*c+z*s
      v(jj,i)=-x*s+z*c
45    continue
      z=dsqrt(f*f+h*h)
      w(j)=z
      if(z.ne.0.0d+00) then
        z=1.0d+00/z
        c=f*z
        s=h*z
      endif
      f=c*g+s*y
      x=-s*g+c*y
      do 46 jj=1,m
      y=a(jj,j)
      z=a(jj,i)
      a(jj,j)=y*c+z*s
      a(jj,i)=-y*s+z*c
46    continue
47    continue
      rv1(l)=0.0d+00
      rv1(k)=f
      w(k)=x
48    continue
3     continue
49    continue

      return
      end
```

APPENDIX C

RESEARCH PLAN

Objective

The design and fabrication of medium to low cost simulators requires degradation of the fidelity of one or more parameters or systems in the simulator. A reliable quantitative method for specifying the acceptable degree of degradation (dependent on the mission/task being simulated) does not exist. The objective of this effort is to devise and validate such a methodology.

Background

Specification of the desired degree of fidelity for a high cost, full mission simulator is relatively easy. Generally, the buyer and the builder strive to the maximum fidelity achievable within the state of the art and within the money available. It has been long recognized that such simulators represent gross "overkill" for many training tasks and that more cost effective training can be provided with simulators whose fidelity is "good enough" for the training under consideration. The same rationale applies to engineering simulators. Often, the engineering tasks to be accomplished do not require "gold-plated" simulators. These "selective fidelity" simulators are characterized by limited fidelity on selected systems for specific performance tasks. Typical systems whose fidelity might be intentionally degraded include visual, tactile, motion, auditory, mathematical models, etc. Specifying the degree of fidelity degradation and the systems to be degraded is a much more difficult task, primarily because there does not exist a reliable quantitative method for such specification. Currently, the "trial and error" method, expensive in terms of time and manpower, is most widely used. The designer, based on the tasks to be studied, makes a "best guess" at the systems and parameters that are most important. A trial system is fabricated, and customer pilots and/or engineers are asked to evaluate the device. Several iterations of redesign and re-evaluation occur until an acceptable device is produced. If a methodology could be devised and validated to permit the quantitative determination of fidelity levels required for the design missions/tasks, considerable savings could be realized in the production of training devices and improvements in customer satisfaction could be achieved.

Proposed Approach

The Georgia Tech Center of Excellence for Rotary Wing Aircraft Technology (CERWAT) under contract to the University of Central Florida's (UCF) Institute for Simulation and Training (IST) has developed a methodology to permit the quantitative determination of fidelity levels for the required simulation task. This methodology has been developed as part of a DARPA/PM-TRADE Aviation Technology Research Plan. While this methodology provides a framework, it must be evaluated to determine its viability and usefulness.

The methodology developed has drawn from a variety of sources. Much assistance was provided by the U.S. Army Aeroflightdynamics Directorate and a draft version has been sent to numerous agencies for comment. The methodology can be considered a modified FAA and ADS33 approach, although much of the structure has been based on recommendations by the NATO Advisory Group for Aerospace Research and Development (AGARD). Phase II consists of a Simulation and Modeling effort as well as Verification and Validation.

Under funding by the PM-TRADE and the Army Research Institute (ARI), Georgia Tech has developed a real time rotorcraft simulation (FLIGHT SIM) laboratory. FLIGHT SIM combines the capabilities of the Georgia Tech Center of Excellence for Rotary Wing Aircraft Technology (CERWAT) and the NASA loaned Rotorcraft Digital Advanced Avionics System (RODAAS) with the real time rotorcraft parallel processing capabilities of the Advanced Rotorcraft Technology Inc. (ART) FlightLab to establish a university unique research and training evaluation facility. It is proposed to use the Georgia Tech FLIGHT SIM as one of the testbeds for evaluating the methodology. The AH-64 Apache is being modeled with the FlightLab software and is being compared with the McDonnell Douglas Helicopter Company (MDHC) FLYRT simulation program for initial correlation. It is proposed to use system identification methods on the AH-64 flight test data for math model improvement and parameter tolerance validation. U.S. Army operational/test pilots and other military service pilots and engineers, who are at Georgia Tech's CERWAT for advanced degrees and training, will be used for functional/subjective test verification. Thus, it can be seen that the proposed program leverages substantially government funding and involves military graduate students in a high payoff research area. Phase III of the proposed program would provided published results and produce a methodology for service evaluation on different rotorcraft simulators.

Described below is a two-year experimental effort which will methodically analyze the components of simulations and their interactions with each other and the pilot. In this study, the operator and the simulator are considered as a closed loop system. As such, it is necessary to determine the characteristics of each block in the system. The study will baseline and diagram the characteristics of an ideal system for rotary wing aircraft) based on a detailed analysis of the actual system (e.g., the design basis vehicle with the operator in the loop) using actual flight test data. Ideal system design characteristics will be determined via measurement of critical parameters (to be determined). Changes in control blocks will then be studied to determine the effects on the closed loop system response when the characteristics of individual blocks are varied. The blocks subject to change can include the human performance characteristics (i.e., novice versus experienced pilot), forcing function (i.e., mission characteristics), or vehicle/simulation model characteristics (e.g., visual system field of view). Gross changes will first be studied to refine test methods, and to determine key parameters based upon parameter sensitivity. The following is an outline of specific tasks proposed in this research plan.

1. Select a target vehicle for study. The proposed study aircraft would be the AH-64 Apache. Extensive flight test data exist for this vehicle and more detailed data are being developed. Representative sets of maneuvers will be selected for study. For example, attack helicopter mission scenarios will be analyzed and maneuvers important to mission accomplishment will be selected.

2a. Review existing flight test data for the selected maneuvers to determine pilot/system performance. System performance measurements will be separated by maneuver and pilot experience level. It is anticipated that pilot experience levels can be extracted from records on hand in the flight test organization. If pilots who performed the tests are still available, interviews will be conducted to obtain more in-depth data.

2b. Where existing flight test data are not sufficient, test organizations will be requested to collect additional data. This testing would be "piggy-backed" on on-going test on a non-interference basis to avoid the expense and time delays inherent in the scheduling and funding of dedicated flight testing.

3. Determine validity of system performance measurement techniques using Cooper-Harper rating, Frequency Weighted Task Complexity Index, Kalman Filter of stick activity, etc. Explore other

psychophysical measurements techniques. Select most valid technique(s) for use in the study.

4. Perform linear and nonlinear system identification on the target vehicles and simulators. Four methods of system identification are used to perform linear and nonlinear system identification of fixed wing and rotary wing aircraft and to determine the minimal model order (degrees of freedom) that is required to represent the nonlinear vehicle in the simulation. Two different types of techniques will be used to perform system identification of the linear region; one is parametric in nature and the other is non-parametric. In the non-parametric case, frequency domain analysis will be used to derive qualitative characteristics (e.g., natural frequencies, damping ratios, poles, zeros and gains of input-output transfer functions, etc.) from vehicle flight test data; the results will be compared with any available linear model of the vehicle and an improvement in that model will be made where appropriate. In the parametric case, a maximum-likelihood system identification technique will be used to process the flight test data in deriving a linear aerodynamic model (e.g., stability and control derivatives) of the vehicle; the results will be compared with the frequency domain analysis and with any available linear model of the vehicle and an improvement in the linear model will be made where appropriate. The estimation-before-modeling (EBM) technique will be used to perform system identification of the nonlinear region. The EBM technique is a two step approach. In the first step, estimation techniques (e.g., extended Kalman filtering and smoothing) are used to process each individual flight test maneuver or data record to obtain smooth time histories of vehicle states and smooth time histories of non-dimensional aerodynamic forces and moments acting on the vehicle. In the second step, the data results from all maneuvers of the first step are distributed into small subspaces on which non-linear modeling of the vehicle is performed using multiple regression analysis; after which the subspace models are combined to form a global nonlinear model. Next, since a global nonlinear model is well defined with respect to the flight test data, a maximum-likelihood system identification technique will be used to process the flight test data and fine tune as necessary the global nonlinear model. The results will be compared with any available nonlinear model of the vehicle and an improvement of that nonlinear model will be made where appropriate.

