(2)

IDA PAPER P-2329

THE IDA ADVANCED TECHNOLOGY COMBAT SIMULATION PROJECT

Peter S. Brooks
Dennis DeRiggi
Lowell Bruce Anderson
Cy D. Ardoin
Daniel J. Sehnal
Jeffrey B. Tate
Christopher Herrick

September 1990

"Original contains color plates: All DTIC reproductions will be in black and white"

INSTITUTE FOR DEFENSE ANALYSES
1801 N. Beauregard Street, Alexandria, Virginia 22311-1772

IDA Log No. HQ 90-36139

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	September 1990	Final

4. TITLE AND SUBTITLE

The IDA Advanced Technology Combat Simulation Project

5. FUNDING NUMBERS

IDA Central Research

6. AUTHOR(S)

Peter s. Brooks, Dennis DeRiggi, Lowell Bruce Anderson,
Cy D. Ardoin, Daniel J. Sehnal, Jeffrey B. Tate, Christopher Herrick

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Institute for Defense Analyses
1801 N. Beauregard Street
Alexandria, VA 22311

8. PERFORMING ORGANIZATION REPORT NUMBER

IDA Paper P-2329

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Institute for Defense Analyses
1801 N. Beauregard Street
Alexandria, VA 22311

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release, distribution unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT ***(Maximum 200 words)***

As computer speeds and capabilities increase, models which used to require hours or days to execute may now be executed in minutes. While the analyst soon may be able to contemplate doing hundreds of runs in a day, the lack of suitable techniques for analyzing the vast output produced by this many runs is evident. One objective of the IDA Advanced Technology Combat Simulation Project is to develop new techniques for analyzing such output.

The major product of this research project to date has been the development of a new methodology for conducting and analyzing the results of large scale parametric analyses based on computerized models. This response surface methodology comprises three components: the ability to perform hundreds of model excursions rapidly, an advanced graphical analysis system developed by IDA, and a collection of model specific input and output processors. This Graphical Analysis System incorporates advanced topological concepts and algorithms in a sophisticated application of Mathematica and provides the ability to visualize any multidimensional data set. Using the response surface methodology, the analyst can better perform model verification and validation, can quickly home in on the key model sensitivities and can more thoroughly convey the results of an analysis to a broad audience.

14. SUBJECT TERMS

Combat Simulation, Validation and Verification, Graphical Analysis, Visualization, Multidimensional Data, 3 Dimensional Displays, Animation, Parallel Processors, Computer Networks

15. NUMBER OF PAGES

120

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	SAR

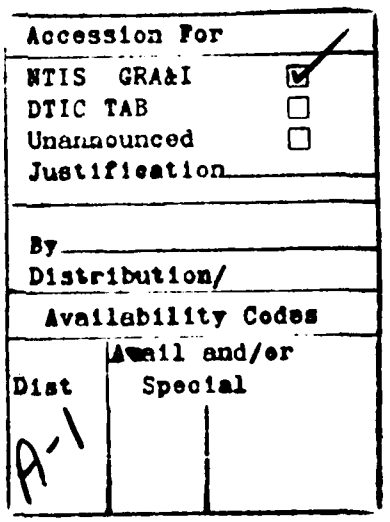
IDA PAPER P-2329

THE IDA ADVANCED TECHNOLOGY COMBAT SIMULATION PROJECT

Peter S. Brooks
Dennis DeRiggi
Lowell Bruce Anderson
Cy D. Ardoin
Daniel J. Sehnal
Jeffrey B. Tate
Christopher Herrick

September 1990

INSTITUTE FOR DEFENSE ANALYSES
IDA Central Research Program



PREFACE

This paper was prepared as part of IDA Project 9000-623 under the IDA Central Research Program. It describes a new methodology for the analysis of multidimensional data sets. Key applications include 1) the validation and verification of a model (in which many parametric variations are performed), 2) resource tradeoffs that include both cost and effectiveness measures of merit, 3) the comparative analysis and display of results derived from different scenarios, and 4) the analysis of collections of intelligence-based indicators expressed as time series. It also may be applied to other situations where variations in several factors are evaluated according to several measures of merit.

This project was made possible by the generous cooperation of several IDA components, including the Supercomputing Research Center, the Computer & Software Engineering Division, the Strategy, Forces & Resources Division and the Information Services Directorate.

The authors acknowledge the expert contributions of the following individuals, to whom much is owed: Robert Atwell, Charles Davisson, David Dierolf, G. Watts Hill, III, Jeffrey H. Grotte, Ken Ratkiewicz, Phillip Merkey, Paul B. Schneck, Eleanor L. Schwartz, Shawn Sheridan, William Stoltz, Victor Utgoff, Lowell Miller, Valyncia O. Lindsey and Diane Wright. Throughout the project, Eileen Doherty and Barbara Fealy provided administrative assistance and editorial support. The various presentations and other results produced by this study were enhanced by their collective efforts. The paper also benefitted from the careful reviews conducted by Eleanor Schwartz and David Dierolf.

ABSTRACT

As computer speeds and capabilities increase, models which once required hours or days to execute now may be executed in minutes. While the analyst soon may be able to contemplate doing hundreds of runs in a day, the lack of suitable techniques for analyzing the vast output produced by this many runs is evident. One objective of the IDA Advanced Technology Combat Simulation Project is to develop new techniques for analyzing such output.

The major product of this research project to date has been the development of a new methodology for conducting and analyzing the results of large scale parametric analyses based on computerized models. This response surface methodology comprises three components: the ability to perform hundreds of model excursions rapidly, an advanced graphical analysis system developed by IDA, and a collection of model specific input and output processors. The Graphical Analysis System incorporates advanced topological concepts and algorithms in a sophisticated application of Mathematica (a software product from Wolfram Research, Inc.) and provides the ability to visualize any multidimensional data set. Using the response surface methodology, the analyst can better perform model verification and validation, quickly home in on the key model sensitivities, and more thoroughly convey the results of an analysis to a broad audience.

The Graphical Analysis System allows the user to specify any cross-section of a collection of input variations and then examine how several output measures simultaneously varied over this set of input changes. To use this system, the user first executes a collection of runs of any type of model, not necessarily a combat simulation, where the cases are organized as a multidimensional lattice, or grid. Multiple output measures are then presented as response surfaces. Multiple surfaces may be shown in one plot, and each may be colored by another output measure. Contours also may be displayed. The plots may be animated over any input variation, or over time if it is represented in the model. A final technique of the Graphical Analysis System is the capability to determine those combinations of inputs variations that would achieve a user-specified value for some model output measure.

Because the response surface methodology is applicable to most models and to many types of analyses, a broad range of the defense analysis community (and possibly other analytic communities) may benefit from the use of these methods.

CONTENTS

PREFACE iii

ABSTRACT v

I. INTRODUCTION 1

II. APPROACH 3

A. Multidimensional Parametric Analyses 3

B. Prototype Architecture 3

C. A Graphical Analysis System 6

D. Large Scale Computations 6

III. A GRAPHICAL ANALYSIS SYSTEM 7

A. Purpose 7

B. General Applicability 7

C. Implementation 8

D. Features 8

E. A Demonstration Analysis 10

IV. COMPUTER OPTIONS FOR LARGE SCALE NUMERICAL EXPERIMENTS 21

A. Multiflow Trace 21

B. Encore Multimax 22

C. Powerful Desktop Computers 23

D. A Networked Collection of Workstations 23

E. Overall Comparisons 23

V. POTENTIAL APPLICATIONS 25

A. Overview 25

B. Validation and Verification 25

1. Analysis of Input Data 26

2. Sensitivity Analyses 26

3. Comparison With Externally Generated Measures 26

C. Economic Modeling Applications 27

D. Database Applications 27

E. Dynamic Systems 27

VI. RESEARCH ACTIVITIES 29
A. Improvements to the Graphical Analysis System 29
B. Automatic Detection of Key Sensitivities 29
C. Interpolation of Results for New Input Regions 30
D. Conclusion 30
REFERENCES R-1

TABLE

1. Comparisons of Several Computers 24

FIGURES

1. Architecture of System 5
2. Key Features of the Graphical Analysis System 9
3. Two Basic Model Calculations: Potential Ground Gained vs Breakthrough Threshold 13
4. Depiction of Non-breakthrough Strategies 15
5. Boundary Surface of Non-breakthrough Strategies 17
6. Maximum Ground Gained is Basically a Function of Fraction Forward Deployed .. 19
7. Potential Applications of the Graphical Analysis System 25

APPENDICES

A. DISTRIBUTED COMPUTING ON A LOCAL AREA NETWORK
B. THE APPLICATION OF GRAPHICAL ANALYSIS METHODS TO MODEL VALIDATION AND VERIFICATION: AN EXAMPLE
C. AUTOMATIC DETECTION OF KEY SENSITIVITIES

ANNEXES

B-1. GRAPHICAL REPRESENTATION OF SELECTED CEM OUTPUT
B-2. TABULAR REPRESENTATION OF SELECTED CEM OUTPUT

I. INTRODUCTION

The general problem faced by analysts who work with large scale combat models may be summarized as follows: Because both the models themselves and the problems they are designed to address have become quite complex, a thorough analysis may require sensitivity excursions on many inputs, examined individually and in combination. Two factors limit the number of runs that effectively can be performed during a study -- the time required to set up and execute a single run, and the fact that these models tend to produce large amounts of output. Too few cases might therefore be examined.

Simply using faster computers only exacerbates the overall problem. Models which once required hours or days to execute can now be executed in minutes. The analyst soon may be able to contemplate doing hundreds of runs in a day; however he or she must now contend with ever greater amounts of model output.

The objective of the Advanced Technology Combat Simulation Project is thus to develop new methods for the development and interpretation of large numbers of model runs. By joining advanced hardware, software and graphics technologies with advanced mathematical structures, better ways of more efficiently using large scale combat simulations can be realized. In particular, this project has developed one approach for using visual methods as a tool for analysis.

The central idea underlying these new visual methods of analysis is to display the model results in terms of response surfaces. Fast computers are used to execute a complete grid of cases. Each model output measure can then be plotted as a height above each grid point and the heights connected to form a surface. A Graphical Analysis System, developed under the IDA Central Research Program, allows the analyst to work with a multidimensional grid of cases and to display multiple surfaces (i.e., model output measures) at the same time. Furthermore, the Graphical Analysis System is applicable to any multidimensional data set, no matter how it was generated.

Our initial experience at IDA indicates there are several immediate benefits from using these methods. Presenting large amounts of data in terms of response surfaces

allows the analyst to examine the model's behavior more thoroughly. In doing so, subtle correlations may be observed and key sensitivities discovered. More importantly, anomalies of the model due to coding errors or logical inconsistencies may be revealed if the grid of cases represents a structured variation of the inputs and if multiple output measures are displayed at the same time.

There are other ongoing efforts within the military operations research community aimed at improving the overall capabilities of the models and the ease with which an analyst may use these models. The umbrella concept for these ideas, the Analyst's Workstation, may benefit from the incorporation of ideas similar to those developed by this project.

The remainder of this paper is organized as follows. The general approach of conducting large scale parametric analyses is outlined in Chapter II. There are two key components to this approach: a Graphical Analysis System (described in Chapter III) and the requirement for a computer capable of performing sufficiently many runs of a given application in a reasonable period of time. Chapter IV discusses several options investigated by this project. Chapter V provides an overview of the different types of analyses that may benefit from the use of the graphical analysis methods. In conclusion, Chapter VI discusses three research topics.

There are several technical appendices describing other topics undertaken during this project, including distributed computing on a local area network and the automatic detection of key model sensitivities. Also presented is an example application of the Graphical Analysis System to model verification and validation.

II. APPROACH

A. MULTIDIMENSIONAL PARAMETRIC ANALYSES

With the ability to execute rapidly many runs of a given simulation comes the opportunity to conduct structured analyses in which several inputs are varied independently. The focus here is on the exploration of a region of input values through the use of the notion of a grid of cases. The points of the grid may represent, for example, all possible combinations of variations in several inputs, where each input is varied at several levels independently. This is in contrast to the more common approach of conducting a small number of cases where, from case to case, many inputs are varied.

One advantage of this parametric approach is that the impact of each input's variation can be isolated with ease; in this grid structure, there are always cases that differ only in the values of a specific input. A second potential advantage is that one does not have to know precisely the values of certain inputs. If the values are known to lie within given intervals, then by conducting all parametric variations of these inputs and displaying the results as surfaces, the complete range of results may be observed.

B. PROTOTYPE ARCHITECTURE

This project demonstrated the approach of constructing and executing multidimensional parametric variations by joining several computing technologies in a unified architecture. The key elements of this prototype architecture are: 1) a workstation-type computer and a high resolution graphics terminal in the analyst's office, 2) a remotely located high-throughput computing capability (i.e., a parallel processor, a supercomputer or a coordinated network of processors) and 3) high speed data communications connections between the two sites. The system of computers shown in Figure 1 was made available to this project. It describes one realization of this architecture.

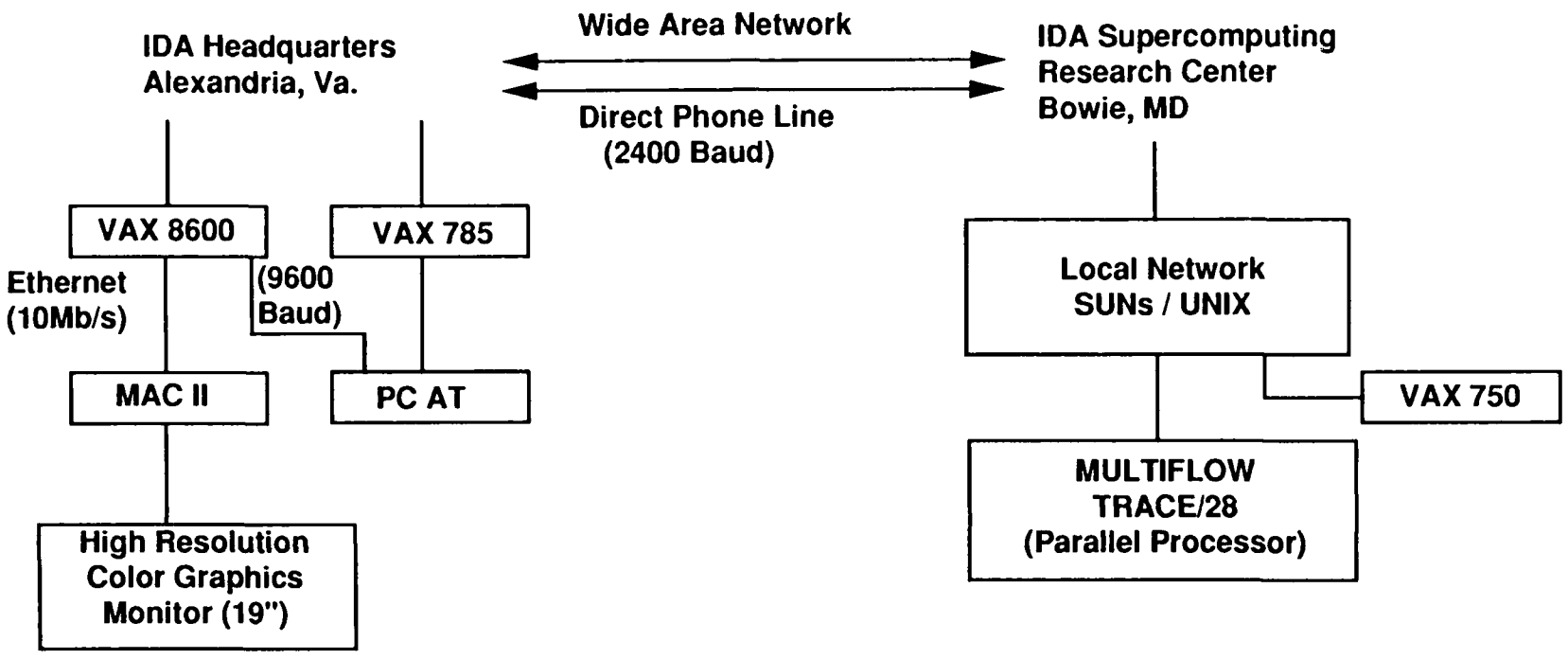
To use this collection of computers to conduct analyses based on a grid of cases, the following steps were performed.

First, the analyst determines which inputs or groups of inputs are to be varied during a particular analysis. To facilitate this determination, the main simulation and its input data are installed on the local computer. Here the analyst constructs the input variations which are to be joined in all possible combinations. Second, this small set of input variations is passed to the remote computing site. A utility is needed that creates all possible combinations of variations and creates the resulting input sets for the main simulation. The main simulation, also installed at this site, is then executed for all the cases in the grid.

Finally, the results of the simulation for all cases in the grid are collected and passed back to the home site. The Graphical Analysis System is then used to view the results in terms of response surfaces.

For the arrangement shown in Figure 1, a constraining factor was the data transfer rate available over the wide area network. A rate of one kilobyte per second, on average, was experienced. This mode of file transfer made use of a wide area network; at times the experienced rate was as low at .03 kilobytes per second. In subsequent phases of this project, another remote computing site was used. The home site was connected to this remote site by a T1 line which transferred data at an observed rate of 50 to 100 kilobytes per second.

While the widespread availability of fast data communication rates between remotely located sites will certainly increase in the future, the costs of the various options today are expensive. For communications over the wide area network, the only cost is for the system software (FTP) and for hardware to allow the connection of a gateway to the network. The data transfer rate of this file transfer method is affected mainly by the overall network load. If a dedicated line is used, there are at least two options. One may use a dedicated phone link, capable of a transfer rate of 56 kilobits per second (this would cost several thousands of dollars in hardware and annual leasing costs). Alternatively, one may use a (dedicated) T1 line, capable of a transfer rate of 1.5 megabits per second. The cost of this option is tens of thousands of dollars in hardware and annual leasing costs. IDA currently makes use of all three of the above methods for different applications.



MAIN SOFTWARE USED

Operating Systems:	VAX/VMS, UNIX, Mac OS, MS DOS
Communications:	FTP, PC/TCP, Smarterm 240
Model Calculations:	VAX FORTRAN, ANSI STD FORTRAN 77, Multiflow FORTRAN
Graphics:	Mathematica, PostScript

Figure 1. Architecture of System

C. A GRAPHICAL ANALYSIS SYSTEM

The presentation of a table of data as a surface has been until recently a laborious and expensive undertaking. What this project has shown is that there are now readily available software programs that make such depictions easy, rapid and inexpensive to create. This project has taken the basic capabilities of one such software program to solve the general problem of displaying arbitrarily chosen cross-sections of multidimensional data sets.

In particular, this new Graphical Analysis System identified and implemented the various standard ways that one might wish to present results of a collection of simulation runs. The most informative displays show, in one plot, comparisons across cases. For line plots, one plot shows multiple curves. For displays involving two dimensional grids of cases, one plot has surfaces. For grids involving three dimensional grids of cases, one objective may be to identify which cases yield equal values of some output measure. What is unique about this system is its ability to work easily with higher dimensional grids of cases, and an arbitrary number of output measures, as described below.

D. LARGE SCALE COMPUTATIONS

Large simulations normally produce tens of thousands of individual output numbers during each execution. The challenge of determining which are the key numbers is highlighted when one considers trying to analyze scores or hundreds of runs. If the runs are organized according to a structured variation of the input data set, then this task may in fact be quite tractable. The key point is that, in this situation, one focuses on the responsiveness of the output to the structured input variations. It may occur that for many input variations for a given simulation, a collection of 100 to 200 measures may suffice for answering the question: Did the input variations make a difference?

There remains the problem of which 100 or so measures to examine on a routine basis. When only large stacks of paper output are available, a limited set of output data is normally referenced; many output measures are ignored. With the capability to display the results graphically and rapidly, more output measures will likely be examined. It is the combination of performing more runs plus having the capability to analyze a greater number of output measures that will allow the analyst to address this problem.

III. A GRAPHICAL ANALYSIS SYSTEM

A. PURPOSE

The Graphical Analysis System is designed to provide the analyst a rapid, interactive capability to examine how multiple output measures respond to combinations of input variations. By displaying simulation results in terms of families of curves and surfaces, one can easily examine the results of many runs of the simulation, identify which of many output measures show important sensitivities, and determine behavior of the model that should be examined further. Not only can this new tool facilitate more complex analyses of multiple input variations, but it might also help one detect coding errors, logical anomalies or other aspects of the simulation that would diminish its correctness. This system also provides a new format for presenting the results of an analysis.

B. GENERAL APPLICABILITY

The Graphical Analysis System uses any multidimensional data set as its input. For example, such a data set may be generated from many runs of a given simulation, from only one run of one or more simulations, or even from a non-computer source (as discussed below).

The primary situation in which the Graphical Analysis System is applicable is when a given simulation is run many times and in which the cases may be organized as a multidimensional grid, or lattice. Here, each lattice point represents a particular combination of variations of several inputs. One is interested in how perhaps several output measures responded to the input variations represented by the grid. The multidimensionality is derived from 1) the dimensionality of the grid of input variations, plus 2) the multiple output measures that are recorded, plus 3) the time dimension, if it is represented in the simulation. If the input variations represent resource tradeoffs, then the costs associated with each case in the grid may be displayed in the same framework. This is because the cost associated with each case is simply another output measure, even though it may be derived from a separate calculation.

For a single run of a given simulation, the analysis may concern how a specific table of results, say a killer-victim scoreboard, varies over the time dimension of the simulation. One could represent the scoreboard values as a surface, and animate this surface over time. One also could display multiple surfaces in one plot, with each surface representing a different time sample. Multiple surfaces also could be used to represent the values of related calculations.

The data need not be generated by a computer simulation. In Chapter V, Potential Applications, databases with quantitative entries and intelligence-based indicators are offered as two such examples where the graphical methods may be of value.

C. IMPLEMENTATION

The Graphical Analysis System is implemented using the new software Mathematica, from Wolfram Research, Inc. Mathematica is a sophisticated, mathematically oriented package that contains as built-in commands many elements necessary to create complex graphical images. In addition, Mathematica provides a type of programming language, whereby the user can write programs that both perform calculations and display the results. The Graphical Analysis System is, from this perspective, a series of programs and algorithms written in the Mathematica language that allow the user to select and display various cross-sections of a multidimensional data set.

The Graphical Analysis System is now being used on a Macintosh II system. However, Mathematica is available on many computers, including the Sun, Compaq, DEC and other workstations. Since the built-in Mathematica commands as well as the programming structure operate in the same manner on all platforms, the Graphical Analysis System potentially can be used on these platforms as well.

D. FEATURES

There are two key advances represented by this system. First is the ability to display multiple surfaces on one plot, where each surface may be colored and on which there could be superimposed contour lines. The functions used to derive the height, color and contour lines for each surface may be distinct, independent functions. The second advance is the user interface, which provides the analyst a flexible mechanism for selecting which cross-sections of the multidimensional data set are to be displayed.

To use this system, the user first executes a collection of runs of any type of model, not necessarily a combat simulation, where the cases are organized as a multidimensional lattice, or grid. In general, any vector-valued (i.e., multiple output measures) function defined on a multidimensional grid of points may be used. The grid dimensions may be arbitrarily selected as the independent axes in line or surface plots and as the animation dimension for a series of plots. Output measures are then presented as response curves or surfaces. Multiple curves or surfaces may be shown in one plot, and each surface may be colored by another output measure. Contours also may be displayed. The plots may be animated over any input variation, or over time if it is represented in the model. A final technique of the Graphical Analysis System is the capability to determine those combinations of inputs variations that would achieve a user-specified value for some model output measure.

These features are summarized in Figure 2. In the following section, several of these capabilities will be demonstrated.

Arbitrary Choice of Axes and Animation Indices
One Independent Variable • **Multiple curves on each plot; color** • **Animation of plots**
Two Independent Variables • **Surfaces with color; contour curves** • **Multiple surfaces on each plot** • **Animation of plots**
Three Independent Variables • **Surfaces defined by where a given measure is constant** • **Resulting surface can be colored by a second measure** • **Contour curves of third measure can be displayed on surface**
Arbitrary resizing of plots and multiple windows allow several displays to be viewed simultaneously
Direct output to laser printer

Figure 2. Key Features of the Graphical Analysis System

E. A DEMONSTRATION ANALYSIS

As a demonstration of this research, the Graphical Analysis System was used to illuminate two aspects of the Force to Space Ratio (FSR) model, which calculates the optimal force employment strategy for a given force density. First, while the FSR model in fact searches over all possible employment strategies, it was observed that the optimum lies on a particular surface. This might lead in the future to a faster search algorithm. Second, the optimal strategy for Blue was shown to be quite robust. Figures 3, 4, 5 and 6 will be used to discuss these findings in more detail.

The Force to Space Ratio model was recently developed by IDA to investigate how force employments in Central Europe might change in response to decreases in force densities. Only a brief description is given here. More detail can be found in [5].

In this model, four input parameters describe any given Blue force employment strategy: 1) Fraction of Forces Forward Deployed (the remainder held in reserve), 2) Number of Defensive Lines (within which the forward deployed forces are arrayed), 3) Fraction of Reserve Forces used for Counterattack (the remainder are used to replace losses) and 4) Withdrawal Fraction (one minus this fraction is the threshold level of loss that Blue is willing to incur before he withdraws from that defensive line and establishes another line further to the rear).

One parameter is used to describe Red's force employment strategy: Velocity of Attack.

The objective for Blue is to keep Red from completely breaking through his defensive lines. Among all strategies that achieve this, the goal is to choose the strategy for which Red's maximum penetration is at a minimum. A Fortran program was created to perform the calculations. This program performs a complete enumeration of all strategies for Blue and Red and determines the optimal strategy, in the min-max sense. The Graphical Analysis System was used to illuminate certain aspects of this optimal strategy.

Figure 3 indicates that Blue's strategies are divided into two regions -- breakthrough versus non-breakthrough strategies. The two basic calculations made by the FSR model for each Blue force employment strategy are shown as surfaces. Because Blue wants to avoid Red's breaking through his defenses, Blue considers only those strategies (combinations of Fraction Forward Deployed and Number of Defensive Lines; the other

two Blue parameters are not varied in this figure) for which the Breakthrough Threshold is above the Potential Ground Gained for Red. Of these, Blue chooses the strategy yielding the least penetration distance.

Figure 4 indicates how the region of non-breakthrough strategies changes as the Withdrawal Fraction is varied from .9 to .1. Each of the curved surfaces represents the 'Potential Ground Gained for Red' calculation for the specified value of the Withdrawal Fraction (cf. red surface in Figure 3). Now, however, a coloring function is used to indicate which points represent non-breakthrough strategies for Blue (blue coloring) and which points result in breakthrough strategies (red coloring). The change from blue to red as one goes from the top surface to the bottom surface indicates that a surface separates the non-breakthrough from the breakthrough strategies, i.e., the blue region from the red region. The Graphical Analysis System explicitly calculates and displays this surface.

When this third Blue force employment parameter, Withdrawal Fraction, is allowed to vary, there is a surface that separates breakthrough from non-breakthrough strategies in the strategy space (the cube in Figure 5). The optimal strategy lies on this surface at the point where the Red ground gained is minimum. This surface can be colored by the 'Potential Ground Gained by Red' function. The figure shows, for example, that within the greenish band, the Red Ground Gained values vary between 118 and 122 kilometers, within the yellowish band they vary between 66 and 69, and within the broad orange band this measure varies between 42 and 49. The optimal value is 42. Blue therefore has wide latitude is selecting strategies without greatly affecting Red's potential ground gained.

As a final observation, Figure 5 indicates that the main variation of the 'Red Potential Ground Gained' values for those strategies on the boundary surface is in the Fraction Forward Deployed dimension. This is seen by the nearly constant color bands as one traverses the boundary surface along any path where the Fraction Forward Deployed value is constant. Thus, if the primary concern is to determine the least ground gained that Blue might be able to enforce upon Red, it is the behavior of the color function on this boundary surface that is the calculation of interest. This essential behavior for this surface is shown in Figure 6.

Figure 6 shows how the Red Ground Gained function varies as one moves along the boundary surface of Figure 5 for any fixed value of the Fraction Forward Deployed. The two black curves in Figure 6 bound the variation. Not only is the difference between

the curves small, but the interval about the min-max value (i.e., the lowest point on the lower curve) is relatively flat and somewhat broad. What is important, therefore, is perhaps not the min-max force employment strategies and the associated penetration distance for Red, but rather the value associated with the flat region of the curves in Figure 6. It may occur, although it has not been proven, that the underlying theory and model could provide a direct calculation of the curve (or curves) shown in Figure 6.

In summary, the Graphical Analysis System provided insights to the model's calculations that would have been difficult to obtain using traditional methods of looking at tabular results. Moreover, these insights were gained quickly, requiring only one day. Even the production of the figures used here can be done by the individual analyst in only minutes using commonly available software.

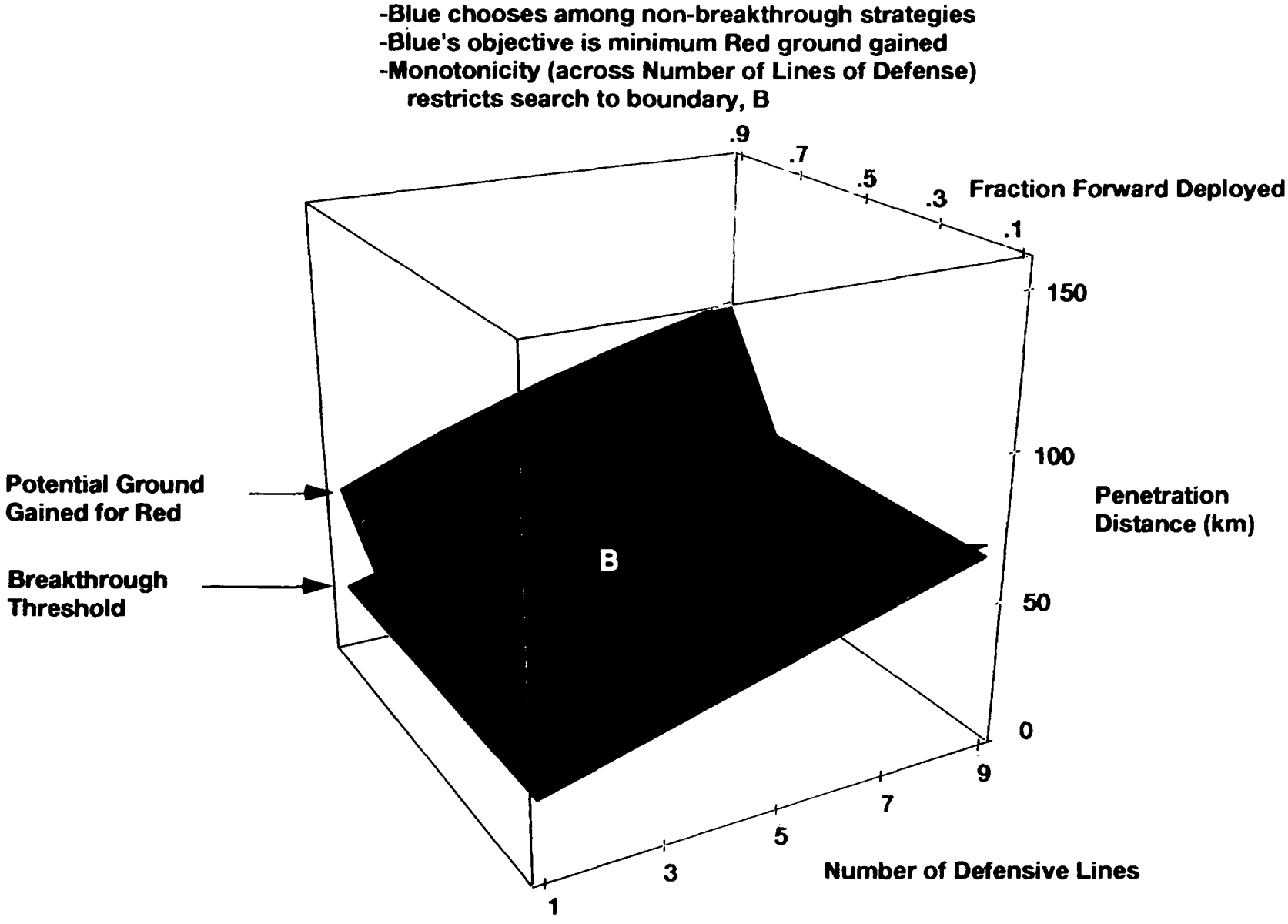


Figure 3. Two Basic Model Calculations: Potential Ground Gained vs Breakthrough Threshold

-Surfaces depict variation with respect to withdrawal fraction
  – Red = Strategy yields Red breakthrough
  – Blue = Strategy yields no Red breakthrough
-Blue wants minimum ground gained within blue region

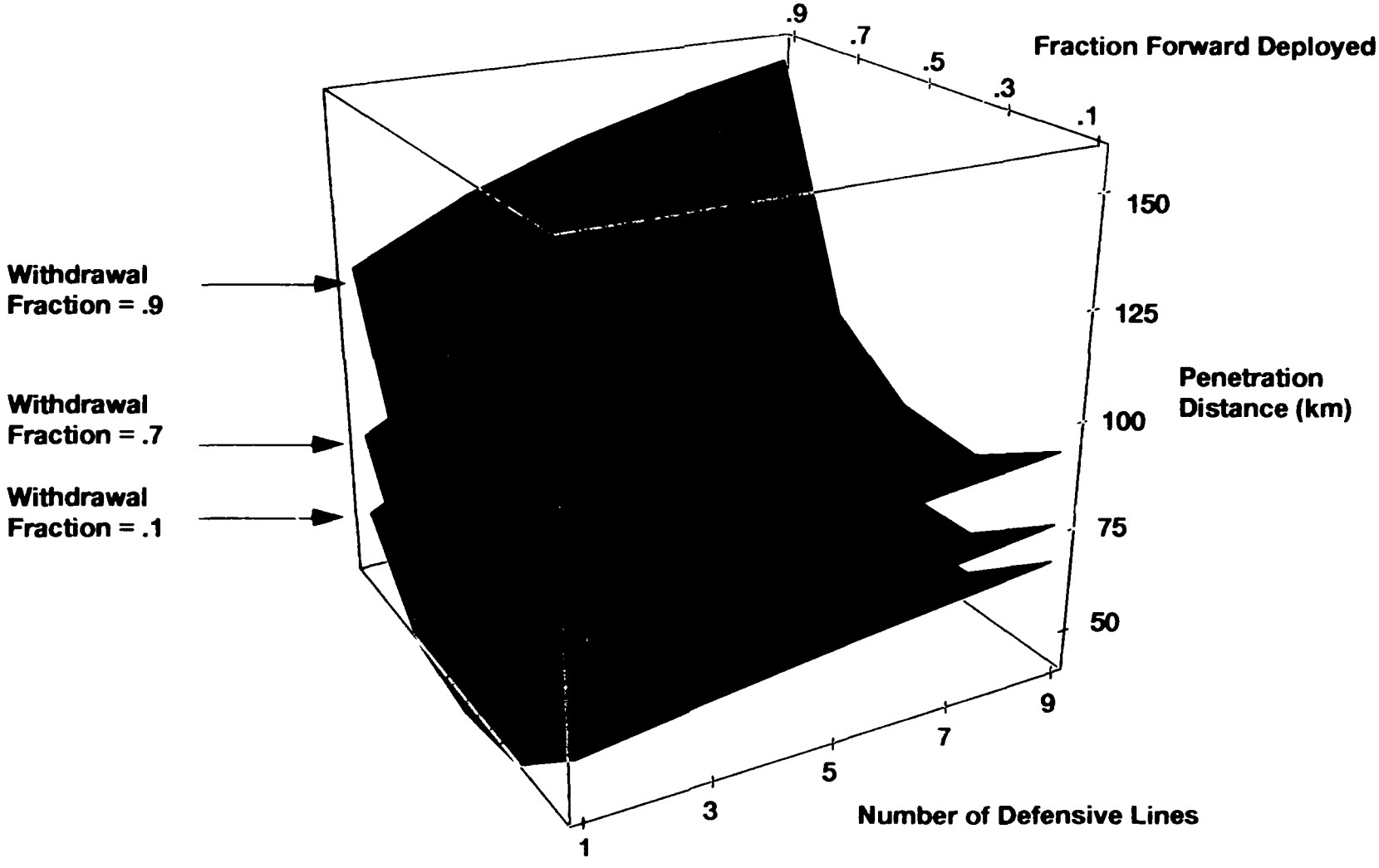


Figure 4. Depiction of Non-breakthrough Strategies

- Surface is colored by ground gained by Red
- Results most sensitive to Fraction Forward Deployed
- 118-122 shows range of ground gained within color bands

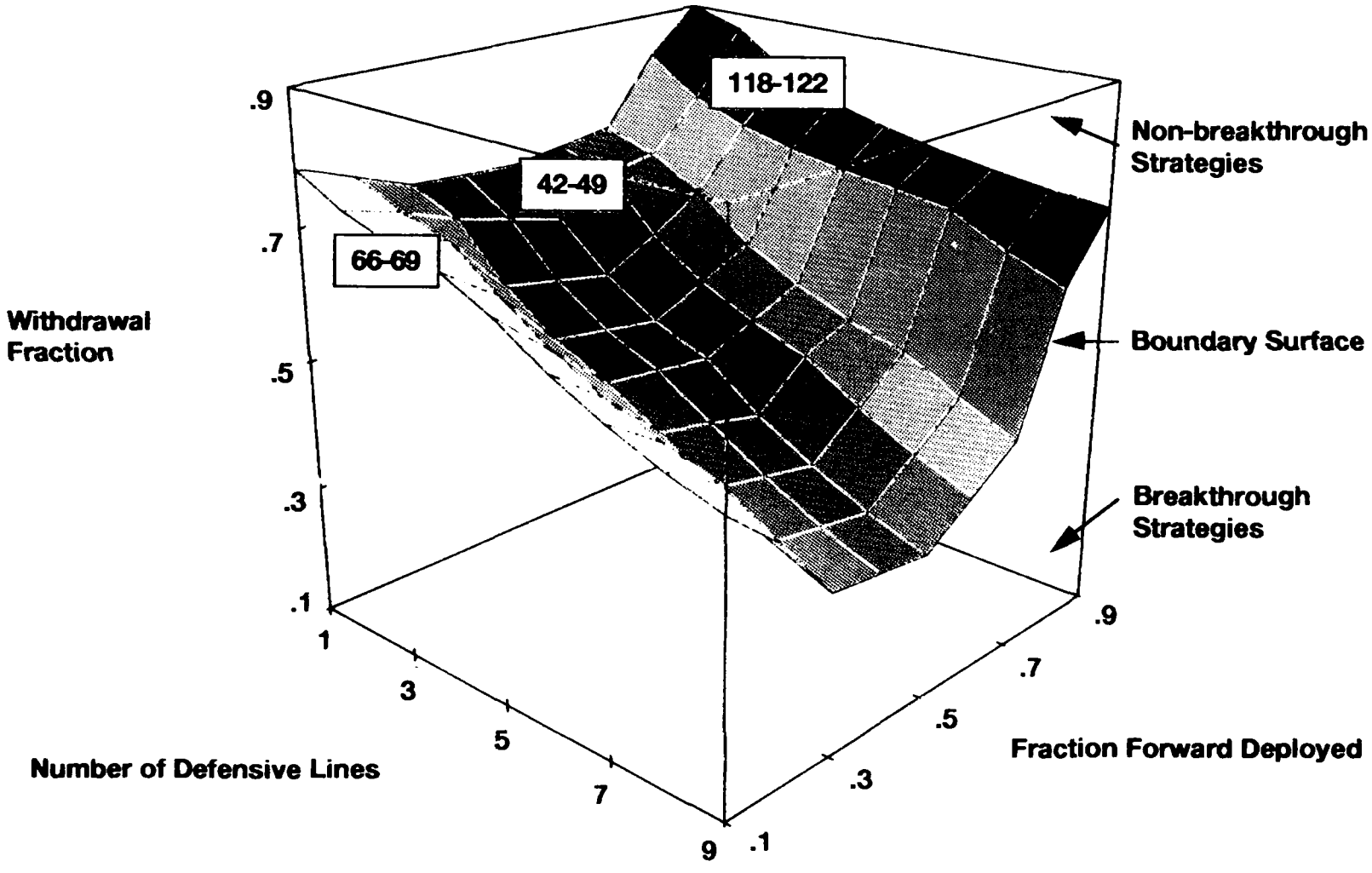


Figure 5. Boundary Surface of Non-breakthrough Strategies

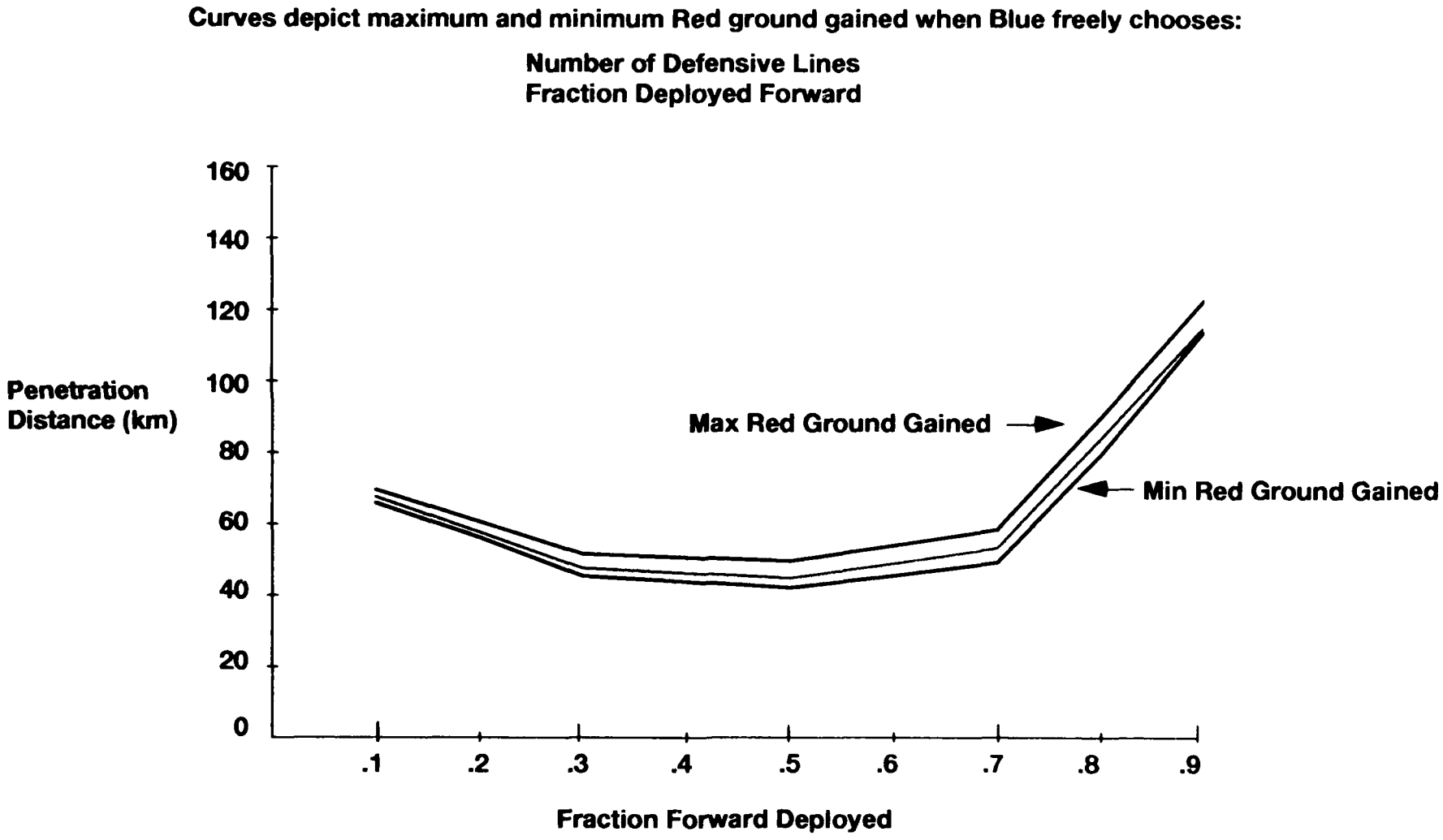


Figure 6. Maximum Red Ground Gained is Basically a Function of Fraction Forward Deployed

IV. COMPUTER OPTIONS FOR LARGE SCALE NUMERICAL EXPERIMENTS

A major theme of this project has been to examine how one could differently structure an analysis if the available computing speeds were essentially unlimited. This perspective is not too futuristic. The speeds of computers now commonly available to the analyst are such that what used to take hours or sometimes days to execute on a mainframe computer may now be executed on a desktop machine in less than one hour.