5. Compare the airloads generated in the simulator to those observed in the aircraft and adjust the simulator to match the aircraft. The objective is to establish, improve, update, and validate the simulation fidelity for a given vehicle. The nonlinear simulation model will be modified to include the minimum degrees of freedom as determined from

Task 4. The system identification techniques of Task 4 will be used if necessary to refine the linear and nonlinear models of the simulation so that its time responses will better match the time responses of the flight test data. This task will be used in feedback fashion with Task 4 to derive the best possible final nonlinear simulation model of a vehicle. This task includes a final handling qualities specification evaluation.

6. Catalog simulator features (based on criteria to be developed). Determine the possible ranges of fidelity for systems on the simulators to be used in the study. Create a simulator test matrix.

7. Run target simulator for quantitative performance measurements of maneuvers selected in Task 1. A pilot model (in software) will be used to replicate the pilot inputs from the flight test data. The data acquisition system will be refined and validated.

8. Run target simulator with a pilot in the loop. Where feasible, pilots who flew flight test maneuvers should be used. Pilot performance will be compared based upon the techniques selected in Task 3. Perform variations and adjustments as described in tasks 10-12 below in an iterative fashion.

9. Create a software program to adjust simulator parameters and systems in tasks 10-12 below.

10. Adjust dynamics model (inertias, additional/fewer stability derivatives, integration routines, time increments). Iterate with Task 8.

11. Vary simulator parameters (e.g., visual system, force feel system, etc.) based upon parameters determined in Task 6. Iterate with Task 8.

12. Vary activation/fidelity of aircraft systems simulated. Iterate with Task 8.

13. Prepare documentation based upon data generated. Generate reports, specification and recommendation. This can be visualized as providing a "Buyer's Guide to Selective Fidelity Simulators."

Recommendation

The proposed research is sorely needed by the simulation user community and should be started at the earliest possible date. Georgia

Tech has considerable expertise and on-going efforts in the simulation of aircraft systems and in system identification. CERWAT is a world-recognized leader in rotorcraft research and has established a state of the art rotorcraft simulation capability for high fidelity blade element math modeling. This institution is uniquely qualified to perform this research.

APPENDIX D

Addendum
to
Final report for University of Central Florida Subcontract
Purchase Order Number 101700

Review of the subcontract does not reveal a simulation networking requirement, however, in verbal discussions, agreement was made to provide information relative to networking. That information is provided in this addendum.

This work was performed for PM-TRADE and is the subject of discussions related to funding of the follow-on work described herein. Please note that it is proposed that networking of simulators be performed with UCF in the proposed Phase II. More detailed discussions will be held concerning this effort at a later date.

SIMNET CONNECTION CONSIDERATIONS

OBJECTIVE

Under funding from the U.S. Army Program Manager for Training Devices (PM-TRADE) and the Army Research Institute (ARI), a flight simulation laboratory (FLIGHT SIM) is being established at Georgia Tech. The FLIGHT SIM Laboratory will initially study man-in-the-loop real-time rotorcraft flight simulation for training, evaluation, and integrated mathematical model development. Real-time helicopter and tiltrotor aircraft simulations have traditionally neglected high frequency phenomena in order to achieve real-time operation on affordable computers. FLIGHT SIM will be able to study these higher order phenomena in real time by incorporating a parallel computing architecture based around an eight-processor Silicon Graphics VTX380 Powervision series workstation. This computing power communicates through an independent intelligent parallel processing interface with the Rotorcraft Digital Advanced Avionics System (RODAAS) simulator cab which serves as a pilot workstation. The entire FLIGHT SIM system is capable of performing as a node within the SIMNET network.

NETWORK GOALS

The first SIMNET site was activated in 1986 at Ft. Knox, KY. Later in 1987 a second SIMNET site as activated at the Grafenwoehr training center, FRG. A 384 Kbyte fiber-optic link to Europe (termed the "fat pipe") allows trans-Atlantic communication to Germany on a shared basis with NASA. The SIMNET network currently extends between Ft. Rucker AL, Ft. Knox KY, Ft. Leavenworth KS through a T1 backbone across the continental United States which is accessible by simulator sites such as the Institute for Defense Analyses in Rosslyn Virginia, Ft. Hood, TX, and Fts. Benning and Stewart in Georgia.

Most of the existing network consists of T1 connections and accommodates 64 kbs data rates (a T1 sub-channel equivalent to the "DS-0" rate, or the capacity of a single Pulse-Code Modulated (PCM) voice channel). Both Ft. Rucker and Ft. Knox are predominantly *operational* sites where training is ongoing. Additional operational sites will be added to the network, but so will *experimental* sites. Experimental sites will likely operate on a non-interference basis with the operational sites and will tend to interact with the operational sites for the purpose of developing better simulators and protocols.

The Georgia Tech FLIGHT SIM laboratory will fall in the category of an experimental site. Other experimental sites might be the Institute for Simulation and Training at the University of Central Florida, Orlando FL, or the Stanford Research Institute in California.

NETWORK REQUIREMENTS

In 1983 the Department of Defense initiated an advanced large scale research program in simulation networking known as SIMNET. During this program the core technologies for constructing large networks of object-oriented war fighting simulators were developed.

SIMULATOR-TO-LONG-HAUL-NETWORK

Because SIMNET simulators must continuously communicate in real time, a physical network having adequate bandwidth to accommodate simultaneous operation of multiple remote simulators had to be established. To reduce the bandwidth requirement of this network, each simulator is required to maintain its own version of reality based only upon status changes received from the other participating simulators. Therefore, each simulator creates its own visual environment (including its own visual renditions of the various participating vehicle simulators), and computes the current positions and attributes of other simulated vehicles based on the most recent position, velocity, and attribute update received from each vehicle. A stationary tank may only confirm its inactive status every few seconds, whereas a jinking A-10 firing its weapon might provide a rapid stream of position, velocity, and weapons updates to the other participating simulators.

Long Haul Networking

Since only *changes* in a simulator's status are communicated between remote simulators, the bandwidth of the SIMNET network can be satisfied with leased T1 lines. The term "Long Haul" networking simply refers to long distance digital communications between simulator sites.