To investigate the implications of this trend in greater computing speeds, it was necessary to actually use a fast computer that would allow the execution of a large number of runs of some large simulation in a reasonable amount of time. Initially, a particular advanced parallel processing computer, the Multiflow Trace computer, was used. In subsequent phases of this project, a second parallel processor, the Encore Multimax computer, was investigated. For comparison, the performance of two desktop computers, the Compaq 386 and the Sun, was measured. These latter two machines, and their successors, are attractive from the cost/performance point of view. As a final option, therefore, this project investigated how a networked collection of such desktop computers also could be used to execute large numbers of runs of a given application.

A. MULTIFLOW TRACE

The Multiflow Trace is an advanced parallel processing computer that uses a new compiler technology, rather than special user coding of programming instructions, to exploit the parallel aspects of a simulation. The Trace computers are the first to implement the Very Long Instruction Word (VLIW) concept. The primary advantage represented by this compiler is its ability to parallelize a Fortran or C language program written according to standards without the user having to rewrite any source code.

This project used a Multiflow Trace 28/200, the largest of the older series of this computer. For two large combat simulations, the IDA Tactical Warfare (TACWAR) model and the IDA Theater Land-Air Model (TLAM), the Trace performed 15 to 20 times faster

than a VAX 785. While the newer Trace 28/300 would likely be faster, it may be unavailable; the Multiflow Corporation ceased conducting business in the Spring of 1990.

The key element of the VLIW approach is to use multiple processors to process the instructions of each line of a user's program in a parallel fashion. A simple example is given to illustrate how this works. In calculating *(a+b) * (c+d)* (using a Trace computer with 4 processors, e.g., the Trace 28/200), the quantities *a, b, c,* and *d* can be loaded into the processors in the first clock cycle. In the second clock cycle, *a+b* and *c+d* are performed. In the third cycle, the two results are added. In the fourth cycle, the result is stored.

By contract, a serial processor would require three cycles to load *a*, load *b* and add *a+b*. Three more cycles are needed to calculate *c+d*. Two more cycles are then required to multiply the results and store the answer. In this example, the Trace computer requires half the number of cycles as a standard computer.

The Trace computer also performs sophisticated code rearrangement in order to exploit portions of loops that in fact can be performed in parallel. One example is the zeroing out of an array.

B. ENCORE MULTIMAX

The Multimax computer represents a different approach to parallel processing than the Trace machine. The Multimax that this project used has 10 processors that can be used independently, although there are parallel Fortran programming commands available with the Multimax compiler. Each board runs at approximately the speed of the VAX 785. Up to nine simultaneous executions of a given application can be performed; the remaining processor is left free to perform administrative functions.

One can enhance the performance of this particular Encore computer in several ways. The internal memory could be increased (though the current system has 96 Megabytes of main memory; it is one of the largest memory amounts of any Multimax computer in use today). The data transfer rate from disk into memory could be increased with new hardware. The current rate is one Megabyte per second; a speed of 10 Megabytes per second is possible. More processors can be added; up to 20 are possible. Finally, each processor could be replaced by a faster processor. The current boards, of the 320 series, are four MIPS in capability. The 510 series boards are 16 MIPS in capability.

The Multimax computer thus has significant potential to perform rapidly many runs of a major model. In addition, when it is connected to a network by a fast data communications link (as is now the case with the T1 connection between the Multimax site and the VAX site), the output from large numbers of runs can be manipulated with ease.

C. POWERFUL DESKTOP COMPUTERS

There are versions of the Compaq 386, Compaq 486 and the Sun computers that are able to run the larger IDA combat simulation models. These machines perform at about the same speed as a VAX 8600 for the TACWAR model. Since they are relatively fast and inexpensive, sufficient numbers of runs of a given simulation may be obtainable from these machines also.

D. A NETWORKED COLLECTION OF WORKSTATIONS

It is commonly the situation that an organization may have several workstation type computers on a local area network to which is also connected large disk storage capability. Therefore, another potentially simple approach for performing large numbers of runs of a given simulation is to use a collection of such workstations (each of which can execute a given simulation). To make this workable, a mechanism is needed to allow the coordinated execution of a collection of runs to take place on a (local) network of such computers. Appendix A outlines one approach for doing this.

E. OVERALL COMPARISONS

To compare the various options for conducting large scale parametric analyses, the TACWAR model was installed and executed on several of the computers listed above. The TACWAR model is a 100,000 line Fortran program that simulates two-sided combined ground and air combat.

The following comparisons are noted.

The overall model comprises 282 Fortran source code files (total of 3.4 Megabytes), plus 75 Fortran files (total of .17 Megabytes) containing the code that contains the various common block specifications and parameter statements. There is one input file of size .9 Megabytes.

Table 1. Comparisons of Several Computers

System	VAX 8600	Multiflow Trace 28/200	Encore Max 320
Time to Compile	24 min.	19 hours	11 min.
Size of .OBJ	2.5 Mb	22.3 Mb	3.2 Mb
Size of Executable	1.7 Mb	10.4 Mb	2.0 Mb
0 cycles	28	23.2	126
2 cycles	49	28.8	198
4 cycles	70	34.4	270
10 cycles	134	52.6	495
30 cycles	346	109.0	1197
60 cycles	669	193.6	2265

Notes:

1) The TACWAR model comprises two phases of operation. The input data is initially read by the model in the first phase, cycle 0. The simulation of combat occurs on a cyclic basis after cycle 0. For these TACWAR runs, a cycle represents 12 hours of combat and corresponds to the major Do Loop of the main calling program of TACWAR. The timing statistics for the different cycle amounts (i.e., 0, 2, 4, 10, etc.) are all in seconds.

2) The times for 0 cycles are extrapolations based on the cycle 2 and the cycle 4 data.

3) The time for 60 cycles on the Trace is an extrapolation. Due to unknown reasons, the three computers did not give the same results for the same base case input file. The Multimax and the VAX 8600 diverged even on cycle 1. The difference between the Trace and the VAX case was large. By cycle 32, a programmed exit related to supply lines being too long caused the model to stop. The reason for the differing results across the computers may be due to different internal mechanisms for dealing with accumulated roundoff errors.

4) The long compilation time for the Trace reflects code rearrangement and loop unrolling activities of the compiler. This allows fine-grained parallelism to be exploited.

5) The compilation procedure on the Encore Multimax made use of ten processors. Thus the total time to compile was more than 100 minutes. The elapsed time is recorded in the table.

6) The execution statistics for the Encore Multimax were made using a single processor. For this version of TACWAR, nine simultaneous runs could be completed in 2265 seconds. Not all applications may be able to use this number of processors, due to limitations on virtual disk space. For the Theater Land-Air Model, only five runs could be simultaneously executed, due to some very large arrays in the model.

7) The computers represented here were selected mainly because they were available to the project. They are not of comparable generations, nor are they representative of capabilities that currently may be available.

V. POTENTIAL APPLICATIONS

A. OVERVIEW

In addition to testing a model, as was done for the FSR model, there are other examples of analyses that could be approached using graphical analysis methods. A common characteristic of many of these potential applications is that they involve evaluating tradeoffs among several input parameters, and several output measures of merit are jointly considered. A few of the topics identified in Figure 7 will be discussed in this section.

Combat	• Model Validation and Verification • Force Structure Tradeoff Analyses • Weapon System Survivability/Vulnerability Models • Cross-scenario Comparative Analyses
Economic	• Rubber Aircraft / Ship / Submarine Design Models • Pricing Models for Telecommunications Networks • Industry Input / Output Models
Database	• Questionnaire Responses • Census Data
Dynamic Systems	• Network Performance Models • Intelligence-based Indicators

Figure 7. Potential Applications of the Graphical Analysis System

B. VALIDATION AND VERIFICATION

There are three major aspects of validation and verification that could be addressed by the Graphical Analysis System: a) the analysis of input data, b) the analysis of the sensitivity of a model's output to variations in that model's inputs, and c) the comparison of a model's output with externally generated measures.

1. Analysis of Input Data

The objective of this activity is to provide some measure of the consistency and validity of a relatively large model's input data separate from an analysis of that model's output. Two possible ways to effect this assessment are: 1) to create a small model using the main or central algorithms of the model being validated, or 2) to compare the inputs to the outputs of a higher resolution model. In both cases, the focus is on the relative contributions of the multiple systems or capabilities being modeled as represented by the input data. The Graphical Analysis System can be used to simultaneously display the contributions of multiple systems according to multiple criteria.

2. Sensitivity Analyses

Two types of sensitivity analyses are of interest here: 1) using a given input database, conduct many sensitivity excursions on individual model inputs, and 2) conduct sensitivity analyses jointly on groups of inputs, where several inputs are varied independently. The objective is to reveal coding errors, cases of non-intuitive behavior (e.g., more resources cause one to do worse), and the extent to which certain inputs can be varied from a given input database's values and still remain valid. A key effort in this regard is to create additional output measures that are automatically recorded by the model for the purpose of model validation and verification.

3. Comparison With Externally Generated Measures

Even when a model's logic and basic input data have been reviewed and deemed acceptable, it is appropriate to calibrate the results using externally generated measures. This calibration is useful not only when developing the initial input database, but also when new data are obtained (e.g., from test exercises of new systems) that need to be reflected in subsequent analyses.

The Graphical Analysis System can facilitate calibration (and cross-model comparisons) by displaying in one plot the different data sets to be compared. The key feature is the ability to display multiple surfaces in one plot. Because there may be several inputs that can simultaneously be varied to calibrate a model, the ability to show multiple calibrations simultaneously can be quite important.

One example application of the Graphical Analysis System to the testing of a model is given in Appendix B.

C. ECONOMIC MODELING APPLICATIONS

IDA has developed several cost and pricing models that estimate a system's procurement and operating costs based on its design parameters. Known as rubber design models, they allow the analyst to experiment with parametric variations of the key descriptors of the system. The Graphical Analysis System can facilitate the analysis of such variations by showing how the assessments varied as multiple parameters were varied.

D. DATABASE APPLICATIONS

Another application occurs when questionnaire data contain responses that can be quantified numerically, e.g., ranges of salary, years of education, seniority in the workplace. Instead of calculating statistical measures to describe the responses, one could simply display the two and three dimensional histograms of responses and thus see the distributions directly. One also can display statistical measures on the responses surfaces (i.e., the three dimensional histograms) through the use of color.

E. DYNAMIC SYSTEMS

The Graphical Analysis System also can be used in cases where there is no underlying computer model generating output data. Instead, some system is being monitored and perhaps scores of indicators are being recorded. The Graphical Analysis System can portray collections of indicators as time-varying response surfaces. By viewing how a given collection varies over time, and by simultaneously displaying groups of related measures (i.e., as multiple surfaces), correlations among the indicators may be more easily discerned.

VI. RESEARCH ACTIVITIES

This project is currently pursuing research in three areas: a) improvements to the Graphical Analysis System, b) the automatic detection of key sensitivities and c) the interpolation of results for new input regions.

A. IMPROVEMENTS TO THE GRAPHICAL ANALYSIS SYSTEM

The graphical display of model output is a very efficient way for the analyst to gain an understanding of the sensitivities of the model, or to determine whether there are critical regions of input values that should be examined more closely. It is important, therefore, to be able to obtain precise numerical information from a graphical display of results. Although these results may be depicted as curves and surfaces, Mathematica provides the capability to obtain precise coordinate data by simply positioning the cursor on points of interest. In this manner, the graphical displays on the computer monitor become a type of input mechanism. By knowing the coordinates of the points of interest, the analyst is then better able to exploit the insights gained from the basic graphical displays.

Other improvements concern the user interface, the efficiency of various internal algorithms of the Graphical Analysis System, and how the system might be redesigned to take advantage of either a multitasking operating system or its implementation within a local area network.

B. AUTOMATIC DETECTION OF KEY SENSITIVITIES

If the dimensionality of the multidimensional data set is sufficiently large, it may be too time-consuming to examine graphically all possible ways of displaying the data. If the experiment that produced the data was designed to test the sensitivity of various measures to certain input variations, then there are several analytic measures that can be used to determine which displays of the multidimensional data would and would not show any interesting variations.

One can analytically test, for example, whether in any one plot a family of surfaces is affine, constant or all close to one another. For an animated series of plots of families of

surfaces, one can measure if the sequence eventually shows any of the above behavior. The use of such automated filtering of the data can assist the analyst in determining which graphics to look at and which to ignore. In fact, even knowing that certain displays would result in 'uninteresting' pictures may be valuable information in and of itself.

C. INTERPOLATION OF RESULTS FOR NEW INPUT REGIONS

While it is now feasible to perform large scale parametric analyses involving hundreds or thousands of cases at a time, there are situations nevertheless where one either cannot or does not want to execute every case in a multidimensional grid of cases. One case occurs when a new input region is being examined for the first time. Here, one might construct a multidimensional grid of cases, with several grid points in each dimension. Initially, though, one could execute only the 'corner' points of the lattice. Then, having determined that there is significant variation across the extremes of the lattice of cases, one could execute a succession of other lattice points. To display the results derived from the partially completed lattice of cases requires an interpolation method.

D. CONCLUSION

The scope of analysis undertaken for a given study normally is limited by how many model runs one can effectively perform and understand. The computer technologies now becoming widely available have the potential to increase significantly one's ability to more easily and thoroughly use computerized models. By designing structured experiments, by using faster computers to execute more runs, and by using new tools to analyze the model output, the ability to perform complex analyses will be greatly enhanced.

REFERENCES

[1] Sobell, Mark G., *A Practical Guide to the Unix System*, The Benjamin/Cummings Publishing Company, Inc., Redwood City, CA, 1989.

[2] Wolfram, Stephen, *Mathematica, A System for Doing Mathematics by Computer*, Addison-Wesley Publishing Company, Inc., Redwood City, CA, 1988.

[3] *The Multiflow Trace Computer, User's Guide and Reference Manual*, Multiflow Computer, Inc., Branford, CT, 1989.

[4] *The Encore Fortran-77 Manual*, Encore Computer Corporation, Marlborough, MA, 1986.

[5] Biddle, Stephen D., *Force to Space Ratios and Defense Effectiveness*, IDA Paper P-2380, Institute for Defense Analyses, Alexandria, VA, forthcoming.

APPENDIX A

DISTRIBUTED COMPUTING ON A LOCAL AREA NETWORK

CONTENTS, APPENDIX A

A. PROBLEM STATEMENT 1

B. ASSUMPTIONS 2

C. PROPOSED MODEL SOLUTION 2

D. NEAR TERM IMPLEMENTATION 4

1. The Administrative Software 4
2. Remote Servers 5
3. Driver 5

E. TECHNICAL CONSIDERATIONS 6

1. Failure Detection and Handling 6
2. Resource Control 6

F. GENERALIZATIONS 6

1. Extensions 6
2. Impact of Relaxation of Assumptions 7
3. Scheduling and Resource Control Alternatives 8

G. OTHER OPPORTUNITIES 9

1. Potential Applications of the Initial Distributed Computing Model 9
2. Potential Applications of an Expanded Distributed Computing Model 9

A. PROBLEM STATEMENT

The purpose of this section is to outline how a networked collection of computers can be coordinated in their execution of multiple runs of a given model. The main application of such a capability is to those analyses that seek to perform many sensitivity excursions of a specific model. By making use of more computers simultaneously, one has a potentially easy and inexpensive way to achieve a significant increase in the number of runs that can be performed in a given time period. In this manner, a more thorough and extensive analysis could be performed.

The specific problem introduced above represents one way to execute a model in a distributed computing environment. There are perhaps more interesting, but significantly more difficult, problems that could be addressed, e.g., how to distribute the calculations of one run of a model across the network. Such a capability would allow one model run to execute faster, perhaps, or allow larger problems to be solved. The overall benefit of this capability is application specific. It depends on the degree of parallelism contained within the model's calculations. The problem that is addressed by this paper is, by comparison, more widely applicable since any model could be used.

To make the problem tractable, this paper will make several simplifying assumptions concerning the communications protocols, the computers on the network and the disk storage capacity associated with the network. The implications of these assumptions and how they might be relaxed in subsequent investigations will be discussed. Furthermore, in order to make the discussion as specific as possible, a variant of the IDA TACWAR model is assumed to be the application that is to be executed many times. By choosing a particular user application, an actual context for the approach outlined in this paper may be created.

The TACWAR model is a 100,000 line ANSI Standard Fortran program. Each run, or execution of TACWAR proceeds as follows. First, the file FILENAMES.n is read. This file contains the names of the two input files: INPUTCORE.DAT and INPUT.n, where the suffix n is an index specifying the run, e.g., n equals 1, 2, etc. Note that each run reads in the file INPUTCORE.DAT. Each run produces one file, FULLOUTPUT.n, a file 3 Megabytes (Mb) in size. As a last step in each run, a subroutine is called which reads in FULLOUTPUT.n and write out a file REDOUTPUT.n, whose size is .03 Mb. At the conclusion of each run, FULLOUTPUT.n can be deleted; REDOUTPUT.n is saved.

The overall task is to have the network execute a series of runs of TACWAR, where the series is specified as follows. The user wishes to execute, say, 1000 runs of TACWAR overnight. First, the user creates the files INPUT.n and FILENAMES.n for n equal 1 to 1000. These are

assumed to reside in one directory, /user/tacwar. Each run is considered independent from the others. After the completion of the entire series of runs, the files REDOUTPUT.n are copied back to the /user/tacwar directory. The last step is to then execute a second Fortran program that reads in all the REDOUTPUT.n files and write out one file OUTPUT.ALL .

It is assumed, furthermore, that the executable image of TACWAR is 8 Mb in size, that OUTPUT.ALL is 1 Mb in size and that each run requires approximately 10 minutes. Finally, each run of TACWAR obtains in some as yet unspecified manner the index n that distinguishes each run.

B. ASSUMPTIONS

The following assumptions are made in order to make the task potentially less complicated. Based on the initial findings, these assumptions can be relaxed and a more general solution constructed.

The first assumption is that the environment is homogeneous, i.e., each computer is the same, each computer has the same system software and each computer will be running the same user application software (but with different input files). The software restrictions simplify the configuration management by requiring only one copy the of software application that is to be run across the distributed network. The environment restriction simplifies the communications because it removes from consideration the problems involved with different machine representations of data. Other restrictions on the environment include a unified view of the distributed file system; this simplifies the distributions and collection of applications and data.

The assumption that there is one application that must be executed many times with different input files to produce a set of output files simplifies the system in that each copy of the application is independent of all other copies. The communications between executing programs will be restricted to an initial set of input files and a final set of output files. The effect here is to simplify the scheduling of processes.

C. PROPOSED MODEL SOLUTION

While several methods for implementing the solution exist, only one model is considered in this section. The 'centralized model' of execution is used for this prototype. The centralized model states that one processor in the network assumes responsibility for all distributed processing in the network. The responsible processor is called the 'server.' A user is required to register his application with the server, and the server is responsible for executing the user's requests.

For this prototype, the centralized model will be used because it makes resource allocation, monitoring, fault detection, and scheduling easier to implement.

In the centralized model, each processor in the network that participates in the distributed execution of programs contains special software called the 'remote server.' The remote server implements the protocols necessary for the processor to communicate and cooperate in the overall effort. Thus, the network is configured by designating a server to control all distributed processing and designating a group of processors to participate in the distributed processing as remote servers.

A request for distributed processing by a user consists of 1) the application program to be executed, 2) the inputs to the application program, 3) a program to process the outputs from the application program, and 4) a list of remote servers that should or should not be used by the application. Given this information, the server is able to construct the distributed processing scheme.

The application program in this model is slightly different than what is conventionally thought of as an application. In this model the application program contains three parts. First, the application program specifies the resource requirement needed in order to successfully execute the program on a processor. For example, the amount of main memory used, the amount of temporary disk space used, and the processing time normally required. Second, the application program contains an optional output filter that is used to reduce the size of the output before transferring the output to a central location for any subsequent processing. Finally, the applications program contains an execution 'script' that combines the source program and an output filter together into a single executable entity.

The inputs and outputs of the application program are given qualified names that serve to distinguish the many files. The qualifier will be a numeric suffix to the input and output file names (e.g., Input.7 is the input file used by the 7th application program to be scheduled). The program to process the output file will be activated by the server after all of the distributed applications have completed.

Finally, the list of remote servers is an optional specification that allows the user to control exactly which remote servers may be used in the distributed processing. This does not guarantee that these remote servers will be used. Resource requirement could result in further restrictions. For example, an application that requires 8 Mb of storage may not be executable on all of the remote servers.

Given a properly formulated request for distributed processing, the server then constructs a 'driver' program that will begin executing on the server at a given time. The driver is the part of the server that is active while the application is running across the network. The driver program is responsible for activating the available remote servers, insuring that the remote servers are capable of provided the necessary resources, distributing the application program, scheduling the execution of the application, and monitoring the progress of the system.

Two activities of the driver are of particular interest. The scheduling activity is a simple demand driven schedule. The remote server accepts work from the scheduler, performs that work, reports the results and then awaits further work assignments from the scheduling portion of the driver. The scheduler of the driver determines which instance of the application should be assigned to the remote server by comparing the resource requirements of the applications with the resources available at the remote server, and then makes that assignment.

The second activity of interest is the monitoring process. The monitoring process is necessary to handle unexpected faults that may occur within a remote server. Should a processor experience a severe fault, the remote server will (if it can) report the trouble to the driver. The driver will then take action based upon the type of trouble. The driver may have to abandon all or part of the computation. The driver may then try to recover from the error. At times, it may not be possible for the remote server to report troubles to the driver. To handle these conditions, a maximum expected computation time is specified for each application. If the remote server does not respond within that time frame, the driver will assume that the processor associated with the remote server is no long healthy and will take appropriate actions. In particular, the run that was not successful will be rescheduled.

D. NEAR TERM IMPLEMENTATION

There are three basic components that would have to be developed for a prototype system capable of distributed execution as outlined above. While the IDA Computer & Software Engineering Division (CSED) has tools with capabilities similar to those needed here, modifications are required. To fully implement an initial prototype may require, therefore, approximately 4 professional staff months of effort.

1. The Administrative Software

This software allows a user to register on the server an application for distributed execution. This software does not currently exist, but its implementation does not represent a problem since the prototype will handle only simple configurations and is not expected to recover

should the server processor crash. The software will have to perform a check on the network and a set of data files to determine if the application submitted is acceptable for execution. The administrative software will then customize a driver to conduct the distributed execution of the applications at a later point in time.

2. Remote Servers

A remote server is a command interpreter with a simple language: run program X with parameters Z. The remote servers are loaded into all available machines by the operating system. This implies that the system administrator will have to place the remote server in the list of network services provided. The operating system will then make sure that the remote server is available to the network. The remote servers sit and wait for commands that are addressed to them specifically. For security reasons, they are owned by the administrative software and will respond only to commands from the administrative software.

Remote servers are available within CSED. These servers are more complex than what is needed for this configuration. The current remote servers, moreover, provide no support for fault detection. The remote servers now available can be modified by removing the complex routing they currently perform and by adding some reporting mechanisms to support fault detection.

3. Driver

The driver must perform two basic operations, scheduling and monitoring. The scheduling system proposed is demand driven and simple to implement. The only part that will require some effort is the comparison of resource requirements to resource availability, but this is not critical to the success of the prototype. A simple method can be used. The monitoring problem is more critical in that the system must detect faults in order to be successful. The difficulty in detecting faults arises when one tries to minimize the number of false detections. False detections will decrease the overall throughput of the system. For the initial prototype system, a simple time out method may suffice. In the future, more sophisticated fault detection schemes may be warranted based on specific user applications and the throughput that is required.

E. TECHNICAL CONSIDERATIONS

1. Failure Detection and Handling

Failure detection and handling is a particularly difficult problem. Nevertheless, it is possible to provide isolation from most failures. In this system, it is possible to estimate the time required to execute a particular part of the application. This time can then be used as an indication of failure. When a processor is assigned a job by the scheduler, it is required to return a result within a specified time frame. If a result is not returned, then the scheduler will assume that the processor has failed, and will attempt the reschedule the job on another processor. Should the first processor return a result after the job is rescheduled, the result will be ignored. This is the simplest form of failure detection and the form that is proposed for the prototype system.

2. Resource Control

Resource Control (controlling access to processors, main memory, communications, and disk space) is in general a challenging problem. For this prototype, we will assume the the amount of disk space and memory used by an application and available on a remote server is predetermined and does not change during the execution. The resource checks thus can be made once, before the driver begins assigning applications to remote servers.

F. GENERALIZATIONS

The current solution is limited in that there is one application program that is to be replicated many times across a network and executed with different input files producing different output files. This represents a limited class of distributed programming applications. A larger class of problems can be handled by relaxing certain assumptions made for the initial prototype.

1. Extensions

The following possible extensions are listed in the order in which they should be added to the prototype should demand for the system exist.

1) Remove the restrictions for a single server to handle all requests for distributed programs. This would require a more complex form of the administrative program that could coordinate a series of request for distributed processing. It would allow a user to register a request from the PC or workstation in his office.

2) Remove the restrictions for homogeneous architectures (still require homogeneous operating systems interfaces). This increases the available hardware but requires a more sophisticated configuration management system since multiple binaries may have to be maintained. One would still require that the system provide a unified file system and we would now require that all systems provide the same communication protocols.

3) Remove the requirement for a unified file system. This will increase the available hardware but will require a more sophisticated remote server and driver (this is currently done in CSED). It would still be assumed that all systems provide the same communications protocols.

4) Remove the requirement that the application being distributed consist of totally independent programs. Now one could allow one program to depend on the output of another program. This would require a more sophisticated driver (scheduler) to implement the data dependency graph between the programs. This would also require a method for the users to specify these dependencies.

2. Impact of Relaxation of Assumptions

Throughout this discussion a series of assumptions have been stated to simplify the solution to the problem. It is assumed that the problem to be solved involves executing many copies of the same program on different input sets. It is also assumed that the problem will be solved on a network of homogeneous computer systems that implement a unified view of the distributed file system.

If the constraints for homogeneous computer systems are removed, then we must examine two areas. First, the computer hardware is different across systems. Different hardware architectures will require that all systems implement the same data-exchange protocols. This is necessary to convert data from one machine to the next. For example, a VAX stores data bytes in a different ordering than a Sun. Thus a common method to communicate is needed to interpret the different byte ordering of data. Different computer hardware will also require that multiple versions of the object code (executable code) exist. This complicates the management of programs from the user's point of view. From the system's point of view, additional information about the operating system will have to be stored in the files that describe the network of available processors.

Second, the computer operating system is different across systems. This represents a much more serious problem, and some operating systems may not be suitable. Therefore, deviation in operating systems may be unacceptable. All operating systems must implement a

remote procedure call (RPC). Operating systems that do not implement this feature will not be considered. RPC allows the driver to schedule and communicate with the remote servers. The driver - remote server interface is implemented via RPC. Operating systems that participate in the unified view of the distributed file system are most advantageous; however, systems that fail to provide this can be supported. This support will require additional protocols to be established between the driver and the remote server. These protocols are needed to communicate the object code and the input and output files needed by the object code. In the unified file system, the object code, input, and output files are distributed automatically by the operating systems.

3. Scheduling and Resource Control Alternatives

More elaborate systems for specifying data dependencies between the programs can be used by simple modification of the scheduling algorithms of the driver. With a more general scheduling algorithm it would be possible to run several different programs where one program may require the output of another program before it can begin running. The user interface will also have to be modified so the dependencies between programs could be specified. This would increase the potential use of the system with only minor modifications in that the system could now be used to execute programs with coarse-grain parallelism provided that the program could be broken into separate pieces. This still assumes that each piece will run to completion before the output is relayed to the next piece of the problem. This is often called a 'filter' since the data flows through the program and is modified to produce the input for the next program. The linear flow of data restricts the focus to programs that have no cyclic dependencies. In other words, a program cannot produce output that will eventually be routed back to itself.

Finally, the most general problem occurs when each program (piece of the problem) executes continuously, and many inputs and outputs are consumed and produced during the execution of each program. In this case, the outputs from one program will have to be relayed to the destination before the program completes. In fact, the program may run continuously, consuming inputs produced by one program and producing outputs for another program. This requires an event driven scheduler that signals each application when data has arrived. The signal will trigger the application to process the newly arrived data. The process may then send output to another process.

G. OTHER OPPORTUNITIES

1. Potential Applications of the Initial Distributed Computing Model

The initial model is designed to address the case where many runs of the same application are to be executed with different input sets. To take full advantage of the potential that the initial system may offer, the application should be a complex model requiring the better part of one hour to execute each run, e.g., the TACWAR model. The resource tradeoff analyses that are now addressed by TACWAR may be affected by factors that currently cannot be examined due to constraints on the number of model runs that can be performed and analyzed. If a network could achieve a throughput of hundreds of runs overnight, then more of these factors may be considered.

2. Potential Applications of an Expanded Distributed Computing Model

The expanded models as discussed in the section on Scheduling and Resource Control Alternatives are useful in implementing coarse-grain parallel solutions to problems. The first alternative, where several different types of programs can be run and a order dependency between the programs is established, will be useful in implementing parallel versions of the 'filter' problem. A 'filter' problem is a process where inputs are entered into one or more program to be processed. When they are completely processed (filtered) by the first set of programs a set of outputs is produced and passed on to be processed by another set of programs. Since an input, X, is passed into a procedure that then outputs Y which is then passed into another procedure and so on, the entire process is logically thought of as a filter. At each stage in the processing the data is transformed and relayed to the next stage.

The second alternative represents a solution to a much broader class of problems. Any course-grain parallel programming problem can be solved with this model; however, the implementation of this model is significantly more difficult than any of the others proposed because a completely general solution for controlling programs and passing data between these program must be used to implement this model. The SDI (Strategic Defense Initiative) Architecture Dataflow Modeling Technique (SADMT) simulation framework is such a model and should be examined as a potential solution to this problem if the demand for parallel computation at IDA becomes significant.

APPENDIX B

THE APPLICATION OF GRAPHICAL ANALYSIS METHODS TO MODEL VALIDATION AND VERIFICATION: AN EXAMPLE

CONTENTS, APPENDIX B

A. INTRODUCTION 1

B. SUMMARY 1

C. ANALYSES 3

1. Structure of the Cases 3
2. Data and Output Measures 3
3. Observations on FEBA Movement Results 5
4. Output Measures That May Help Explain FEBA Movement 8
5. Aircraft Activity Past Day 20 9
6. Anomalies in the Data 10
7. Contribution of BAI Kills 11

D. CONCLUSIONS 11

TABLE

1. Correspondence Between Input Combinations and Case Numbers 4

ANNEXES

B-1. GRAPHICAL REPRESENTATION OF SELECTED CEM OUTPUT

B-2. TABULAR REPRESENTATION OF SELECTED CEM OUTPUT

A. INTRODUCTION

This section describes a short term analysis undertaken on behalf of the US Army Concepts Analysis Agency (CAA). The objective was to demonstrate the utility and potential of Graphical Analysis System developed as part of this IDA Central Research project. This objective was accomplished by applying the new methods to the results of a previously conducted CAA analysis based on the CAA Concepts Evaluation Model (CEM).

The CAA analysis was designed to explore the effects of variations and tradeoffs among several inputs. In effect, though, the cases developed by CAA represented an example of the type of parametric sensitivity analysis that one might perform when testing a model as part of its verification and validation. In the following discussion it is shown how this parametric approach, coupled with the graphical displays of the results, efficiently revealed anomalous model behavior and pointed to what the source of the behavior was. Based on these insights, it was found that due to limits imposed by a certain section of the model's code, some of the input variations that were examined exceeded the bounds of validity.

Though this is only one example of model logic that is either incorrect or not well known by the model users, such defects will accompany nearly any complex model. What this section demonstrates is that one can systematically test a model in order to understand its behavior more reliably and thoroughly.

B. SUMMARY

The objective of the CAA analysis was to assess the contribution to the overall course of combat of aircraft assigned to the Close Air Support Attack (CAS) mission. To conduct this analysis, four key inputs to the CEM model were varied and a variety of output measures were recorded. The input variations concerned both initial inventories of aircraft as well as three parameters that govern the degree to which the tactical fighters are allocated to the CAS mission. The main measure of effectiveness used in assessing the combat results is Mean Forward Edge of Battle Area (FEBA) Movement.

IDA analyzed the CEM outputs using the recently developed Graphical Analysis System. The major objective of IDA's analysis was to assess the relationship between changes to the CAS aircraft allocations and the FEBA output measure. A secondary objective, but one that is important to future analyses, was to comment on which output measures best help one understand and interpret such relationships.

There are five main observations resulting from the graphical analysis of CEM outputs.

1. The effect on FEBA movement of the variations of aircraft inventories and CAS mission allocation parameters is essentially determined by day 20. Thereafter, while FEBA movement does occur, it changes essentially the same amount for all cases.

2. A detailed explanation of the comparative FEBA movement results among the cases analyzed requires a more extensive collection of output measures than those developed by CAA for this analysis. Measures that record cumulative kills, weapons remaining as well as other components of the FEBA movement algorithm may be required. Marginal loss and exchange rates may be only partially related to the overall FEBA movement.

3. Certain combinations of CEM inputs relating to CAS sorties are invalid. If too many aircraft are assigned to CAS or if the individual raid size is too small, then fewer than the intended number of sorties may result. In fact, the model logic may execute zero sorties in any army sector to which more than 256 raids are assigned.

4. Having established that additional numbers of aircraft do not in and of themselves contribute directly to improved FEBA position past day 20 (item 1 above), there is no direct evidence that the remaining aircraft contribute significantly to additional attrition of ground weapons. The possible exception may be that the BAI (Battlefield Air Interdiction) kills occur mainly past day 20. The available data are not able to illuminate this latter point.

5. Other constraints, beyond those cited above, may apply. For example, by comparing cases 1, 4 and 7 in Figures 7 and 11, one observes that more Blue aircraft are killed (thus the sortie constraint likely did not affect these cases), yet fewer Red tanks are killed. This seems to contradict a basic 'more should do better' argument. In order to understand these effects more thoroughly, and to determine whether they are due to errors in the model's logic, output measures must be identified and recorded that will illuminate such instances.

This remainder of this paper is organized as follows. Section C presents the details of the analysis. Section D contains a few recommendations for future work in this area. There are two annexes - Annex B-1 contains graphical representations of CEM output; Annex B-2 presents the CEM output in tabular format.

C. ANALYSES

1. Structure of the Cases

Thirty-six cases were developed by joining the following model input variations in all possible combinations:

Variable Name	Description	Possible Values		
NOPLANS	Added Number of Tac Fighters	300	1000	2500
CASIPS	Minimum Fraction of Initial CAS Allocation That Remain on CAS	.1	.5	.9
CASDATA	Raid Size	12	2	
INITAFR	Initial Allocation of Tac Fighters to CAS	.2	.7	

Table 1 displays the correspondence between the case numbers (1 through 36) and the various combinations of input variations. The values of the latter three variables (CASIPS, CASDATA and INITAFR) replace the corresponding base case values in all 36 cases. The inventory variations represented by NOPLANS are additive to the base case inventory of 1001 tactical fighter aircraft (cf Annex B-2, Measure 2).

2. Data and Output Measures

There are eleven measures provided by CAA as follows:

Measure 1 Mean FEBA Movement - by case number and time sample

Measure 2 Blue Aircraft Remaining at Start of Cycle - by case number and time sample

Measure 3 Blue Aircraft Killed During Cycle - by case number and time sample

Measure 4 Blue Tank Losses During Cycle - by case number and time sample

Measure 5 Red Tank Losses During Cycle - by case number and time sample

Measure 6 Blue Armored Personnel Carrier (APC) Losses During Cycle - by case number and time sample

Measure 7 Red APC Losses During Cycle - by case number and time sample

Measure 8 Blue Aircraft Loss Rate - by case number and time sample
(Loss Rate = Number Killed During Cycle / Number At Start of Cycle)

Table 1. Correspondence Between Input Combinations and Case Numbers

Case Number	NOPLANS	CASIPS	CASDATA	INITAFR
1	300	.1	12	.2
2	1000	.1	12	.2
3	2500	.1	12	.2
4	300	.5	12	.2
5	1000	.5	12	.2
6	2500	.5	12	.2
7	300	.9	12	.2
8	1000	.9	12	.2
9	2500	.9	12	.2
10	300	.1	2	.2
11	1000	.1	2	.2
12	2500	.1	2	.2
13	300	.5	2	.2
14	1000	.5	2	.2
15	2500	.5	2	.2
16	300	.9	2	.2
17	1000	.9	2	.2
18	2500	.9	2	.2
19	300	.1	12	.7
20	1000	.1	12	.7
21	2500	.1	12	.7
22	300	.5	12	.7
23	1000	.5	12	.7
24	2500	.5	12	.7
25	300	.9	12	.7
26	1000	.9	12	.7
27	2500	.9	12	.7
28	300	.1	2	.7
29	1000	.1	2	.7
30	2500	.1	2	.7
31	300	.5	2	.7
32	1000	.5	2	.7
33	2500	.5	2	.7
34	300	.9	2	.7
35	1000	.9	2	.7
36	2500	.9	2	.7

Measure 9 Tank Exchange Ratio for the Cycle - by case number and time sample (Tank Exchange Ratio = Blue Tank Losses/Red Tank Losses During Cycle)

Measure 10 APC Exchange Ratio for the Cycle - by case number and time sample (APC Exchange Ratio = Blue Tank Losses/Red Tank Losses During Cycle)

Measure 11 Total BAI Kills as of Day 60 - by case number

Each of these measures, except BAI Kills, was expressed in terms of a 6 element time-series, corresponding to cycles 1, 2, 3, 5, 8 and 15. As each cycle represents 4 days of combat, the discussion below sometimes will refer to days 4, 8, 12, 20, 32 and 60 rather than the corresponding cycle number. The BAI Kills measure gives the cumulative BAI kills by the end of cycle 15.

Because estimates of cumulative losses were thought to be helpful, five additional measures were constructed. Based on those measures provided by CAA that gave losses occurring within a cycle, estimates of the cumulative losses were derived for aircraft, tanks and APCs. While the calculated result is exact through cycle 3, a simple linear interpolation was used to estimate losses for those cycles not explicitly listed, namely cycles 4, 6, 7, and 9 through 14. The five resulting measures are as follows:

Measure 12 Cumulative Blue Aircraft Killed - by case number and time sample (Calculated from data provided by CAA)

Measure 13 Cumulative Blue Tanks Killed - by case number and time sample (Calculated from data provided by CAA)

Measure 14 Cumulative Red Tanks Killed - by case number and time sample (Calculated from data provided by CAA)

Measure 15 Cumulative Blue APCs Killed - by case number and time sample (Calculated from data provided by CAA)

Measure 16 Cumulative Red APCs Killed - by case number and time sample (Calculated from data provided by CAA)

Annex B-2 contains the data for the 16 measures listed above.

3. Observations on FEBA Movement Results

The main observation regarding the FEBA movement results is that the contribution attributable to additional aircraft or to greater emphasis on the CAS mission is essentially complete by day 20 (i.e., cycle 5). This claim is supported by a visual examination of the

response surfaces of FEBA movement, when this responsiveness is measured with respect to changes in aircraft inventories and CAS mission allocations. In general, the shapes of the various response surfaces do not change between day 20 and day 60. The surfaces do, though, undergo a nearly uniform vertical movement of small magnitude. This indicates that the cause of this movement (after day 20) was likely due to some effect different from increased numbers of aircraft or the other variations in mission allocation parameters.

The FEBA movement response surfaces are shown in Figures 1 through 6 of Annex B-1. Figures 1 through 3 show the FEBA movement values as of day 20, day 32 and day 60 for the eighteen cases where INITAFR, the initial fraction of tactical fighters allocated to CAS, is .2. Figures 4 through 6 show the FEBA movement values as of day 20, day 32 and day 60 for the eighteen cases where INITAFR, the initial fraction of tactical fighters allocated to CAS, is .7 . All 36 cases are represented in these two sets of figures.

An examination of Figures 1 through 3 reveals that the shapes of the FEBA movement response surfaces for day 32 and day 60 are quite similar to the shapes seen in Figure 1 for day 20. The difference among these figures is not the shape of the surfaces, but rather there is a general vertical rise of the pair of surfaces when going from day 20 to day 32 and then to day 60. This vertical rise is nearly uniform across all eighteen cases. Specifically, when going from day 20 to day 32, all points in Figure 1 rise by 1.0 +/- 1.1 (except for case 14). When going from day 32 to day 60, the surfaces rise by 1.8 +/- .9. The units are kilometers of advance of the FEBA.