T1 was developed by the Bell System as a means of dealing economically with the issues of inter-central office communication in the early 1960's. The initial application of T1 technology replaced 24 pairs of copper interoffice trunks with a single twin copper pair providing the same transmission capacity. Basically, 24 voice signals were digitized and then multiplexed into a single 1.544 Mbps data stream transmitted over two pairs of wire. Since then, several levels of T1 have been established. "DS-1" for "Digital first generation" refers to the common 1.544 Mbps communication technology. "DS-2" through "DS-4" allow transmission up to 274.176 Mbps. By dividing the 1.544 Mbps DS-1 T1 channel into 24 partitioned segments, twenty four 64-kbps "DS-0" channels may be formed. The method of transmitting signalling information with each of the twenty four DS-0 channels of the T1 carrier is called "in-band" signalling.

Simulator-to-Ethernet

At the Georgia Institute of Technology, interdepartmental digital communication is achieved over an existing network of Ethernet lines. Ethernet-to-T1 routers (discussed below) are common off-the-shelf equipment. Therefore the most logical and convenient method for interfacing the FLIGHT SIM simulator to SIMNET's T1 network will be through an Ethernet-to-T1 bridge/router.

TCP/IP PROTOCOL STANDARD AT GEORGIA TECH

T1 and Ethernet digital multiplexing schemes determine how digital information is *physically* divided and transmitted over a pair of wires (or substitute, such as fiber optic cables or microwave channels). The coding of the information transmitted via T1 is independent of the T1 standard provided that it remains bandwidth compatible.

The standard for network information encoding at the Georgia Institute of Technology is Transmission Control Protocol/ Internet Protocol (TCP/IP) which places information into packets and seizes control of an Ethernet network line for transmission of these information packets without collision. A protocol called Carrier Sense Multiple Access Carrier Detect (CSMA/ CD) allows multiple systems to interface over TCP/IP without stepping on each other's transmissions.

Ethernet-to-Data Terminal Unit

In order to communicate between SIMNET simulators over a long haul network, an interface must exist that is capable of translating both electrical protocols and software protocols. The standard electrical interconnection scheme used at the Fort Rucker simulation facility is Ethernet (Not only is this the same physical standard used at Georgia Tech, but it is the common standard adopted by the University of Central Florida and several other potential SIMNET sites). The hardware interface used at Fort Rucker is the Bolt-Beranek-Newman, Inc (BBN) "Butterfly." BBN is currently responsible for the hardware and long haul networking at Fort Rucker. Other bridge/routers are also available on the open market as well.

BBN BUTTERFLY

The BBN Butterfly bridge/router is exclusive to BBN and is the equipment chosen to implement currently existing SIMNET sites. The BBN Butterfly is not an off-the-shelf product, and is built to order. Generally DARPA specifies and orders Butterfly units as new SIMNET nodes are brought on line. Single Butterfly units cost approximately $20,000 a piece. Since each unit is made to order, no vendor data sheet is available for inclusion in this report.

RAD BOX

An example of a commercially available alternative to the BBN Butterfly is the PC-based REB-SX bridge/router manufactured by RAD. The REB-SX is a PC-AT-based two slot card set that interfaces ethernet to T1 while operating transparently to any higher layer protocol including TCP/IP. The REB-SX accommodates data rates up to 2,048 Mbps. The two-card set, software, cabling, and the host PC-AT costs approximately $6,000. A data sheet for the REB-SX series remote ethernet bridge/router is provided in the appendix.

Data Terminal Unit-to-T1

A Channel Service Unit/Data Service Unit (CSU/DSU) is required to interface a bridge/router to a T1 line because bridge/routers such as the BBN Butterfly and the REB-SX convert ethernet digital signals to the digital protocol expected on a T1 line, but do not modulate the digital information for transmission. A CSU/DSU provides an interface directly to the T1 line. A similar unit is required to demodulate the signal at the receiving end. A CSU/DSU such as the RAD FCD-1 costs approximately $1,200. A data sheet for the FCD-1 series remote ethernet bridge/router is provided in the appendix.

PROPOSED NETWORKING ARRANGEMENT

A survey of government-owned T1 lines between Atlanta and Fort Rucker, Alabama revealed that none of the facilities in the Atlanta vicinity have such capability specifically dedicated to data transmission. Dobbins Air Force Base and the adjacent Naval Air Station have no T1 capability. Both Fort Rucker and Fort Macpherson (south of Atlanta) have connection to the Defence Commercial Telephone Network (DCTN) which supports teleconferencing over T1 lines. If Georgia Tech were to tap into this conduit to Fort Rucker, data transmission would be possible, however it would likely have to be on a non-interference basis with regular teleconferencing activities originating at either end. The issue of how to tap into the Fort Macpherson DCTN node (if the Army were to approve of such at all) is still a problem. A leased T1 line would have to be run from Georgia Tech to Fort Macpherson. Though nowhere near the expense of maintaining a leased line between Georgia Tech and Fort Rucker (which involves working through an inter-exchange carrier such as Sprint or AT&T), there will still be a significant cost associated with the lease of a local T1 line between Georgia Tech and Fort Macpherson.

A cost effective method to overcome the necessity for a leased T1 line might be to use a "dial-up" T1. Bell South is planning intra-exchange 348 Kbaud dial-up T1 service in the Atlanta area some time in 1992. The per-minute charges for dial-up service are greater than those for a full-time leased T1 line, however the projected monthly usage for such a connection by Georgia Tech are very low. Therefore overall monthly charges could be minimized by paying only for long haul simulation time and drop the connection when it is not in use.

POSSIBLE NETWORKING SCHEMES

Several networking arrangements are possible, and in order to minimize cost, different ones may be set into place to accommodate specific limited needs as the FLIGHT SIM Laboratory develops. For example, during the initial stages of the FLIGHT SIM Laboratory development, emphasis will be placed on establishing the correct operation of the networking protocol and its interface to the FLIGHT SIM hardware. There will be little need for running long haul interactions with the SIMNET network during this stage. In fact, Georgia Tech's efforts to work out initial problems associated with bringing FLIGHT SIM on line would only interfere with normal SIMNET activities.

The following sections discuss the various levels of network connectivity required during the various stages of the FLIGHT SIM Laboratory development.

GEORGIA TECH TO FORT RUCKER

Presently, if Georgia Tech were able to obtain leased or dial-up T1 access to Fort Rucker, the FLIGHT SIM Laboratory could be directly interconnected to the SIMNET network. To date, the standard protocol for communication over the SIMNET network has been SIMNET version 6. FLIGHT SIM Laboratory computer equipment would have to conform to this standard. Also Georgia Tech would have to purchase a BBN Butterfly and matching CSU/DSU to allow compatibility.

These steps would likely be premature for two reasons. First, the need to test software and hardware interfaces early in the development of the FLIGHT SIM Laboratory will require frequent on-line experiments. However such on-line real-time experimentation will be incompatible with normal ongoing SIMNET operations. Second, the likelihood of adopting a new SIMNET standard is fairly high. The new standard (under consideration) is known as "Distributed Interaction Simulation" (DIS) and is a superset of the current SIMNET version 6. Therefore, gearing FLIGHT SIM Laboratory software designs on the SIMNET version 6 standard could result in rapid code obsolescence. Presently, the University of Central Florida is using DIS for their simulators. Unlike DIS, the BBN source codes for SIMNET version 6 are not as well documented, thereby increasing development risk to Georgia Tech (recognizing that all SIMNET installations currently networked have been established by BBN).