A similar description holds for cases 19 through 36, as seen in Figures 4 through 6. Here, the surfaces rise by an average of 1.4, with a range from .5 to 2.8, when going from day 20 to day 32. The surfaces rise by an average of 2.1, with a range from 1.0 to 3.3, when going from day 32 to day 60.

The above discussion allows the analysis of FEBA movement to occur in two phases. In the first, the results are dependent on the contribution of the air combat resources; in the second, when the air forces are heavily attrited, other factors become dominant. Now that the point has been made concerning how these two phases may be observed in terms of model outputs, it is possible to assess the overall contribution of the air resources by seeking a general underlying measure of the degree to which CAS missions affect FEBA results.

A rule of thumb that relates FEBA movement to aircraft lost will be offered as an initial attempt to unify the general effects depicted in Figures 1 and 4 (i.e., the day 20

results). The objective of this rule is to determine a ratio that relates ground gained (relative to a reference case) with the intensity of use of aircraft on CAS missions.

One may compare, for example, case 1 with case 9. For these two cases, the raid size is 12 and as Figure 1 shows, the FEBA results are positively related to increases in aircraft inventories. Moreover, in case 1 the minimum fraction of CAS aircraft that remain allocated to CAS is low (value is .1). Case 1 corresponds, therefore, to a case where the CAS contribution to FEBA movement is likely near the minimum. Similar reasoning for case 9 shows that in this case, the CAS contribution is likely near the maximum. Note, though, in Figure 1, the input INITAFR is at the lower setting of .2. Using the values for Measure 1 (FEBA) and Measure 2 (Aircraft Remaining at Start of Cycle) as given in Annex B-2, one can estimate the following: comparing case 9 with case 1, the FEBA movement at day 20 was 6.7 kilometers greater. While case 9 had 2500 aircraft added to the base amount, there remained 1954 at the start of cycle 5. In case 1, which had 300 aircraft added to the base amount, 815 remained at the start of cycle 5. After some algebra, one determines that between case 1 and case 9, 1061 additional aircraft had been attrited by the start of cycle 5. One may now observe, for these two cases, a ratio of .63 kilometers of improved FEBA movement (by day 20) per 100 aircraft attrited on CAS missions.

When the same ratio is calculated using cases 19 and 27 of Figure 4, one determines a ratio of .64 kilometers of improved FEBA movement (by day 20) per 100 aircraft attrited on CAS missions (11.4 kilometers improved FEBA movement for 1771 additional attrited aircraft). While these two examples may not have precise predictive ability, they may indicate a useful ratio to experiment with in future analyses.

A last observation on the FEBA movement results concerns the apparent negative relationship between FEBA movement and increasing numbers of aircraft when the input raid size equals 2. This relationship is clearly evidenced in Figure 4, in the dark surface. Before the model's internal limit on sortie execution was made known to this study, two possible explanations were formulated and examined using the Graphical Analysis System, as discussed next.

The first hypothesis assumes that raid size does not affect the number of aircraft that fly sorties per day. Rather there would be more sorties of a smaller raid size compared to a corresponding case with raid size of 12. With smaller raids, the likelihood of the aircraft being overwhelmed by the defenses would be greater, leading to fewer Red tanks killed and more Blue aircraft killed. Though there are indications that fewer Red tanks are attrited

when raid size is 2, there are fewer, not more, aircraft killed when compared with corresponding cases that have raid size equal 12. This hypothesis is thus rejected.

The second hypothesis assumes, therefore, that when the raid size equals 2, fewer sorties are flown due to some model constraint. This could explain both the lower Red tank attrition and lower Blue aircraft attrition. Subsequent examination of the CEM code supported this hypothesis.

The next section will provide several examples that highlight the affect of raid size and other mission allocation parameters on FEBA movement.

4. Output Measures That May Help Explain FEBA Movement

The FEBA movement calculations in CEM are "based on the loss rates of the two sides in individual brigade (or partial brigade) sectors, not on force ratio," according to CAA.

An interesting exercise, therefore, is to look at the attrition results of only one cycle of combat. In the following, the aircraft and tank attrition results for cycle 1 are discussed. Figures 7 through 12 of Annex B-1 show what these results look like.

As a first exercise towards obtaining an intuitive understanding of the FEBA movement results based on cycle 1 attrition, Figures 8, 10 and 12 of Annex B-1 are considered. If one examines only the light surfaces in these three figures, the following observations can be made. The light surfaces correspond to the heaviest emphasis on the CAS mission (initial allocation to CAS = .7, and raid size = 12). Figure 8 shows that aircraft attrition increases along with both the inventory and the minimum fraction of CAS allocation that remain on CAS. The monotonicity of the aircraft attrition is related to greater aircraft activity. This greater aircraft activity is evidenced in Figures 10 and 12 also. In Figure 12, the light surface shows that Red tank attrition increases in the same manner as Blue aircraft activity. Due to increased Red tank attrition, one expects, and observes in Figure 10, decreased Blue tank attrition (as depicted by the light surface). This basic relationship among aircraft and tank attrition, as seen in these three surfaces, may in fact be the the main predictor of the corresponding FEBA movement surface (the light surface) in Figure 4.

For Figures 7, 9 and 11, the initial allocation of tactical fighters to CAS is .2. Thus there are potentially many fewer CAS sorties flown, compared to the cases discussed above. When the above reasoning is used for these cases, the following may be observed.

First, there is no difference in aircraft attrition during cycle 1 for the two raid size values (cf. the light and dark surfaces in Figure 7). As discussed, the model enforces an upper bound or constraint on the number of CAS sorties per time period. When there are fewer aircraft initially assigned to CAS, as is the case here, this bound may not be reached even when raid size decreases. Given no significant difference in aircraft activity, therefore, one next observes no significant difference in Blue tank attrition, in Figure 9. What is harder to explain, therefore, is the difference in Red tank attrition in Figure 11 for the cases where 300 aircraft are added. While these cases are good candidates for further examination of other output measures and the model itself, not enough data are currently available to relate these results to FEBA movement.

The next section examines some attrition results from the perspective of the latter period of combat, past day 20.

5. Aircraft Activity Past Day 20

The main observation of this paper, that the aircraft contribution to FEBA movement essentially ends by day 20, requires a subsequent attempt to determine what aircraft related activity occurred past this day. In particular, in the cases where 2500 additional aircraft were added, upwards of 2000 remain at the end of cycle 5. Since FEBA movement was not significantly affected by these greater numbers of aircraft remaining, and since by cycle 15 these additional aircraft had been mostly attrited, the relative worth of these aircraft must be assessed.

One conjecture is that the additional aircraft continued to perform attack missions. Because they do continue to suffer attrition, this is likely the case. What is not discernable, though, is the degree of their success, measured in terms of weapons killed. Had a measure of cumulative kills been developed, one could better investigate this idea. An approximation to cumulative tank kills was developed by IDA. Measure 14 and the other cumulative measures should not be viewed as accurate for the later time periods, though, since the linear interpolation is based on too few data points.

A final observation on aircraft attrition is that the overall rate of attrition on the BAI missions is greater than that on the CAS missions. This may be observed by comparing the numbers of attrited aircraft between, say, cycles 5 and 15, with the CASIPS input. Recall that this input governs the degree to which aircraft may be assigned from CAS to BAI missions.

6. Anomalies in the Data

A worthwhile step towards understanding the results calculated by CEM involves studying key instances where seemingly non-intuitive results occur. In general, these instances look at variations of a single input, and determine whether 'more' of this input produced a more favorable result for Blue.

As was done in Section 4, Figures 7 through 12 of Annex B-1 will be used to comment on the partial derivative, in essence, of certain measures with respect to variations in the inputs studied here. Only a few of those cases that deserve further exploration will be mentioned here.

1. In Figures 11 and 12, regarding the light surfaces, when the minimum fraction of the initial CAS allocation that remains on CAS is .1, the Red tank attrition for cycle 1 decreases as Blue has increased inventories of aircraft.

2. In Figure 11, the case where 300 aircraft are added, the minimum fraction of the initial CAS allocation that remains on CAS is .9 and the raid size is 12 seems at odds with the neighboring cases on the same surface.

3. As a general comment, one would expect that in many measures, as one varies the added aircraft from 300 to 1000 to 2500, holding all other inputs fixed, that the results would be increasingly favorable to Blue. Looking at the tabular data in Annex B-2 reveals many instances where this is not the case.

In attempting to understand the above effects, a useful technique may require developing some new measure that does behave in the expected manner. For example, it is reasonable that changes in aircraft inventories may cause the mission allocation or targeting of weapons to change. Thus, while tank attrition by itself may not increase with increasing aircraft, the sum of all attrition to heavy armor targets may show this desired relationship.

Further examination of the above cases may also lead one to assess the relative effectiveness of aircraft on various missions at various times of the combat. This, coupled with an understanding of what triggers changes in mission profiles may help explain some of the observed ground weapon attrition results.

For the current analyses, the most likely source for the anomalous behavior is a particular section of the CEM model's logic that limited the number of CAS/BAI raids in an army sector to less than 256. If the model attempted to execute this number or more, no sorties would occur for that time period. This limit is likely exceeded when the raid size is

small, when the number of added tactical aircraft is large and when the INITAFR input is high. When this occurs, fewer Blue sorties will execute resulting in fewer Red tanks being killed. In turn, the FEBA movement becomes more favorable to Red.

7. Contribution of BAI Kills

The main observation is that the BAI kills seemed to be independent of raid size. Second, in two cases (cases 30 and 33), the data was conjectured to be in error. The graphical representations of the BAI kills measure are provided in Figures 13 and 14 of Annex B-1.

The data available to IDA was examined to see if there were other indications that supported the CAA values. Nothing was found, apart from the two suspect data for the BAI kills measure, to support this presumption. Furthermore, as Annex B-2 indicates for measure 11, case 23 was reset to a more likely value.

A general question to ask, nevertheless, is whether the BAI kill affect FEBA movement. As the initial analysis showed, significant numbers of kills occur only in those cases where NOPLANS = 2500, and CASIPS = .1, i.e., in those cases where there are many aircraft and only a low fraction (.1) is forced to remain on the CAS mission. One may thus examine the FEBA movement results to see whether these cases are associated with better FEBA results for Blue.

No indication was found to indicate that increased BAI kills positively affects the FEBA movement. Moreover, the data on kills among ground weapons also does not directly indicate when the BAI kills occur. There are two possible explanations for these findings. First, the BAI kills may be occurring, mainly, during a time interval that the available data did not sample. Second, the BAI kills may be simply a calculated subset, or apportionment, of the kills accounted for by the model's main attrition calculations. If this is the case, then one would not expect to see BAI kills correlate with much other that the availability of aircraft to perform the BAI mission.

D. CONCLUSIONS

The primary conclusion that one may draw from the CAA analysis and from this discussion of the CEM results is that the CAS mission may contribute measurably to the combat results, as measured by mean FEBA movement, but that this contribution occurs within a relatively early portion of the conflict. While the greatest improvement

corresponded to those cases where 2500 additional aircraft had been added, all of these additional aircraft did not necessarily contribute to these early results. Thus, it is possible that the same results could have been obtained with many fewer aircraft.

Related to the above observation is the point that many fewer cycles of CEM or other models may have to be run in order to make relative assessments among options dealing with aircraft assigned to attack missions. Until the running time of the major combat models is no longer a constraining factor to thorough analyses, an approach that designs more but shorter runs of a model may yield significantly more insight.

A variety of key measures should be examined automatically and on a routine basis from each run of the model. Among those to consider are measures of cumulative effects (e.g., kills, sorties, rounds fired, etc.), measures of total resources remaining and measures of the various factors that comprise the FEBA movement calculation. Doing so may facilitate testing the validity of new input regions or conducting a comparison of a series of model runs.

As was done for this series of runs, controlled variations of key inputs should continue to be investigated. This discipline has a two-fold benefit. First, it helps uncover the mechanisms underlying any logical anomalies (e.g., more resources cause one side to do worse) by forcing the analyst to develop and examine the key explanatory output measures. Second, it allows one to more accurately assess the level of input variation at which there is no longer any increased benefit. Such questions of 'balance' occur not only with resource levels but also with parameters that govern how existing forces are allocated across a variety of missions.