GEORGIA TECH TO GEORGIA TECH

Perhaps the best approach to developing the networking capability for the FLIGHT SIM Laboratory will be to create a local Georgia Tech network simulating a long haul T1 link to Fort Rucker. Initially this could be implemented by using the existing ethernet networks located in the School of Aerospace Engineering and those at the Georgia Tech Research Institute's Aerospace Laboratory located approximately 15 miles to the north of the Georgia Tech campus. Both facilities are connected by a leased T1 line.

For initial developmental testing, any protocol common to the Georgia Tech network would be suitable for checking out FLIGHT SIM Laboratory driver routines. TCP/IP is the information protocol standard in place on the Georgia Tech network. Similarly, existing CSU/DSU equipment could be used to support SIMNET-like communications between the FLIGHT SIM Laboratory and the simulated SIMNET. A simple SAFOR simulation could be run on one of the many computers networked at the Georgia Tech Research Institute Aerospace Laboratory facility in order to exercise the network communications capability of the FLIGHT SIM Laboratory while providing a dynamically changing object in an otherwise stationary simulated environment.

The use of the Georgia Tech networking infrastructure will permit realistic networking experiments to take place during initial development of the FLIGHT SIM Laboratory without addition cost to the project, and without having to commit to specific networking hardware prematurely.

Georgia Tech to University of Central Florida

Once the FLIGHT SIM Laboratory is on-line physically, it should be exercised against more complex simulated environments to determine the robustness of the helicopter simulation and its conformity to a networking standard. Since the DIS standard is likely to be adopted for universal SIMNET use, it should be incorporated into the FLIGHT SIM Laboratory from the beginning. Currently, the only facility using the DIS standard is the University of Central Florida (UCF). Since the FLIGHT SIM Laboratory would still be in a developmental stage, direct connection to Fort Rucker and the actual SIMNET network would not yet be desirable, however connection to UCF through a dial-up T1 line would be useful.

The University of Central Florida has been experimenting the lower-cost alternatives to the BBN Butterfly such as the RAD bridge/routers. In order to connect the FLIGHT SIM Laboratory simulation at Georgia Tech to the simulator at UCF, both DIS protocol and RAD-type hardware interconnections would have to be implemented on the Georgia Tech end.

Georgia Tech to Fort Rucker *and* UCF

Ultimately the FLIGHT SIM Laboratory would need to be tied into the Fort Rucker facilities, and hence the actual SIMNET network. If during the second stage of FLIGHT SIM Laboratory development, Georgia Tech conforms to UCF standards, a determination will have to be made concerning the present degree of Georgia Tech hardware and protocol compatibility with Fort Rucker. Conceivably Fort Rucker may have adopted the DIS standard by that time, however if that is not the case, UCF has developed a SIMENT translator that allows SIMNET version 6 simulators to communicate with DIS simulators. This translator should be installed at the Fort Rucker facility rather than Georgia Tech to allow communication between both Georgia Tech and Fort Rucker while maintaining communication with UCF as well.

Also, if there are incompatibilities between the BBN Butterfly and lower-cost high-performance equipment such as the

RAD bridge/routers and CSU/DSU units, then Georgia Tech should supply Fort Rucker with a complete set of compatible interface equipment. This permit operation between Georgia Tech and Fort Rucker while maintaining communication with UCF as well.

RECOMMENDED NETWORKING STRATEGY

Assuming that T1 connections can be obtained between Georgia Tech and other present or future SIMNET nodes, the following implementation plan is recommended.

STAGE 1: GEORGIA TECH TO GEORGIA TECH USING DIS

Use the existing Georgia Tech network between the campus and the Georgia Tech Research Institute's Cobb County facility to simulate a long haul network as shown in the *Stage 1* figure. Use the anticipated DIS standard from the onset. Create a simple SAFOR target simulator on one of the presently networked computers located within the Georgia Tech Research Institute's Aerospace Laboratory in order to exercise the FLIGHT SIM Laboratory equipment during its initial development.

STAGE 2: GEORGIA TECH TO GEORGIA TECH AND UCF USING DIS

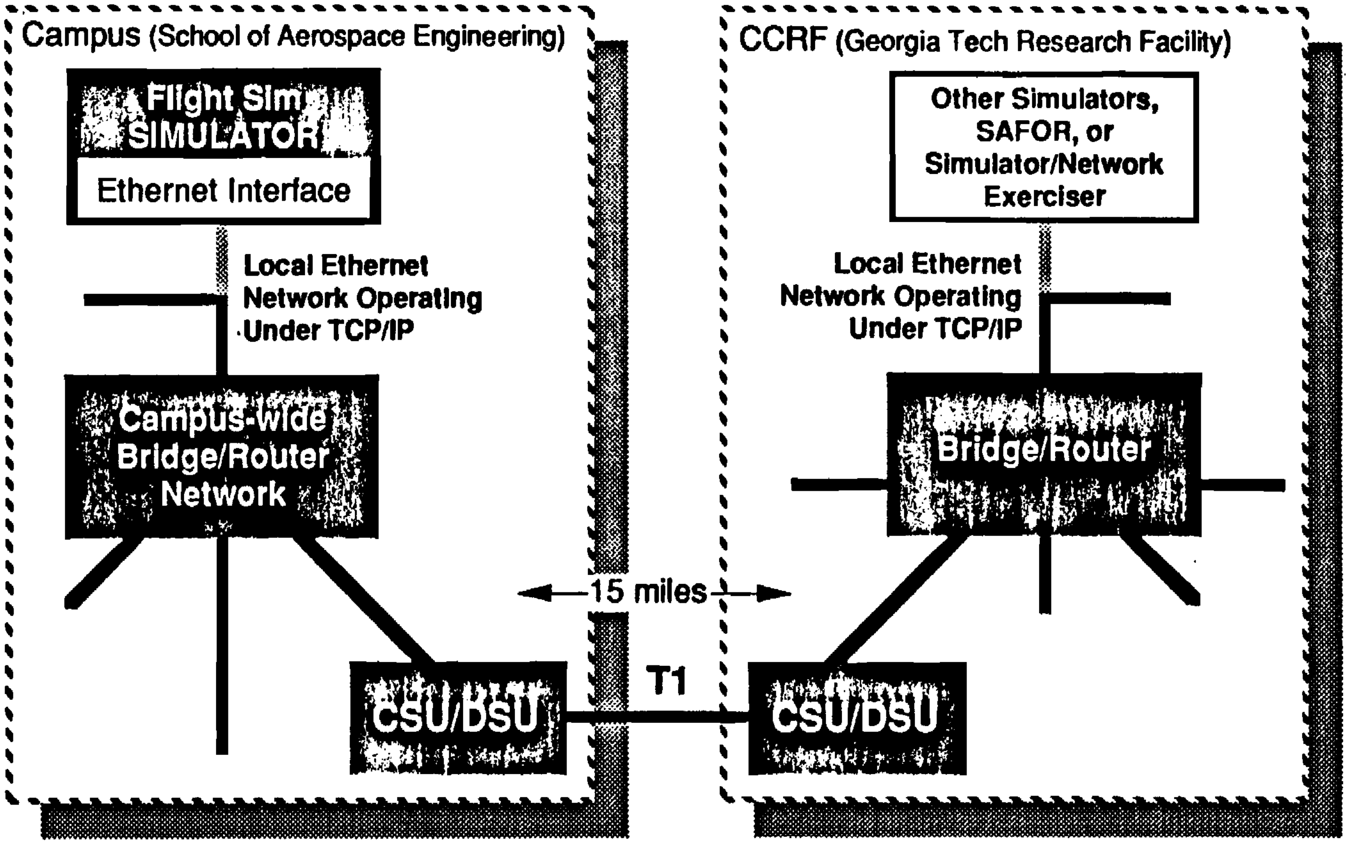
Purchase a low-cost high-performance bridge/router and CSU/DSU that is compatible with the UCF networking equipment. Establish a network using a dial-up T1 line as shown in the *Stage 2* figure. Establish correct operation of the Georgia Tech-implemented version of the DIS protocol. Test the robustness of the simulation in the presence of a richer environment.