ANNEX B-1

GRAPHICAL REPRESENTATION OF

SELECTED CEM OUTPUT

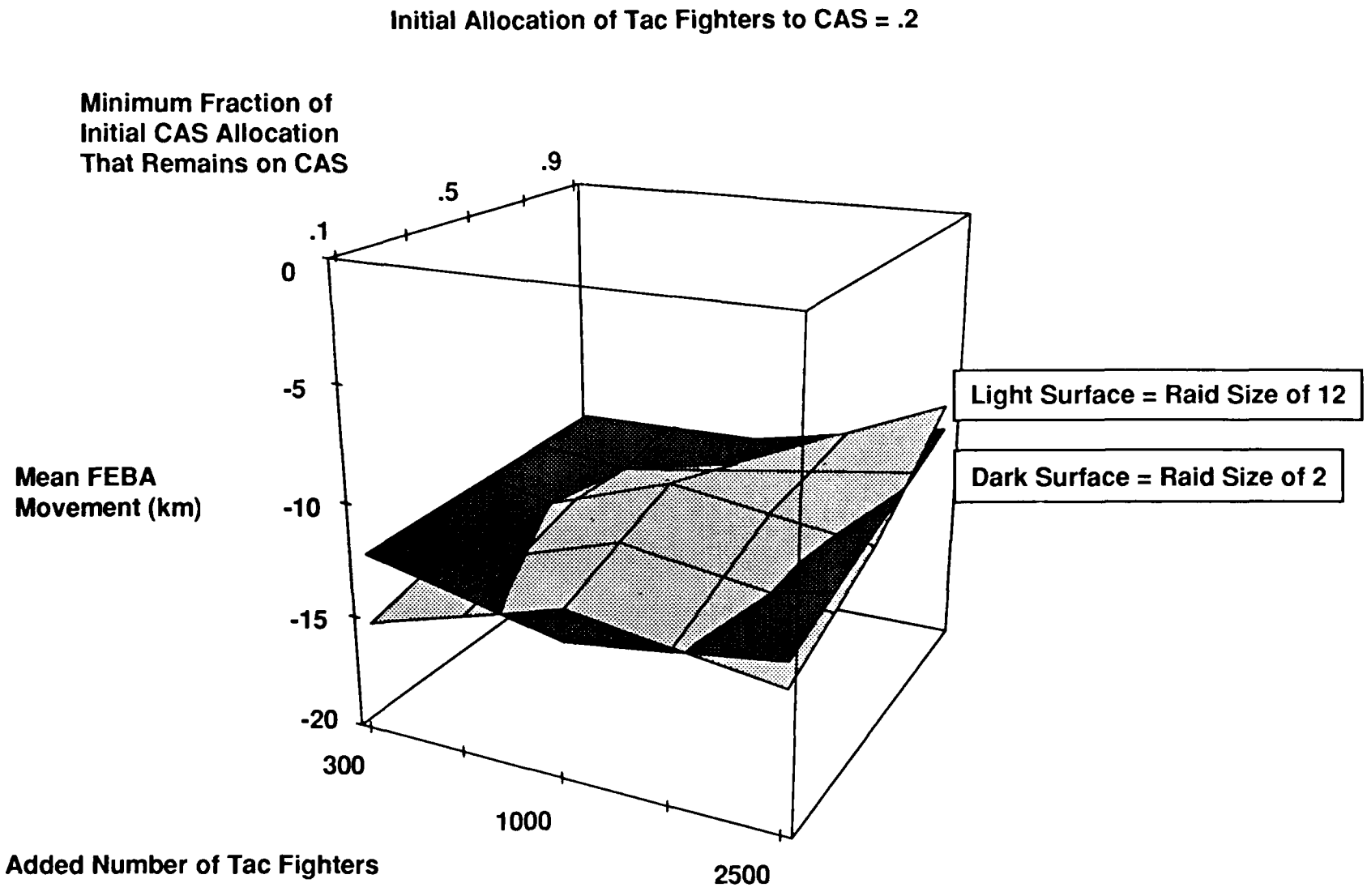


Figure B-1-1. Mean FEBA Movement as of Day 20, INITAFR = .2

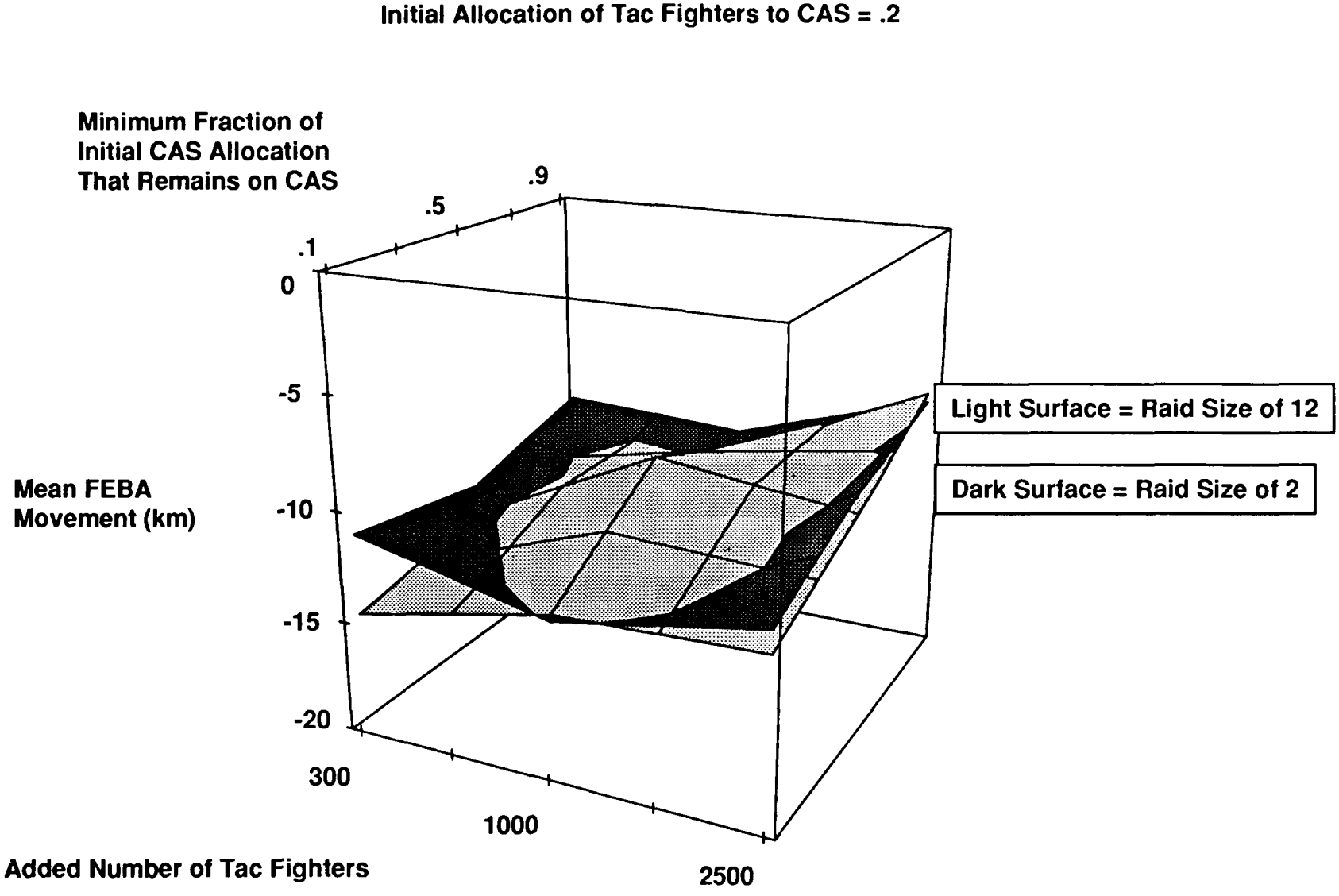


Figure B-1-2. Mean FEBA Movement as of Day 32, INITAFR = .2

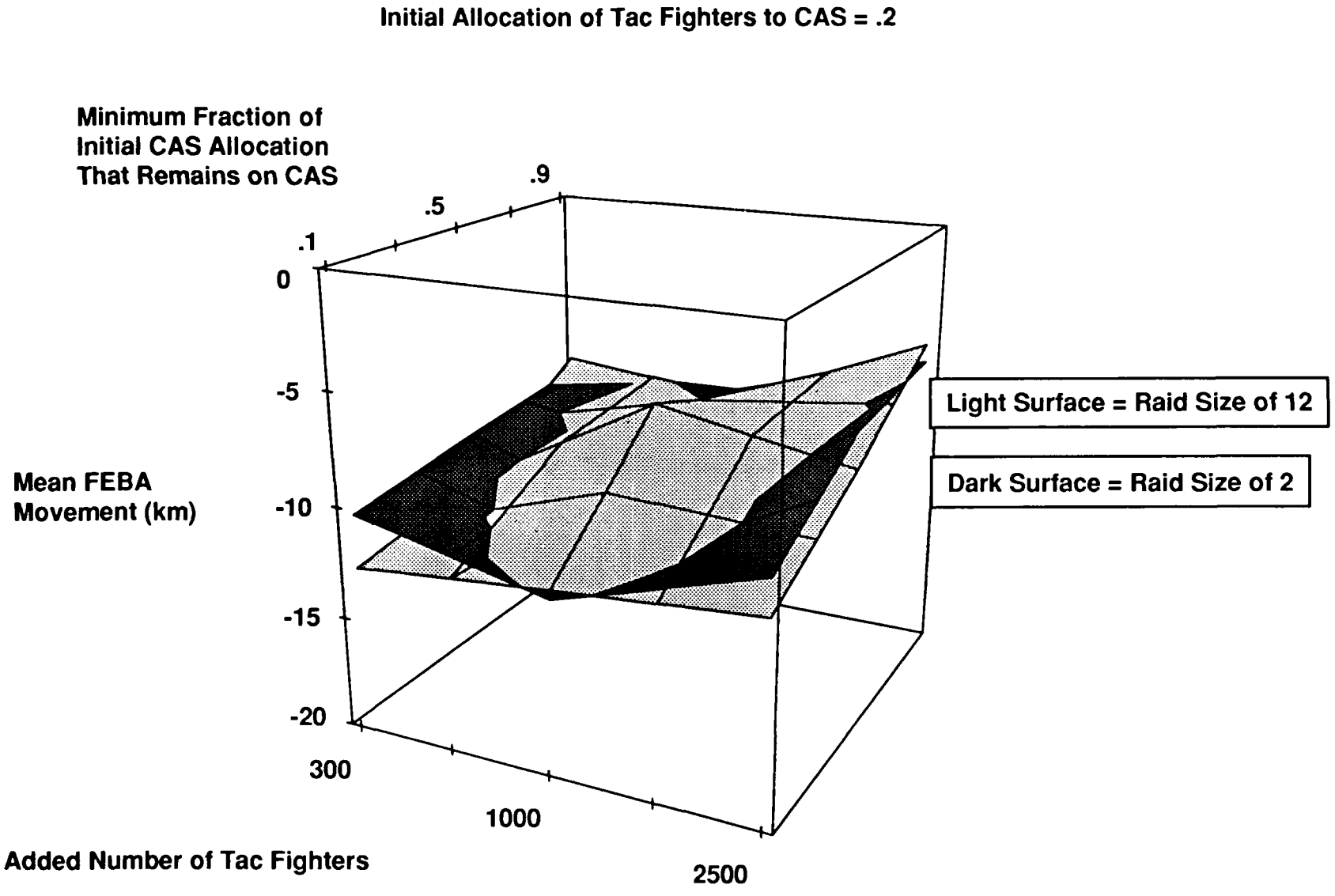


Figure B-1-3. Mean FEBA Movement as of Day 60, INITAFR = .2

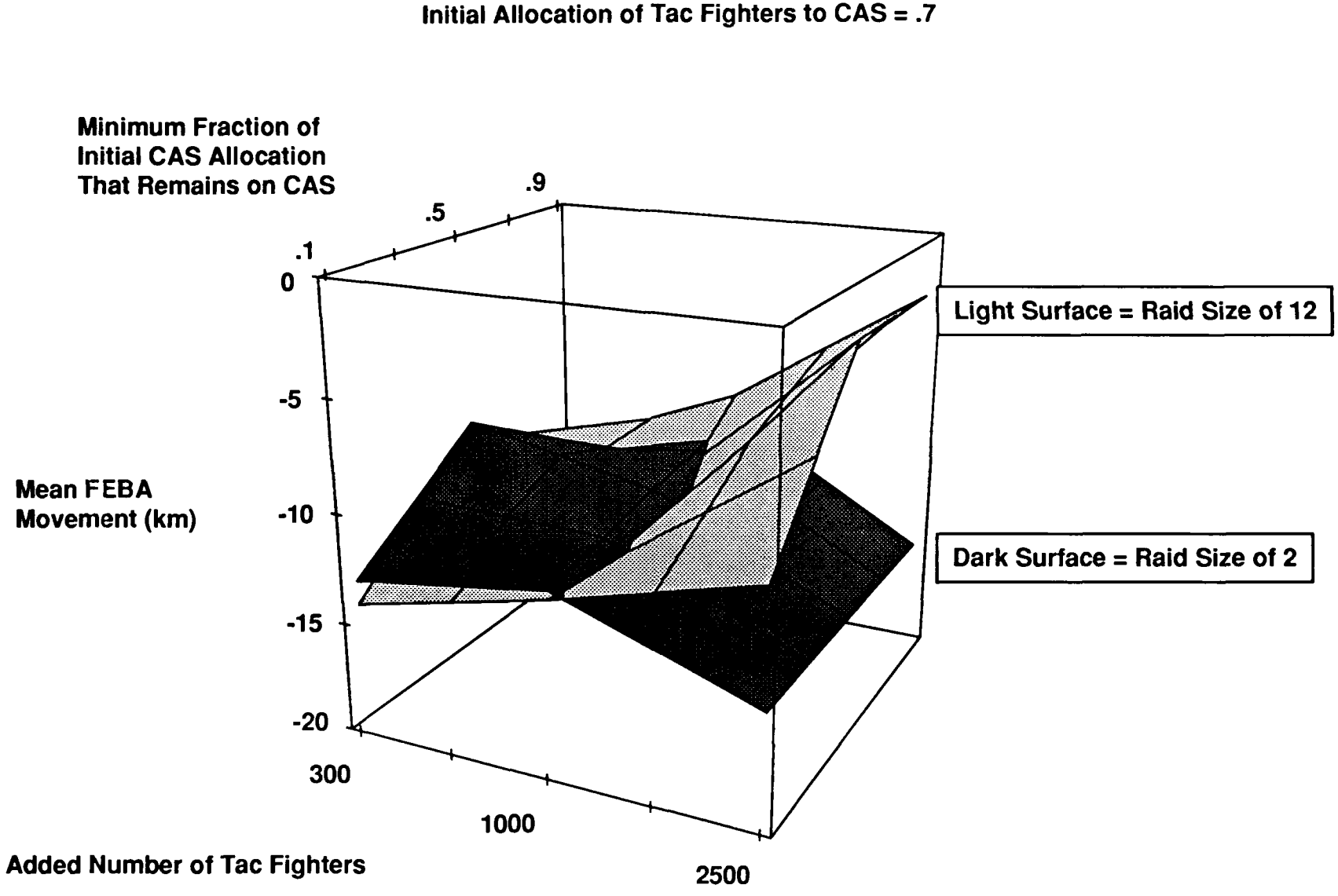


Figure B-1-4. Mean FEBA Movement as of Day 20, INITAFR = .7

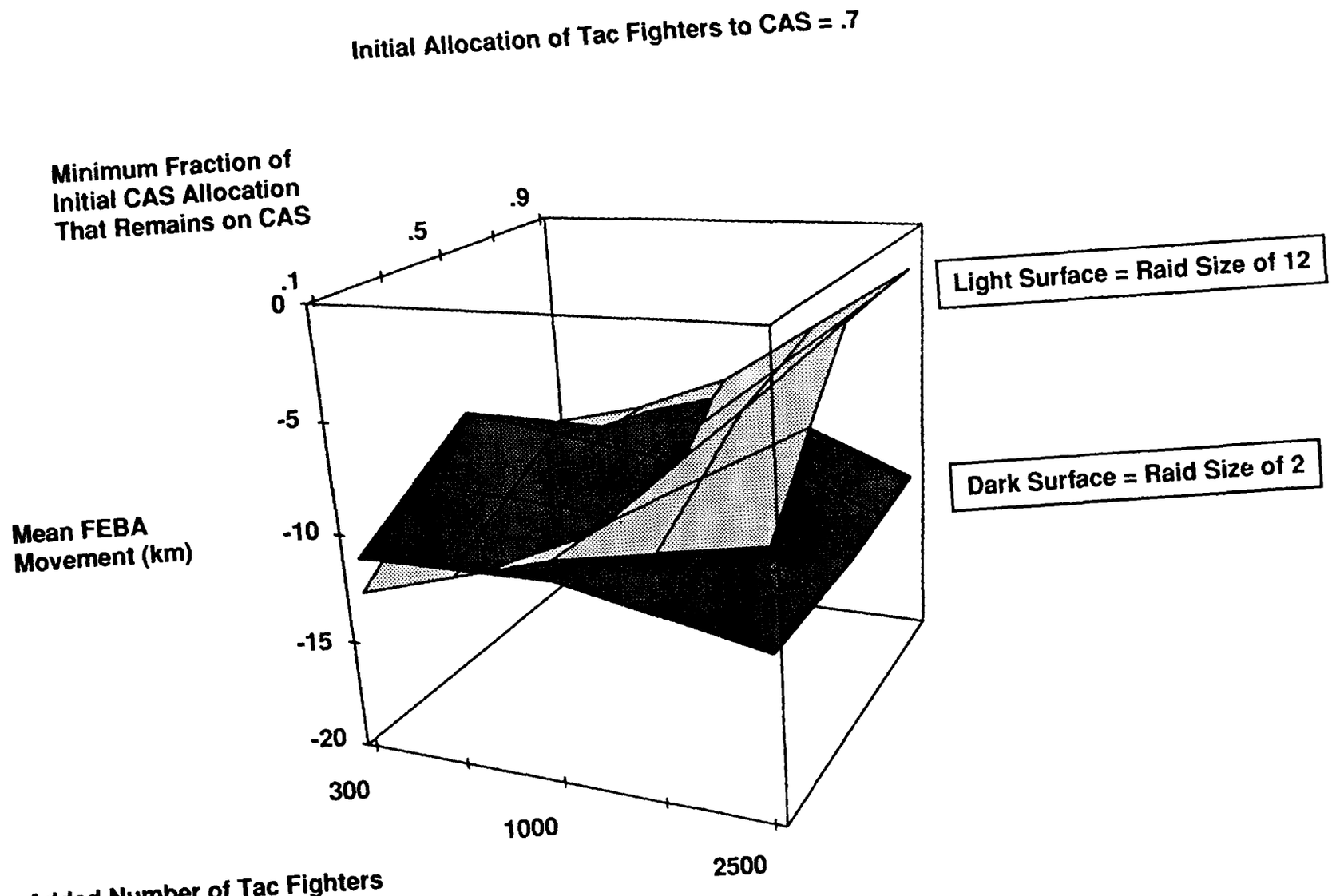


Figure B-1-5. Mean FEBA Movement as of Day 32, INITAFR = .7

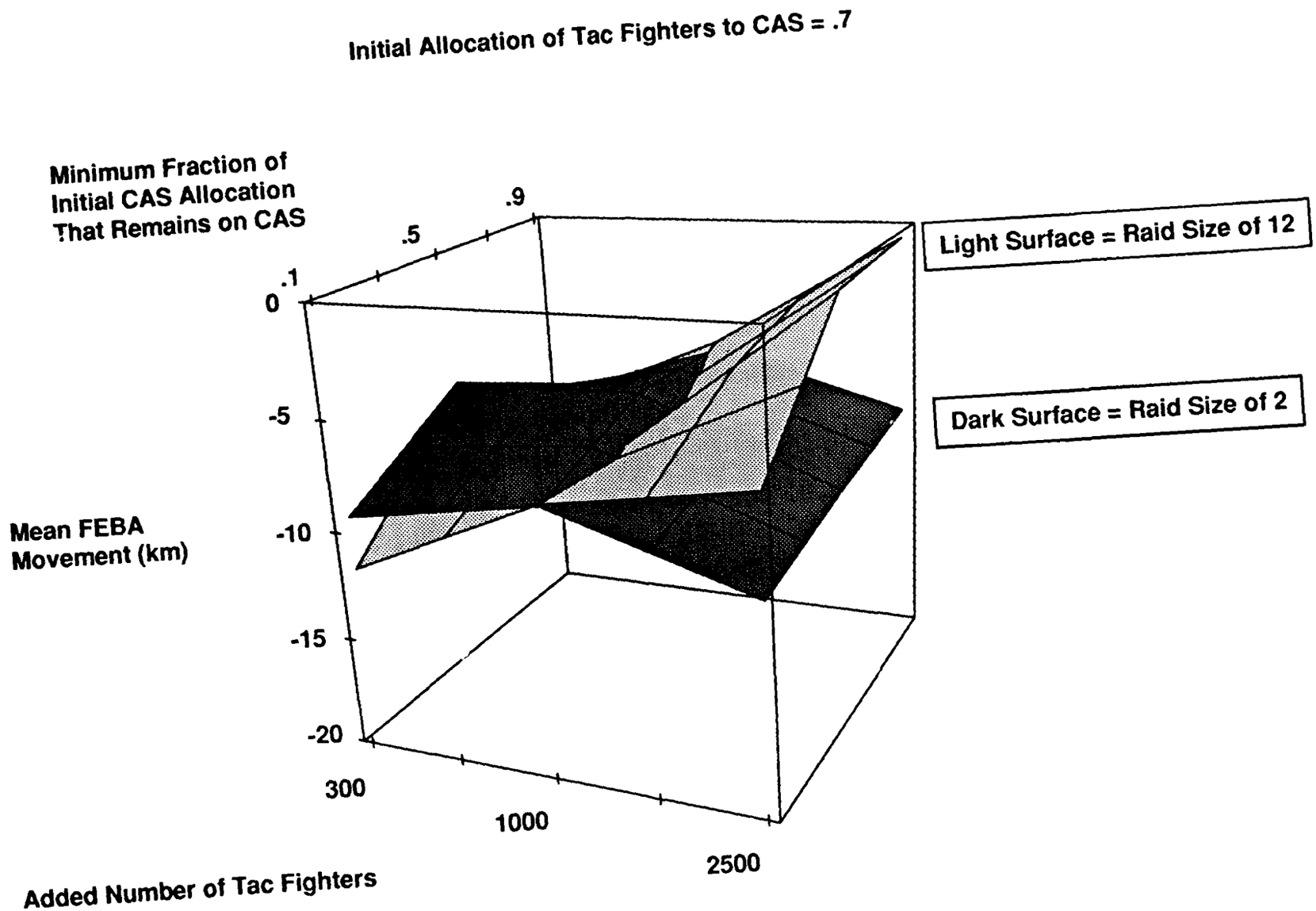


Figure B-1-6. Mean FEBA Movement as of Day 60, INITAFR = .7

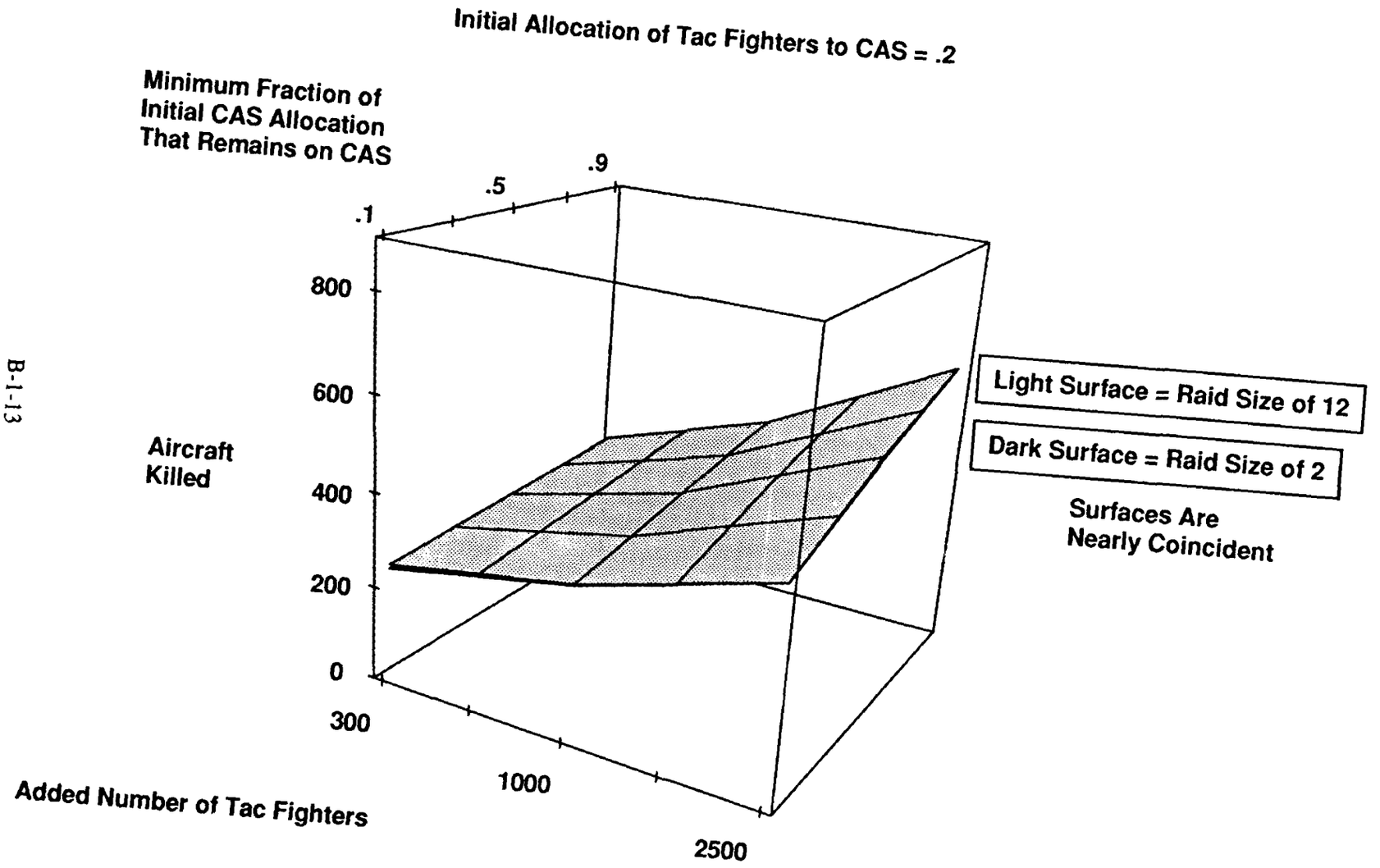


Figure B-1-7. Blue Aircraft Killed During Cycle 1, INITAFR = .2

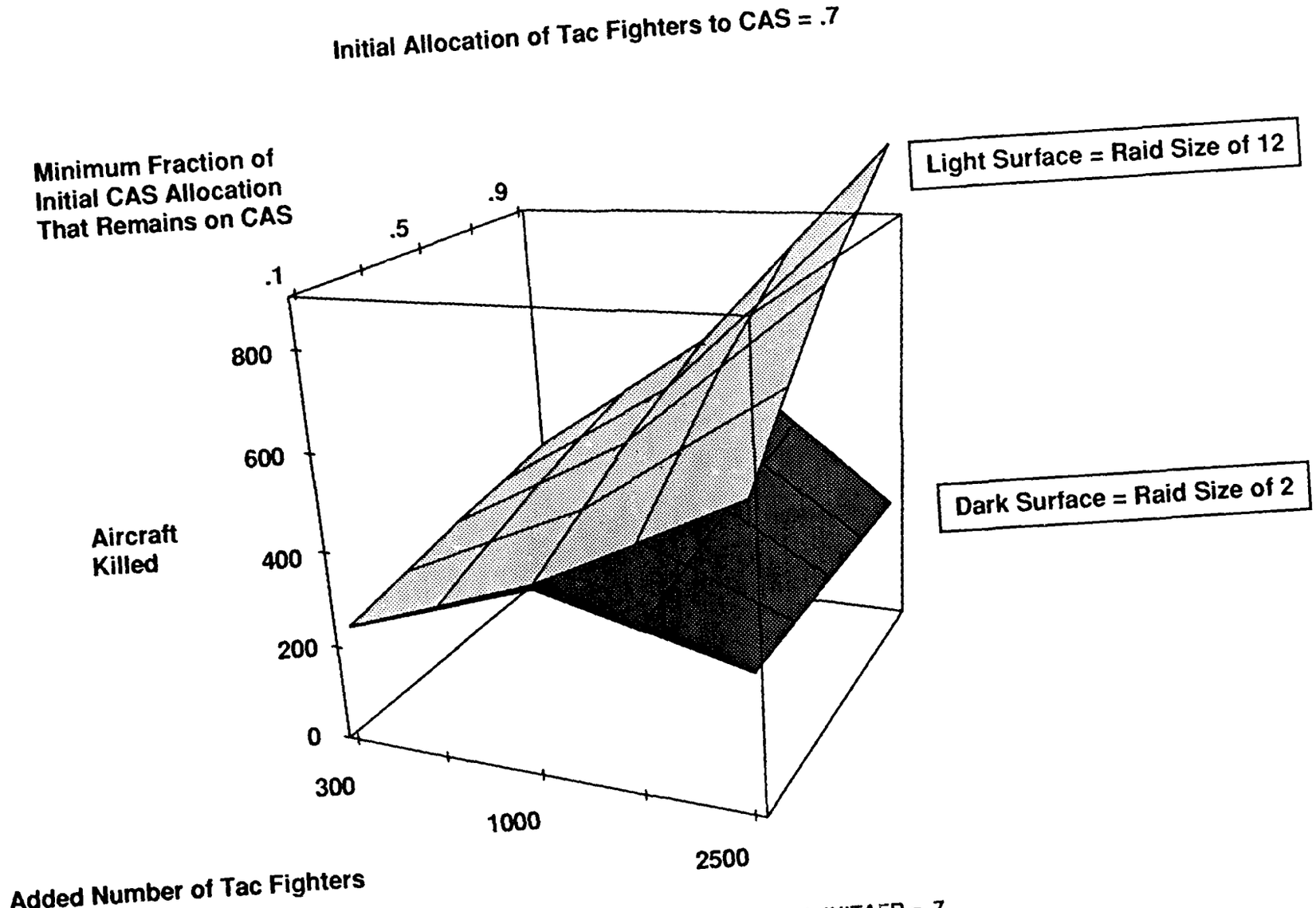


Figure B-1-8. Blue Aircraft Killed During Cycle 1, INITAFR = .7

Initial Allocation of Tac Fighters to CAS = .2