STAGE 3: GEORGIA TECH TO GEORGIA TECH AND UCF USING DIS, AND SIMNET TRANSLATOR TO FORT RUCKER

Maintain networking options with UCF. Provide Fort Rucker with low-cost high-performance bridge/router and CSU/DSU units to allow compatibility between Georgia Tech and Fort Rucker's ethernet standards. This networking arrangement is depicted in the *Stage 3* figure. In the event that Fort Rucker is not using DIS, provide "DIS/SIMNET version 6" software translator. Thus configured, the Georgia Tech FLIGHT SIM Laboratory will have full bidirectional access to the entire SIMNET network as well as the UCF simulators.

STAGE 1
Simulated Long Haul Verification

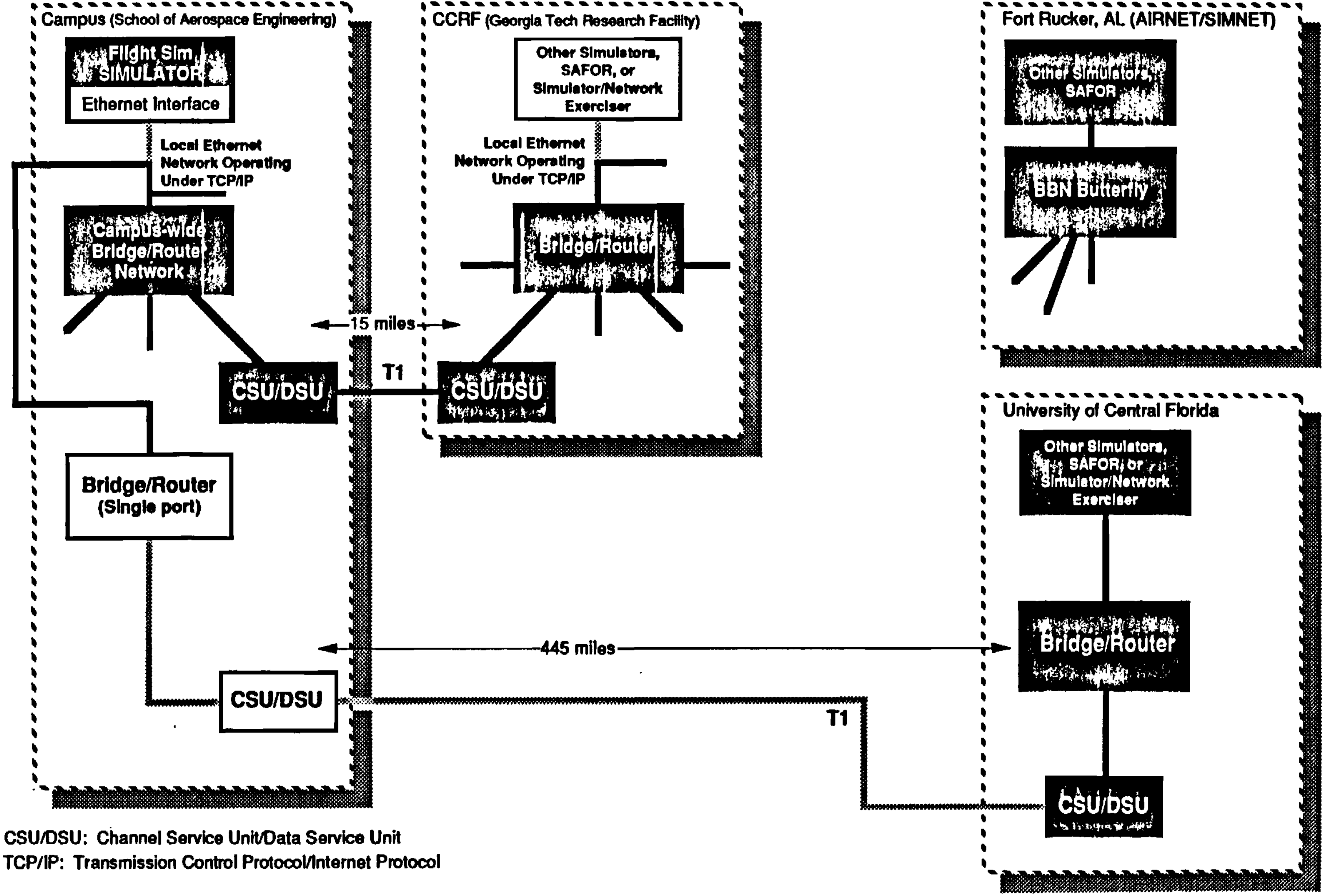
(Black boxes represent systems currently in place or soon to be in place.)


Campus (School of Aerospace Engineering)
Flight Sim SIMULATOR
Ethernet Interface
Local Ethernet Network Operating Under TCP/IP
Campus-wide Bridge/Router Network
CSU/DSU
15 miles
T1
CCRF (Georgia Tech Research Facility)
Other Simulators, SAFOR, or Simulator/Network Exerciser
Local Ethernet Network Operating Under TCP/IP
Bridge/Router
CSU/DSU

CSU/DSU: Channel Service Unit/Data Service Unit
TCP/IP: Transmission Control Protocol/Internet Protocol

STAGE 2 Long Haul DIS Interaction

(Black boxes represent systems currently in place or soon to be in place.)

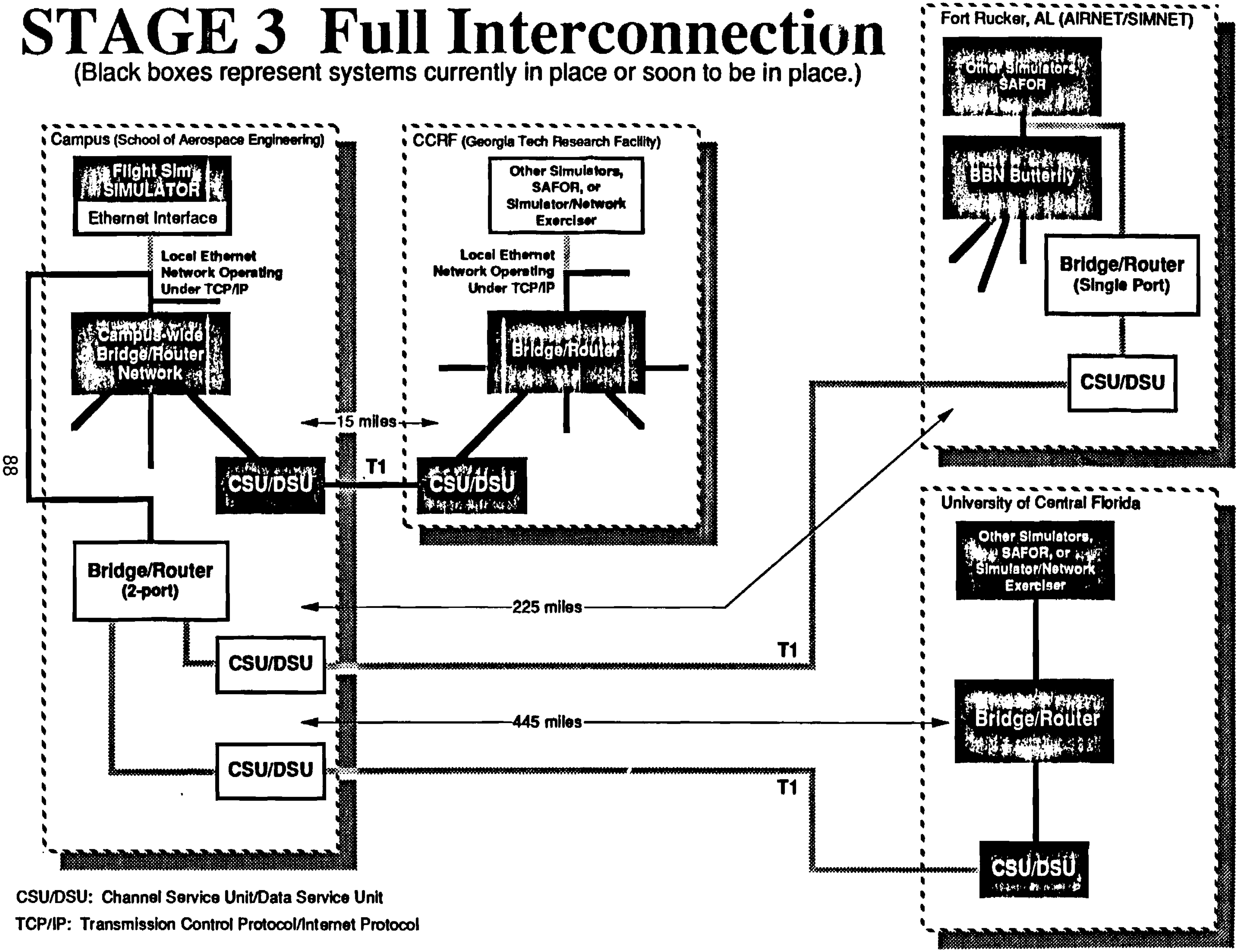


Campus (School of Aerospace Engineering)
Flight Sim SIMULATOR
Ethernet Interface
Local Ethernet Network Operating Under TCP/IP
Campus-wide Bridge/Router Network
CSU/DSU
Bridge/Router (Single port)
CSU/DSU
15 miles
T1
CCRF (Georgia Tech Research Facility)
Other Simulators, SAFOR, or Simulator/Network Exerciser
Local Ethernet Network Operating Under TCP/IP
Bridge/Router
CSU/DSU
Fort Rucker, AL (AIRNET/SIMNET)
Other Simulators, SAFOR
BBN Butterfly
University of Central Florida
Other Simulators, SAFOR, or Simulator/Network Exerciser
Bridge/Router
CSU/DSU
445 miles
T1

CSU/DSU: Channel Service Unit/Data Service Unit
TCP/IP: Transmission Control Protocol/Internet Protocol

STAGE 3 Full Interconnection

(Black boxes represent systems currently in place or soon to be in place.)



Campus (School of Aerospace Engineering)

Flight Sim SIMULATOR

Ethernet Interface

Local Ethernet Network Operating Under TCP/IP

Campus-wide Bridge/Router Network

CSU/DSU

Bridge/Router (2-port)

CSU/DSU

CSU/DSU

CCRF (Georgia Tech Research Facility)

Other Simulators, SAFOR, or Simulator/Network Exerciser

Local Ethernet Network Operating Under TCP/IP

Bridge/Router

CSU/DSU

15 miles

T1

225 miles

T1

445 miles

T1

Fort Rucker, AL (AIRNET/SIMNET)

Other Simulators, SAFOR

BBN Butterfly

Bridge/Router (Single Port)

CSU/DSU

University of Central Florida

Other Simulators, SAFOR, or Simulator/Network Exerciser

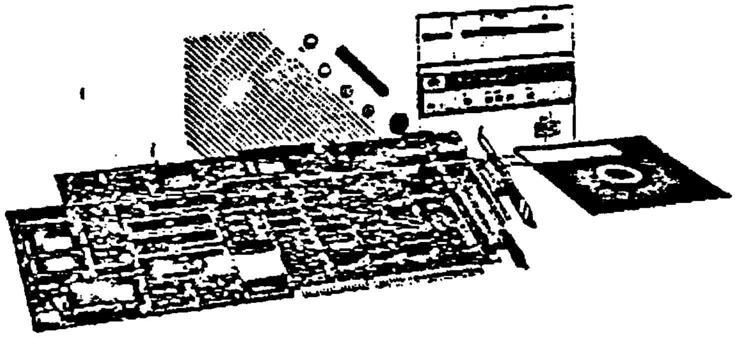
Bridge/Router

CSU/DSU

CSU/DSU: Channel Service Unit/Data Service Unit

TCP/IP: Transmission Control Protocol/Internet Protocol

APPENDIX RAD REB-SX AND RAD FCD1 DATA SHEETS

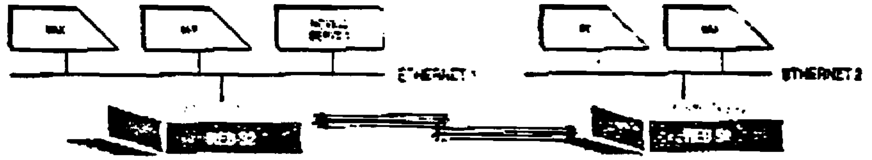


REB-SX

REMOTE ETHERNET BRIDGE/ROUTER

FEATURES

- **Combined transparent bridge and smart router**
  - Transparent routing, any protocol
  - Intelligent traffic load distribution
  - Self-learning and auto-configuring
  - End-to-end data integrity
  - Multiple WAN serial link connections, data rates from 4.8 kbps up to 2.048 Mbps.
- **Dynamic Shortest Path First (SPF) routing**
  The most advanced routing algorithm available today. Each bridge/router continuously learns all available paths, checks the network, and automatically determines the best path to send packets.
- **Triangulation and path control**
  Full support of multiple paths between LANs. Redundant, passive links turned into active, valid links for traffic load distribution. User controllable routing on a per-application basis.
- **Redundancy and instantaneous alternate routing**
  Build redundancy into the network. Continuous data flow through split-second switchover to an alternate route.
- **Load balancing**
  Load balancing can be used with redundant parallel serial links. The bridge/router monitors the load of the links and distributes data packets as necessary to achieve maximum utilization of the combined bandwidth.
- **Access control**
  Multi-level blocking of unauthorized internetwork communications. Protects valuable resources, keeps traffic, like specific local broadcast messages, off costly WAN serial links.
- **Diagnostics**
  Diagnostics automatic at start-up and through user request. Auto-fault detection and recovery.