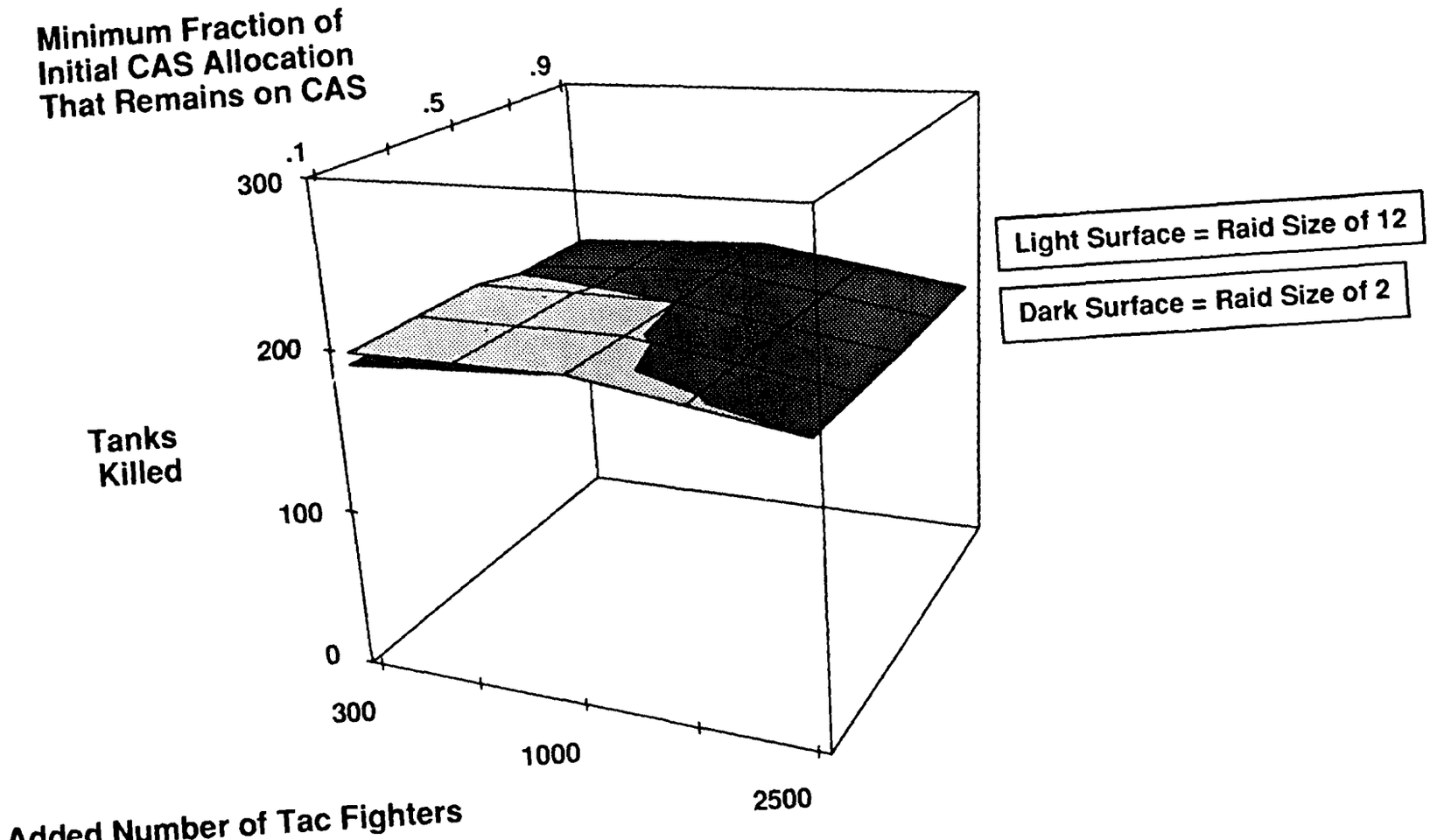


Figure B-1-9. Blue Tanks Killed During Cycle 1, INITAFR = .2

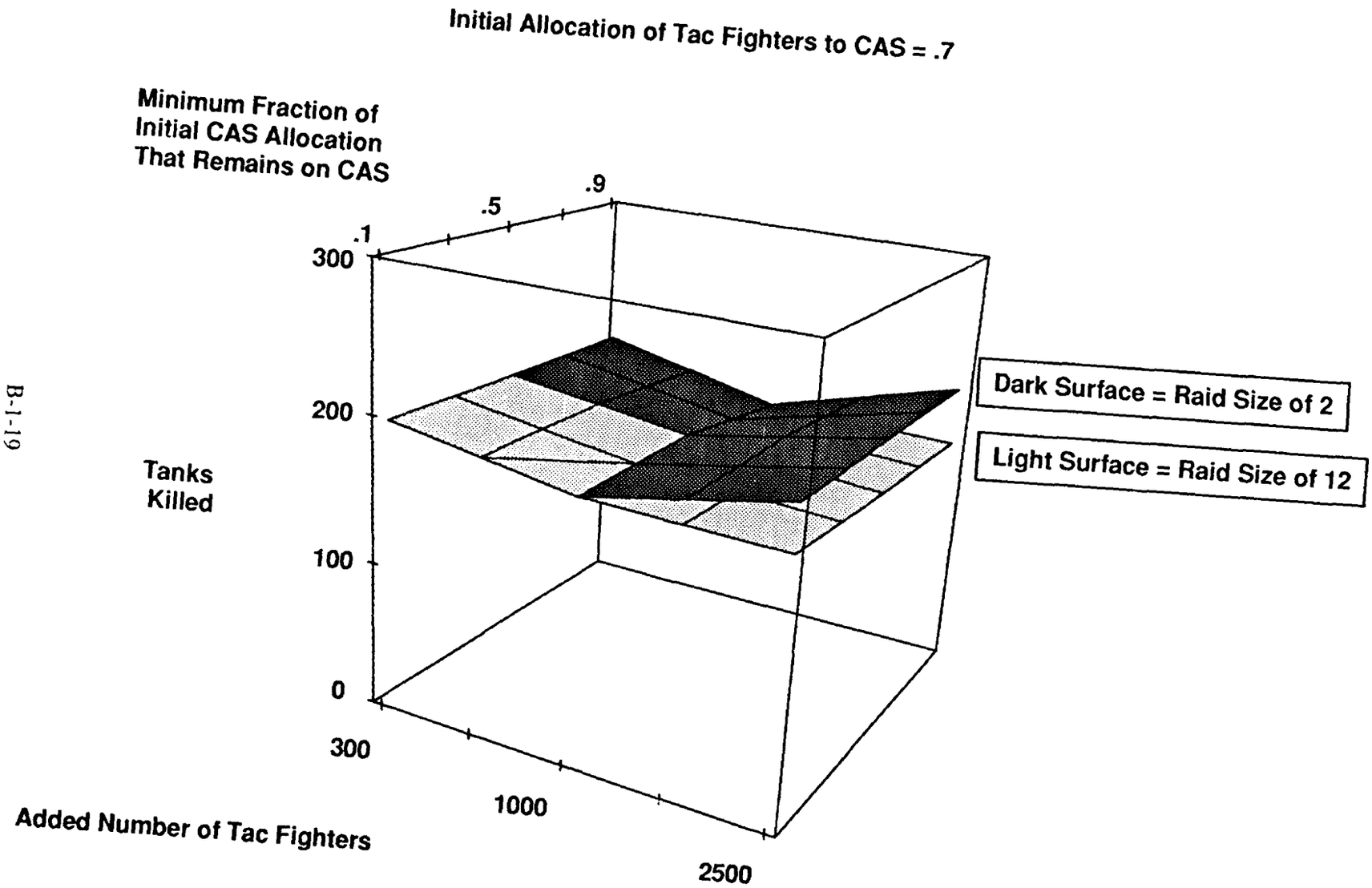


Figure B-1-10. Blue Tanks Killed During Cycle 1, INITAFR = .7

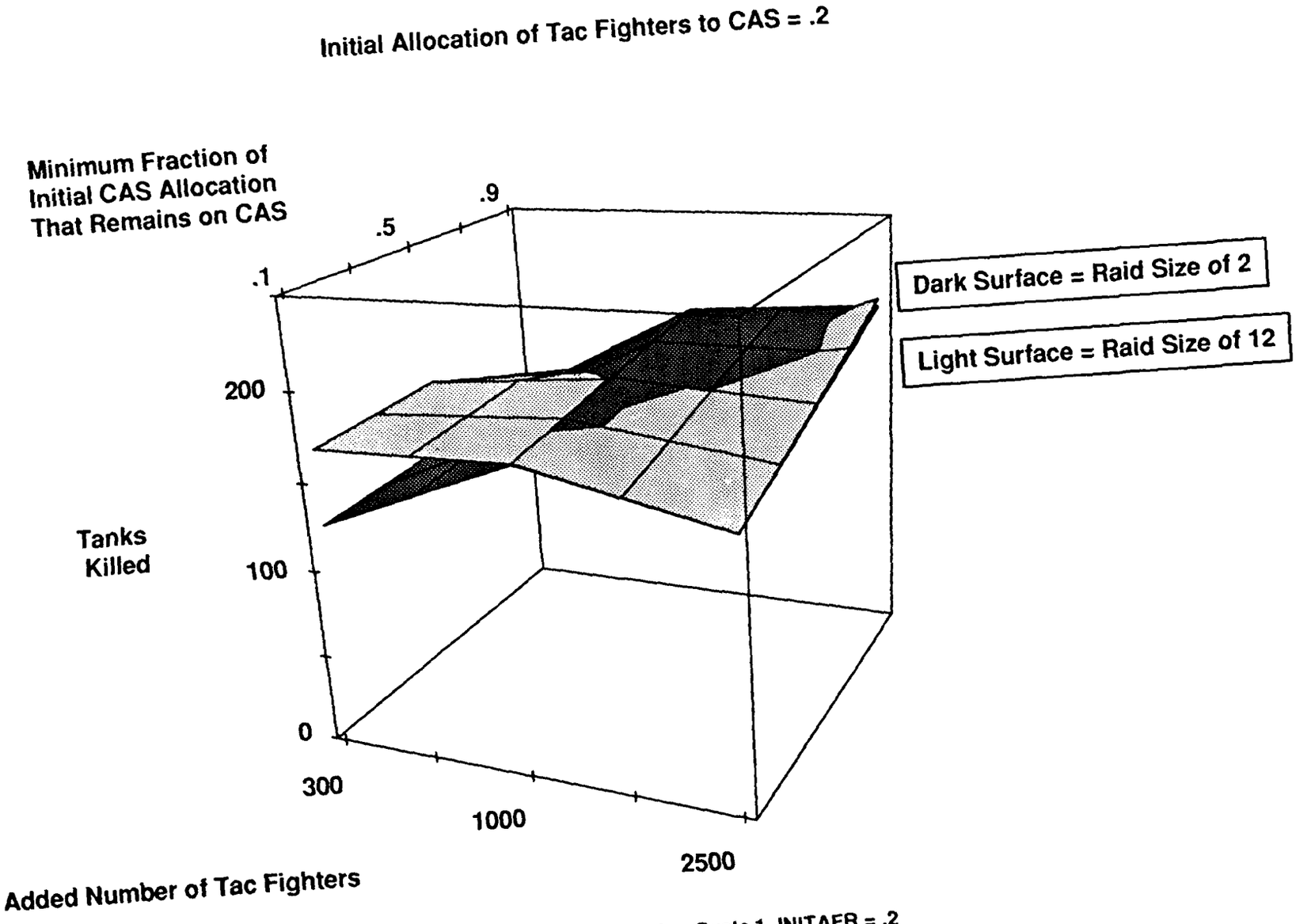


Figure B-1-11. Red Tanks Killed During Cycle 1, INITAFR = .2

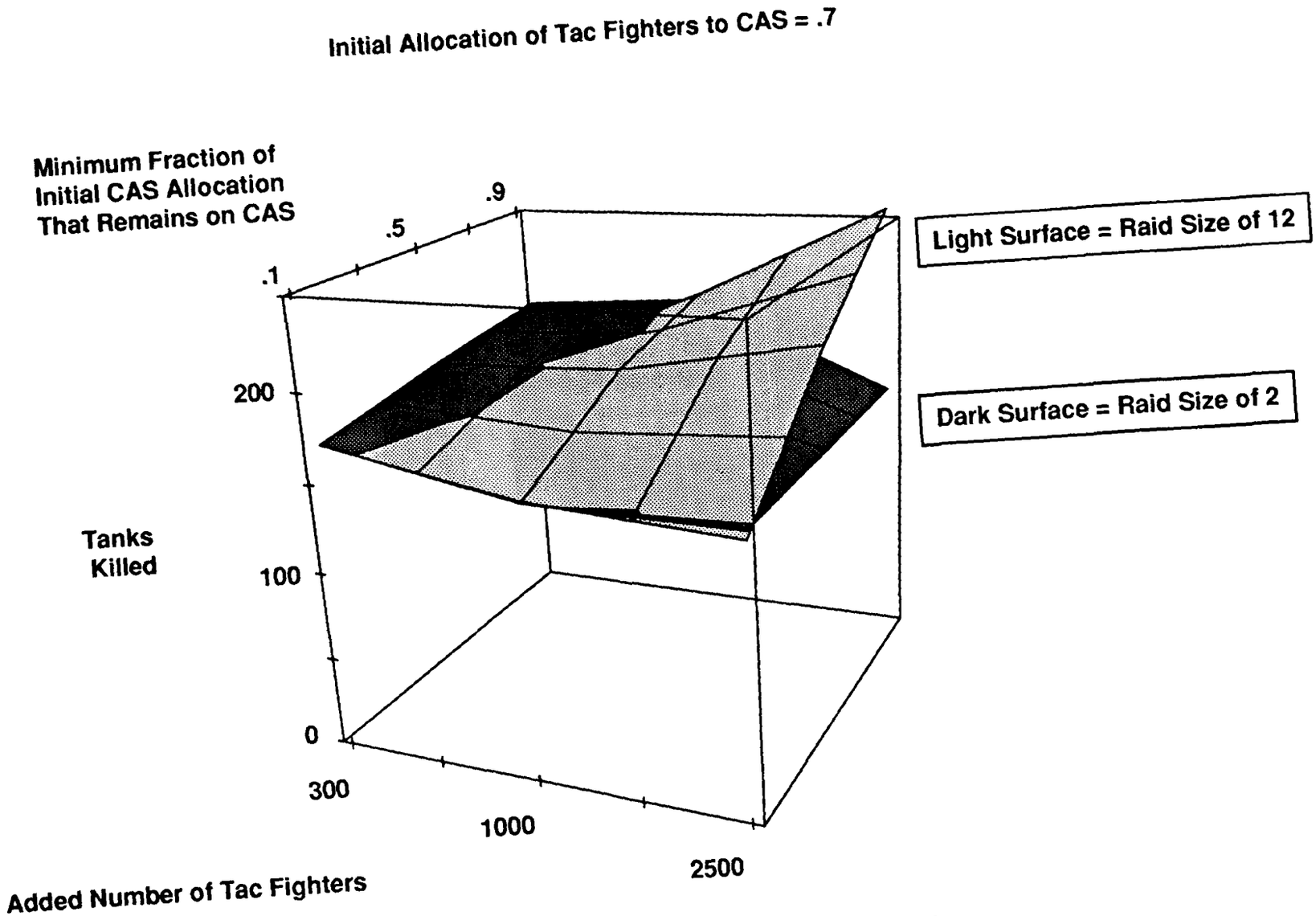


Figure B-1-12. Red Tanks Killed During Cycle 1, INITAFR = .7

- BAI Kills occur mainly when percent of CAS that are forced to remain on CAS is .1
- BAI Kills independent of raid size
- Initial allocation of Tac Fighters to CAS (INITAFR) is .2

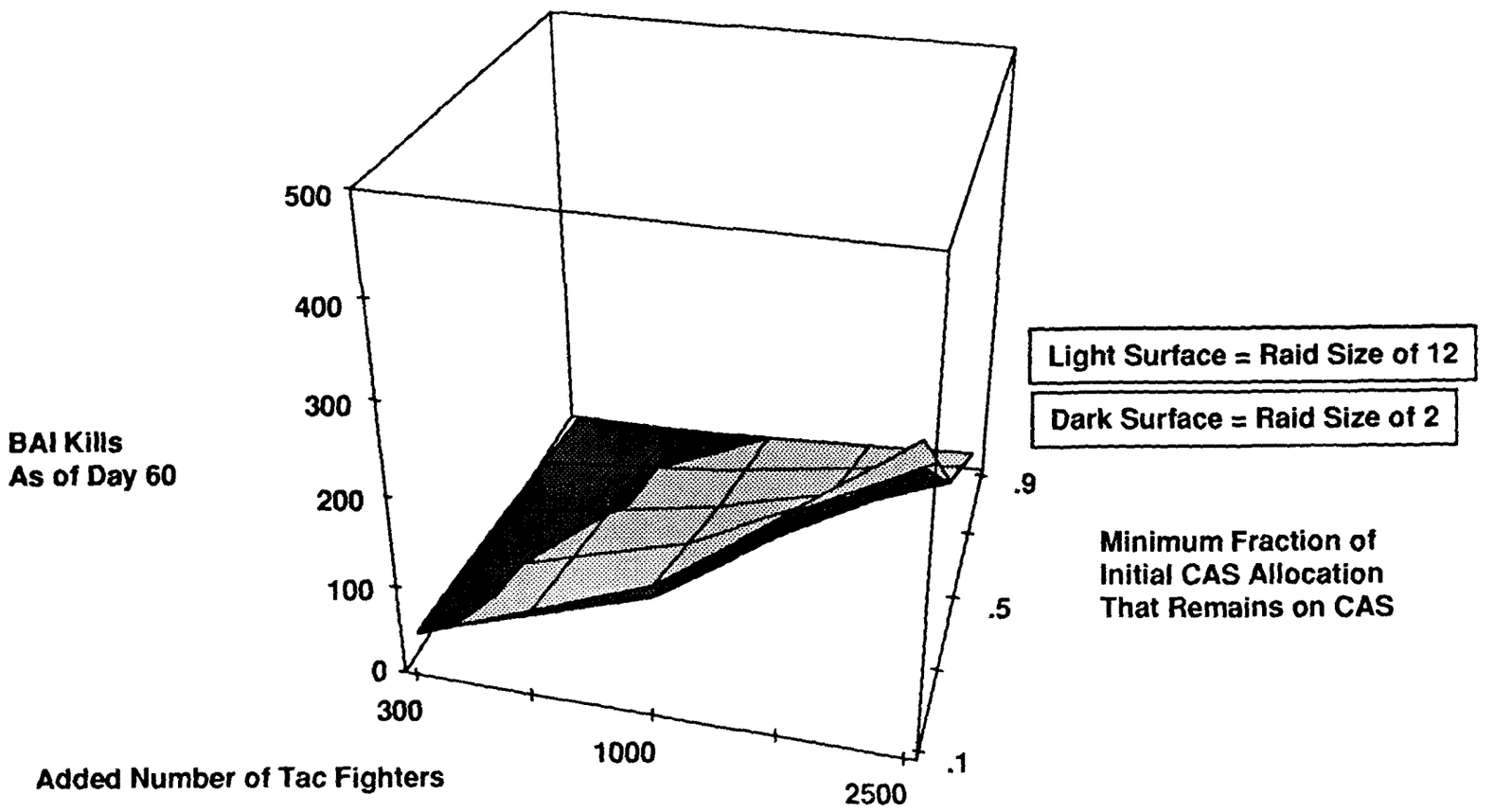


Figure B-1-13. BAI Kills as of Day 60, INITAFR = .2

- BAI Kills occur mainly when percent of CAS that are forced to remain on CAS is .1
- BAI Kills should be independent of raid size
- BAI Kills are greater when INITAFR = .7 (as in this figure) than when INITAFR = .2 (previous figure)
- Cases 30 and 33 Seem to be in Error