DESCRIPTION

The REB-SX Remote Ethernet Bridge/Router is a high performance, reliable bridge/router capable of interconnecting geographically dispersed Ethernets or IEEE 802.3 local area networks. Network interconnections are provided by REB-SX-to-REB-SX serial communication links at data rates up to 2.048 Mbps.

The REB-SX bridge/router features transparent routing based on a dynamic Shortest Path First (SPF) algorithm. SPF is considered the most advanced routing method available today and one of the proposed routing standards. Each REB-SX maintains an updated routing table indicating the shortest path and the best alternate path to any LAN in the network. The REB-SX continuously checks the network paths and responds to events with split-second switchover to an alternate path.

The REB-SX bridge/router operates transparently to any higher layer protocol including Transmission Control Protocol/ Internet Protocol (TCP/IP), DECnet™, Xerox™ Network Services (XNS), IPX™, OSI protocols and others. Moreover, the REB-SX bridge/router simultaneously works with different operating systems and network types, for example Novell's Netware™, 3COM's 3+™, UB's NetOne™, Banyan's Vines™ and others.

RAD

RAD Network Devices

REB-SX single and dual link bridge/routers provide a complete solution to interconnect two or three LANs. They can be combined with other RND Ethernet bridge/routers in any network, including those with closed loop and star topologies, and built-in redundancy. Multiple paths between LANs are actively used during normal operation, allowing efficient distribution of traffic. The network administrator has the option of assigning costs to each link and values to routing variables, enabling user control over the direction of routing, if required.

Applications are guaranteed bandwidth, without compromising alternate routing. Use redundant parallel links more effectively through load balancing to maximize the combined available bandwidth. Packet order is guaranteed.

Powerful and flexible, the REB-SXs can be used in both simple and complex networks in almost any combination. They immediately recognize all nodes in their own LAN segments and within the larger, overall network. They respond immediately to network changes such as new or relocated nodes.

REB-SX's bilateral access control mechanism ensures security, and bandwidth savings by establishing user groups and stopping unauthorized internetwork communications. REB-SX enables or disables internetwork communications between any two LAN addresses. Up to 16 mutually exclusive user groups can be established as well as hundreds of subgroups with limited access rights.

In addition, with access control it is possible to stop specific local traffic from entering the WAN and save costly bandwidth for important data.

RND's RIM Remote Internetwork Management station is available to monitor and control the LAN/WAN network including viewing statistics, changing operating parameters, troubleshooting and other activities.

Two REB-SX models are available: the differences are in the number and types of their communication link interfaces. This ensures maximum flexibility in network design. The REB-SX uses synchronous, full-duplex communication and works with T1 or M1-CEPT, coax, twisted pairs, fiber optic, microwave, satellite, modem or digital phone network links.

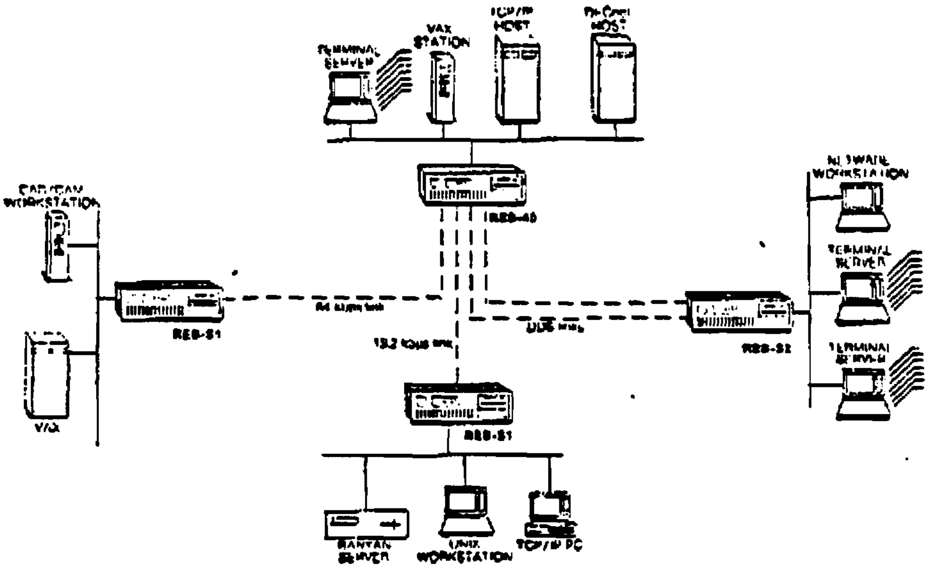
This document contains trademarks registered by their respective companies.
Specifications subject to change without notice.

SPECIFICATIONS

- **Models**
  REB-S1 Single link bridge/router
  REB-S2 Dual link bridge/router
- **LAN Interfaces**
  One Ethernet/IEEE 802.3 Interface
- **WAN Serial Links**
  - One or two serial links
  - RS-422 (449)/V.11 (X.21) or V.35 standard interfaces
  - Data rate from 4.8 kbps up to 2.048 Mbps (aggregate for two links).
  - External or Internal clocking
  - SDLC based link protocol
- **Filter, Forward Rates**

Packet filter rate	10,000 per second
Packet forward rate	2,000 per second

- **Bridge Buffers**
  256KB
- **Indicators**
  LEDs on front panel indicate proper operation, connectivity, Ethernet activity and dynamic load
- **Dimensions**
  PC-AT based
  Two full sized add-on cards
  Card Height 12.2 cm/4.8 in
  Card Depth 33.3 cm/13.1 in
- **Environment**
  Temperature 10° to 40° C (50° to 104° F)
  Humidity Up to 90% (non-condensing)
- **Software**
  REB-SX program supplied on a 5 1/4" diskette



ETHERNET WIDE-AREA NETWORK

• Intelligent routing • High performance • Access control

RAD Network Devices

U.S. Headquarters
7711 Center Avenue
Huntington Beach, CA 92647
Tel: (714) 891-1964
Fax: (714) 891-7788

International Headquarters
Atidim Technological Park, Bldg. 1
Tel Aviv 61131, Israel
Tel: 972-3-5447361
Fax: 972-3-5447368

RAD

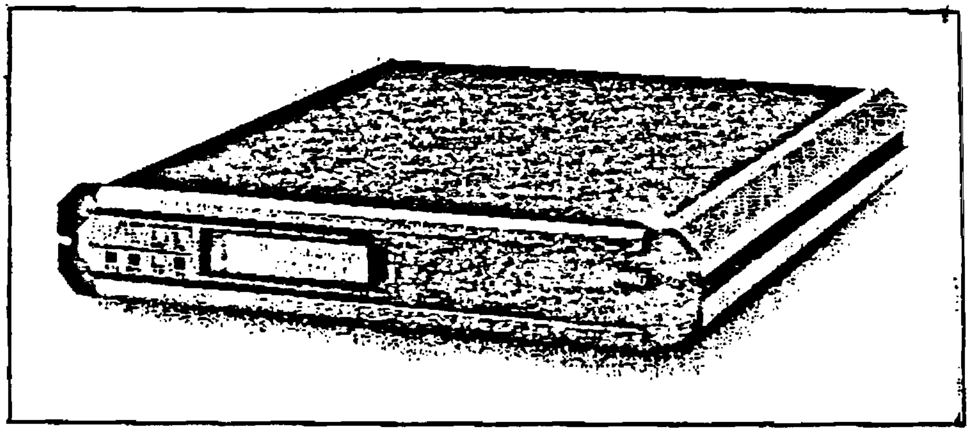
FCD-1

Fractional T1 CSU/DSU

FEATURES

- Fractional T1 CSU/DSU
- Two models: basic (FCD-1M) and advanced (FCD-1X)
- Selectable Framing Format: ESF or D4
- Selectable "ones density" control: Transparent, B7ZS or B8ZS
- V.35, RS-530 or X.21 interface
- Selectable sync data rates: n times 56 or 64 kbps
- Multiple clock source selection for T1 and user ports
- Front panel set-up and control. Supervisory Port available on FCD-1X
- Storage of last 24 hour T1 line diagnostics information

DESCRIPTION



- The FCD-1 is a CSU/DSU for ractional T1 services. Without equiring a T1 multiplexer, a ynchronous data channel can be onnected over the public T1 etworks while paying only for the andwidth required. Two models are vailable: The basic model, FCD-1M, nd a more advanced one, FCD-1X.

- Data is packed into timeslots (DS0s) in one of two selectable methods: either bundled into consecutive timeslots or placed in alternate timeslots. The FCD-1X provides additional flexibility, giving the user full control over timeslot allocation without restrictions.

- Both models provide a single user interface with a V.35 interface. The data rate can be programmed to operate at any multiple of 56 or 64 kbps. The FCD-1X also provides an RS-530 interface, supporting RS-449/422 and X.21 using an interface adaptor supplied with the unit.

- The FCD-1 is compatible with virtually all carrier-provided T1 services. It supports both D4 and ESF framed formats. Zero suppression over the line is selectable for either Transparent, B7ZS or B8ZS. All parameters meet AT&T PUB 62411 requirements, and the integral CSU guarantees a range of up to 1 mile.

- Multiple clock source selection provides maximum flexibility in connecting both the T1 and user interfaces. The T1 line may be clocked from the recovered Receive Clock, or from an internal oscillator. The user interface may be set to DTE or DCE with external transmit clock and is clocked accordingly. In the FCD-1X, the user interface also supports connection to DCE where both receive and transmit clocks are inputs to the FCD-1X. Using this clocking mode the T1 link may also be locked to the clocks from user interface.

- Both models provide set-up and control from front panel LCD and push buttons. The FCD-1X also features a Supervisory Port for connecting a terminal or PC. Status and diagnostic information can also be obtained either via the front panel LCD or the Supervisory Port.

- Maintenance capabilities include local and remote loopbacks at various points for rapid location of faults. When operating in the ESF format, T1 line statistics are stored in memory in compliance with both the ANSI (both models) and the AT&T (FCD-1X only) versions. This information may be retrieved by the service supplier, or locally through the supervisory port of the FCD-1X.

SPECIFICATIONS

T1 DATA LINK

- **Framing**

  D4, ESF
- **Bit Rate**

  1.544 Mbps (+/- 130 ppm)
- **Line Code**

  AMI
- **Zero Suppression**

  Transparent, B7ZS or B8ZS
- **Impedance**

  100 Ohms, balanced
- **Signal Levels**

  —Receive : 0 to -34 dB
  —Transmit:
  1) 0 dB, -7.5dB, -15dB
  2) 3V, 10% soft adjustable to be at 0 to 655 feet
- **Jitter Performance**

  As per AT&T PUB-62411
- **Connector**

  D-type 15-pin, female
- **Transmit Timing, Soft Selectable**

  —Internal (+/- 32 ppm)
  —Receive Timing (+/- 130 ppm)
  —External Timing, DCE source (+/- 100 ppm, FCD-1X only)

DATA CHANNEL

- **Interface**

  V.35

  FCD-1X also supports RS-530. RS-449/422 and X.21 interfaces are supported using an interface adaptor cable connected to the RS-530 connector
- **Bit Rate**

  n x 56 kbps or n x 64 kbps, where n equals 1, 2, 3, ... 24
- **Clock Modes**

  —Receive and Transmit Clock to the synchronous DTE
  —Receive clock to the synchronous device and transmit clock from the synchronous device
  —Receive and Transmit clock from the synchronous DCE (FCD-1X only)
- **Control Signal**

  —CTS follows RTS or constantly ON, soft selectable
  —DSR constantly ON, unless in test mode
  —DCD constantly ON, unless in RED ALARM
- **Time-slot Allocation**

  —Sequential (Bundled)
  —Alternate
  —User defined (FCD-1X only)
- **Diagnostics**

  —DS-1 local analog loopback, towards local DTE
  —Channel loopback towards remote DTE
  —BERT through remote FCD-1 unit
- **Statistics**

  —Full statistics diagnostic capabilities according to ANSI standard T1.403-1989
  —Local support of ESF diagnostic according to AT&T PUB 54016 (FCD-1X only)
- **Supervisory Port (FCD-1X only)**

  —Interface: RS-232
  —Connector: RJ-45
  —Speed: 9600, 4800, 2400, 1200, 300 bps. Autobaud supported
  —Character: 8 bit no parity, 7 bit even or odd parity
- **Front Panel Control (both models)**

  —Liquid Crystal Display: 2 rows of 16 characters
  —3 push buttons: Cursor, Scroll, Enter
- **Indicators**

  —RED ALARM
  —YELLOW ALARM
  —TEST
  —TXD
  —RXD
  —DCD
  —RTS
- **Alarms**

  —DS-1 loss of signal
  —BVP error
  —DS-1 driver failure
  —DS-1 frame slip
  —Invalid clock source
- **Summary of features found only in the FCD-1X model:**

  1. RS-530; RS-449/422; X.21 Interfaces
  2. Supervisory Port for Control and Diagnostics
  3. AT&T PUB 54016 support of ESF Diagnostics (Local)
  4. Serve as a DTE. Both Receive and Transmit Clocks into the FCD-1X
  5. Lock of the T1 Transmit Clock to the clock from the user Data Channel
  6. Free user definition of the time slot allocation
- **Physical**

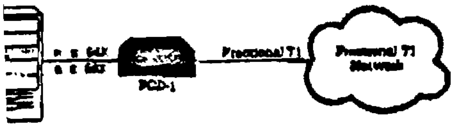
Depth:	12.0 in	(305 mm)	
Width:	10.5 in	(267 mm)	
Height:	1.7 in	(43.2 mm)	(1 U)
Weight:	2.9 lb	(1.3 kg)	

- **Environment**

  Temperature: 32-122°F (0-50°C)
  Humidity: 0% to 90% non-condensing
- **Power**

  115 VAC +/- 10%, 15VA
  47 to 63 Hz

APPLICATION



ORDERING

FCD-1M

Basic Fractional T1 CSU/DSU

FCD-1X

Advanced Fractional T1 CSU/DSC

RM-3

Hardware for mounting one or two stand-alone units onto a 19" rack

RAD

data communications ltd

U.S. HEADQUARTERS
RAD Data Communications Inc.
151 West Passaic Street
Rochelle Park, NJ 07662
Tel: (201) 587-8822
Telex: [illegible]
Fax: (201) 587-8847

INTERNATIONAL HEADQUARTERS
RAD Data Communications Ltd.
8 Hanechoshet Street
[illegible]
Tel: 972-3-[illegible]
[illegible]