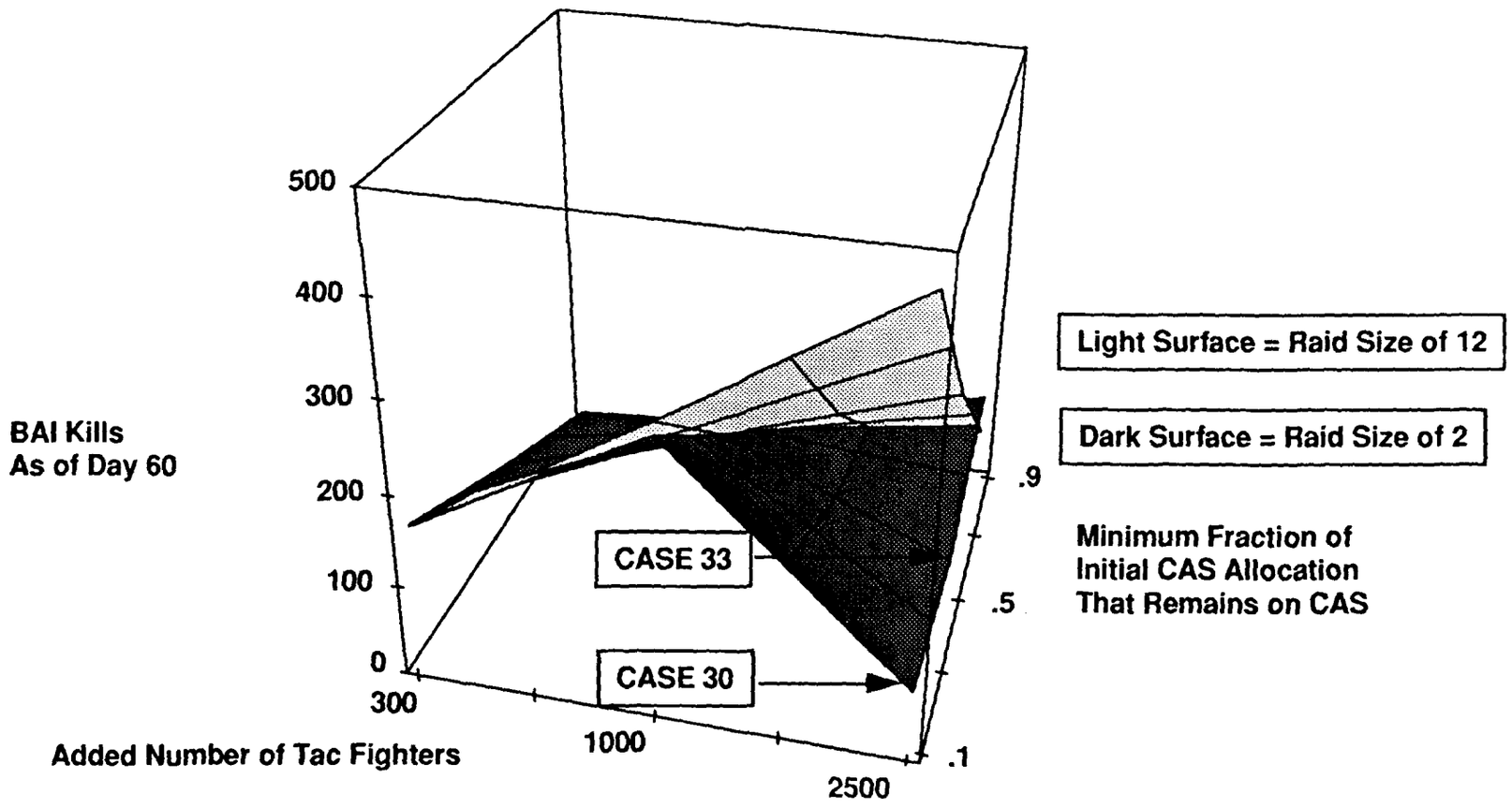


Figure B-1-14. BAI Kills as of Day 60, INITAFR = .7

ANNEX B-2

TABULAR REPRESENTATION OF

SELECTED CEM OUTPUT

Measure 1

Mean FEBA Movement -- by case number and time sample

```
{ 1, -1.2 , -4.7  , -11.4 , -15.4 , -14.7 , -12.8 }
{ 2, -1.2 , -4.2  , - 9.1 , -12.9 , -12.9 , -12.0 }
{ 3, -1.0 , -5.3  , -10.5 , -14.2 , -12.6 , -11.3 }
{ 4, -1.1 , -4.3  , - 9.9 , -13.8 , -13.4 , -11.8 }
{ 5, -1.1 , -3.8  , - 8.5 , -10.8 , - 9.2 , - 6.9 }
{ 6, -1.0 , -5.2  , - 9.0 , -12.1 , -10.3 , - 8.5 }
{ 7, -1.0 , -5.4  , - 9.8 , -12.2 , -10.8 , - 8.2 }
{ 8, -1.0 , -4.2  , - 8.8 , -12.6 , -11.5 , - 9.1 }
{ 9, -1.0 , -5.0  , - 8.6 , - 8.7 , - 7.6 , - 5.4 }
{10, -1.3 , -5.3  , - 7.4 , -12.3 , -11.1 , -10.4 }
{11, -1.2 , -4.2  , -10.5 , -14.3 , -13.2 , -12.4 }
{12, -1.0 , -5.3  , - 9.7 , -13.1 , -11.6 , - 9.7 }
{13, -1.3 , -6.5  , - 8.5 , -12.0 , -12.0 , - 9.6 }
{14, -1.1 , -4.1  ,  -9.7 , -14.0 , -15.2 , -13.0 }
{15, -1.0 , -4.9  , - 7.6 , -11.4 , - 9.3 , - 6.8 }
{16, -1.3 , -6.2  , - 8.5 , -11.7 , -10.2 , - 8.9 }
{17, -1.1 , -4.5  , -10.4 , -11.6 , -10.7 , - 8.7 }
{18, -1.0 , -5.0  , - 9.7 , - 9.8 , - 8.0 , - 6.2 }
{19, -1.1 , -4.1  , -11.9 , -14.2 , -12.8 , -11.8 }
{20, -1.0 , -5.2  , -10.7 , -12.2 , -10.2 , - 7.9 }
{21, -1.0 , -4.7  , - 6.9 , - 9.8 , - 8.5 , - 6.4 }
{22, -1.1 , -4.1  , - 8.0 , - 8.9 , - 8.0 , - 6.3 }
{23, -1.0 , -5.1  , - 8.6 , -12.6 , -10.6 , - 8.0 }
{24, -0.9 , -2.1  , - 4.8 , - 2.7 , - 2.2 , - 1.0 }
{25, -1.0 , -5.1  , - 9.9 , -12.3 , -11.4 , - 9.9 }
{26, -0.9 , -5.3  , - 8.8 , - 8.7 , - 7.9 , - 6.0 }
{27, -0.8 , -1.6  , - 3.2 , - 2.8 , - 2.0 , - 0.6 }
{28, -1.1 , -4.7  , -12.2 , -13.1 , -11.2 , - 9.4 }
{29, -1.0 , -5.2  , -10.7 , -11.8 , -11.1 , - 8.0 }
{30, -1.3 , -6.0  , - 8.4 , -15.0 , -12.9 , -10.8 }
{31, -1.1 , -4.1  , - 8.2 , - 8.7 , - 7.5 , - 6.1 }
{32, -1.0 , -5.1  , - 8.2 , -10.6 , - 8.8 , - 6.0 }
{33, -1.3 , -6.0  , - 8.4 , -15.0 , -12.9 , - 9.7 }
{34, -1.1 , -4.3  , - 9.7 , -13.7 , -12.4 , -10.5 }
{35, -1.0 , -5.3  , - 8.9 , - 9.8 , - 8.3 , - 6.3 }
{36, -1.3 , -6.0  , - 8.4 , -15.0 , -12.2 , - 8.9 }
```

Measure 2

Blue Aircraft Remaining at Start of Cycle -- by case number and time sample

{ 1,	1301 ,	1054 ,	994 ,	815 ,	650 ,	580 }
{ 2,	2001 ,	1681 ,	1568 ,	1253 ,	853 ,	593 }
{ 3,	3501 ,	3065 ,	2895 ,	2467 ,	1883 ,	978 }
{ 4,	1301 ,	1033 ,	960 ,	786 ,	641 ,	580 }
{ 5,	2001 ,	1630 ,	1479 ,	1121 ,	754 ,	585 }
{ 6,	3501 ,	2956 ,	2707 ,	2164 ,	1519 ,	711 }
{ 7,	1301 ,	1013 ,	935 ,	763 ,	635 ,	581 }
{ 8,	2001 ,	1585 ,	1406 ,	1029 ,	696 ,	579 }
{ 9,	3501 ,	2874 ,	2569 ,	1954 ,	1283 ,	629 }
{10,	1301 ,	1063 ,	1008 ,	830 ,	653 ,	578 }
{11,	2001 ,	1684 ,	1577 ,	1257 ,	855 ,	592 }
{12,	3501 ,	3066 ,	1898[1],	2467 ,	1883 ,	980 }
{13,	1301 ,	1043 ,	973 ,	792 ,	635 ,	576 }
{14,	2001 ,	1635 ,	1492 ,	1134 ,	765 ,	591 }
{15,	3501 ,	2692[2],	2714 ,	2176 ,	1526 ,	705 }
{16,	1301 ,	1026 ,	946 ,	763 ,	626 ,	574 }
{17,	2001 ,	1595 ,	1423 ,	1043 ,	710 ,	584 }
{18,	3501 ,	2875 ,	2576 ,	1958 ,	1297 ,	633 }
{19,	1301 ,	1063 ,	998 ,	867 ,	723 ,	606 }
{20,	2001 ,	1622 ,	1485 ,	1205 ,	899 ,	647 }
{21,	3501 ,	2916 ,	2682 ,	2177 ,	1616 ,	907 }
{22,	1301 ,	1005 ,	911 ,	771 ,	653 ,	588 }
{23,	2001 ,	1488 ,	1279 ,	961 ,	726 ,	589 }
{24,	3501 ,	2651 ,	2247 ,	1607 ,	1073 ,	647 }
{25,	1301 ,	959 ,	857 ,	723 ,	625 ,	578 }
{26,	2001 ,	1386 ,	1144 ,	832 ,	649 ,	576 }
{27,	3501 ,	2455 ,	1961 ,	1296 ,	850 ,	590 }
{28,	1301 ,	1065 ,	1004 ,	877 ,	724 ,	609 }
{29,	2001 ,	1631 ,	1491 ,	1204 ,	901 ,	649 }
{30,	3501 ,	3234 ,	3165 ,	2874 ,	2288 ,	1201 }
{31,	1301 ,	1007 ,	919 ,	777 ,	655 ,	582 }
{32,	2001 ,	1495 ,	1289 ,	964 ,	727 ,	591 }
{33,	3501 ,	3234 ,	3165 ,	2874 ,	2155 ,	905 }
{34,	1301 ,	964 ,	865 ,	724 ,	621 ,	573 }
{35,	2001 ,	1395 ,	1153 ,	844 ,	657 ,	575 }
{36,	3501 ,	3234 ,	3165 ,	2874 ,	2061 ,	764 }

[1] This value should be 2827, based on numbers of aircraft killed in measure 3.

[2] This value should be 2962, based on numbers of aircraft killed in measure 3.

Measure 3

Blue Aircraft Killed During Cycle -- by case number and time sample

```
{ 1, 247 , 131 , 140 ,  83 ,  27 ,   1 }
{ 2, 320 , 184 , 205 , 163 ,  89 ,   5 }
{ 3, 436 , 241 , 264 , 221 , 165 ,  85 }
{ 4, 268 , 144 , 141 ,  75 ,  23 ,   1 }
{ 5, 371 , 222 , 237 , 158 ,  68 ,   2 }
{ 6, 545 , 320 , 340 , 250 , 172 ,  44 }
{ 7, 288 , 149 , 147 ,  68 ,  21 ,   1 }
{ 8, 416 , 250 , 254 , 159 ,  50 ,   2 }
{ 9, 627 , 376 , 377 , 260 , 179 ,  18 }
{10, 238 , 126 , 141 ,  86 ,  30 ,   1 }
{11, 317 , 178 , 206 , 166 ,  89 ,   5 }
{12, 435 , 239 , 269 , 221 , 164 ,  84 }
{13, 258 , 140 , 146 ,  80 ,  25 ,   1 }
{14, 366 , 214 , 237 , 162 ,  69 ,   2 }
{15, 539 , 319 , 337 , 250 , 174 ,  44 }
{16, 275 , 151 , 152 ,  70 ,  23 ,   1 }
{17, 406 , 242 , 256 , 154 ,  52 ,   1 }
{18, 626 , 370 , 381 , 259 , 163 ,  21 }
{19, 238 , 136 , 112 ,  69 ,  31 ,   5 }
{20, 379 , 207 , 203 , 131 ,  68 ,  14 }
{21, 585 , 306 , 330 , 226 , 144 ,  55 }
{22, 296 , 164 , 127 ,  60 ,  26 ,   2 }
{23, 513 , 279 , 237 , 112 ,  41 ,   4 }
{24, 850 , 475 , 414 , 228 , 115 ,  18 }
{25, 342 , 172 , 127 ,  56 ,  21 ,   1 }
{26, 615 , 313 , 238 ,  91 ,  28 ,   2 }
{27,1046 , 565 , 446 , 211 ,  85 ,  10 }
{28, 236 , 132 , 110 ,  75 ,  31 ,   6 }
{29, 370 , 211 , 201 , 133 ,  64 ,  13 }
{30, 267 , 140 , 188 , 173 , 205 ,  90 }
{31, 294 , 160 , 130 ,  63 ,  24 ,   1 }
{32, 506 , 277 , 241 , 109 ,  42 ,   5 }
{33, 267 , 140 , 188 , 173 , 324 ,  68 }
{34, 337 , 171 , 133 ,  53 ,  21 ,   1 }
{35, 606 , 312 , 240 ,  94 ,  31 ,   1 }
{36, 267 , 140 , 188 , 173 , 390 ,  48 }
```

Measure 4

Blue Tank Losses During Cycle -- by case number and time sample

```
{ 1, 196.9 , 241.5 , 192.1 , 176.8 , 121.7 , 209.0 }
{ 2, 196.8 , 250.1 , 149.3 , 124.7 , 181.9 , 162.0 }
{ 3, 177.0 , 251.9 , 173.0 , 151.2 , 150.8 , 186.2 }
{ 4, 196.8 , 250.6 , 159.7 , 156.2 , 137.9 , 192.7 }
{ 5, 195.5 , 217.1 , 147.6 , 191.0 , 183.8 , 237.4 }
{ 6, 174.8 , 225.4 , 187.2 , 130.7 , 164.2 , 211.8 }
{ 7, 176.4 , 247.5 , 162.6 , 198.2 , 144.0 , 188.8 }
{ 8, 175.7 , 213.6 , 179.9 , 157.4 , 152.3 , 167.8 }
{ 9, 174.2 , 237.8 , 201.9 , 147.1 , 178.5 , 164.9 }
{10, 189.5 , 229.6 , 181.0 , 137.2 , 160.3 , 184.1 }
{11, 196.7 , 252.0 , 159.3 , 198.6 , 133.0 , 159.3 }
{12, 177.0 , 251.5 , 174.0 , 118.8 , 156.0 , 206.8 }
{13, 189.3 , 261.4 , 201.1 , 173.5 , 143.3 , 169.2 }
{14, 195.5 , 240.6 , 155.6 , 165.3 , 120.6 , 192.5 }
{15, 175.5 , 243.7 , 197.7 , 193.8 , 160.7 , 195.1 }
{16, 189.2 , 278.0 , 224.0 , 215.5 , 177.1 , 148.3 }
{17, 195.1 , 250.2 , 156.5 , 123.6 , 147.5 , 150.6 }
{18, 174.1 , 238.2 , 240.5 , 129.3 , 155.5 , 163.1 }
{19, 196.7 , 240.5 , 156.3 , 142.2 , 135.9 , 247.5 }
{20, 176.8 , 252.9 , 175.5 , 145.8 , 105.4 , 176.8 }
{21, 174.4 , 234.9 , 178.1 , 191.2 , 166.8 , 155.5 }
{22, 195.4 , 228.7 , 189.2 , 153.8 , 141.9 , 160.8 }
{23, 179.2 , 234.3 , 167.9 , 170.6 , 154.8 , 240.7 }
{24, 168.6 , 262.2 , 192.7 , 158.9 , 165.2 , 222.3 }
{25, 180.5 , 246.5 , 191.8 , 152.5 , 163.9 , 170.2 }
{26, 167.2 , 233.7 , 221.9 , 226.8 , 182.5 , 139.5 }
{27, 165.2 , 238.5 , 177.9 , 155.8 , 136.7 , 197.6 }
{28, 196.6 , 252.8 , 180.5 , 143.4 , 129.8 , 268.7 }
{29, 176.8 , 252.7 , 171.0 , 149.1 , 152.0 , 128.5 }
{30, 205.3 , 225.5 , 209.5 , 146.8 , 122.4 , 129.9 }
{31, 195.4 , 221.7 , 224.6 , 178.9 , 170.6 , 194.4 }
{32, 179.2 , 233.8 , 184.4 , 136.6 , 173.4 , 187.8 }
{33, 205.3 , 225.5 , 209.5 , 146.8 , 143.4 , 157.6 }
{34, 195.2 , 277.5 , 163.9 , 162.9 , 156.2 , 167.0 }
{35, 167.3 , 233.7 , 163.2 , 130.6 , 143.5 , 193.4 }
{36, 205.3 , 225.5 , 209.5 , 146.8 , 226.5 , 208.6 }
```

Measure 5

Red Tank Losses During Cycle -- by case number and time sample

{ 1,	167.3 ,	134.6 ,	111.4 ,	113.9 ,	126.8 ,	89.1 }
{ 2,	169.6 ,	139.6 ,	86.9 ,	65.5 ,	94.6 ,	75.6 }
{ 3,	146.2 ,	148.8 ,	99.1 ,	104.9 ,	74.0 ,	81.0 }
{ 4,	170.1 ,	140.3 ,	88.8 ,	68.5 ,	69.6 ,	108.6 }
{ 5,	179.6 ,	135.4 ,	81.3 ,	90.4 ,	113.8 ,	122.0 }
{ 6,	173.8 ,	144.9 ,	163.9 ,	84.6 ,	100.3 ,	79.4 }
{ 7,	142.5 ,	145.0 ,	91.5 ,	82.1 ,	102.2 ,	74.2 }
{ 8,	159.9 ,	138.0 ,	133.0 ,	90.0 ,	75.6 ,	101.9 }
{ 9,	202.4 ,	185.3 ,	144.5 ,	94.6 ,	105.1 ,	70.3 }
{10,	124.9 ,	172.6 ,	147.4 ,	69.2 ,	80.4 ,	77.7 }
{11,	169.1 ,	140.2 ,	86.9 ,	168.9 ,	68.2 ,	73.4 }
{12,	145.9 ,	147.8 ,	104.2 ,	73.6 ,	76.5 ,	83.5 }
{13,	128.9 ,	165.5 ,	117.2 ,	99.2 ,	72.9 ,	71.1 }
{14,	180.5 ,	171.1 ,	143.6 ,	161.8 ,	67.2 ,	114.1 }
{15,	172.3 ,	156.8 ,	208.1 ,	103.7 ,	103.5 ,	70.1 }
{16,	132.6 ,	200.2 ,	183.4 ,	164.9 ,	82.5 ,	68.0 }
{17,	191.0 ,	159.7 ,	127.8 ,	77.3 ,	103.5 ,	76.0 }
{18,	198.1 ,	183.5 ,	185.7 ,	89.4 ,	87.1 ,	77.5 }
{19,	170.5 ,	136.1 ,	102.5 ,	66.9 ,	69.1 ,	96.0 }
{20,	151.4 ,	149.0 ,	112.1 ,	80.3 ,	63.3 ,	77.3 }
{21,	145.1 ,	154.5 ,	140.9 ,	107.2 ,	86.3 ,	66.5 }
{22,	181.2 ,	159.2 ,	180.9 ,	82.0 ,	84.7 ,	80.2 }
{23,	186.5 ,	159.1 ,	122.0 ,	88.5 ,	78.3 ,	88.5 }
{24,	206.7 ,	234.3 ,	171.9 ,	100.3 ,	85.8 ,	79.0 }
{25,	160.3 ,	146.6 ,	126.5 ,	87.2 ,	81.0 ,	132.6 }
{26,	201.9 ,	164.6 ,	142.4 ,	105.9 ,	81.1 ,	65.7 }
{27,	255.5 ,	233.3 ,	152.0 ,	88.9 ,	77.3 ,	66.1 }
{28,	170.7 ,	148.9 ,	95.5 ,	88.2 ,	69.5 ,	98.5 }
{29,	150.5 ,	150.0 ,	110.0 ,	78.7 ,	90.7 ,	64.3 }
{30,	149.7 ,	167.6 ,	171.5 ,	70.7 ,	104.0 ,	73.7 }
{31,	182.7 ,	149.3 ,	195.3 ,	95.3 ,	84.0 ,	74.3 }
{32,	185.5 ,	158.3 ,	130.1 ,	73.7 ,	124.8 ,	80.1 }
{33,	149.7 ,	167.6 ,	171.5 ,	70.7 ,	147.2 ,	74.8 }
{34,	191.3 ,	177.5 ,	117.0 ,	75.2 ,	74.1 ,	77.9 }
{35,	198.7 ,	160.5 ,	122.7 ,	73.9 ,	71.1 ,	78.7 }
{36,	149.7 ,	167.6 ,	171.5 ,	70.7 ,	204.3 ,	73.4 }

Measure 6

Blue APC Losses During Cycle -- by case number and time sample

```
{ 1,   59.3 ,   71.0 ,   52.3 ,   84.8 ,   89.6 , 134.1 }
{ 2,   59.2 ,   71.0 ,   50.5 ,   67.1 ,   79.1 , 148.4 }
{ 3,   59.1 ,   69.3 ,   63.5 ,   73.5 ,   84.8 , 154.5 }
{ 4,   59.2 ,   70.1 ,   55.3 ,   83.6 ,   82.8 , 155.0 }
{ 5,   59.1 ,   70.1 ,   51.0 ,   82.4 ,  123.0 , 130.5 }
{ 6,   58.9 ,   68.4 ,   52.4 ,   68.2 ,  112.9 , 120 9 }
{ 7,   59.2 ,   62.9 ,   62.5 ,   80.8 ,   90.5 , 127.3 }
{ 8,   59.0 ,   69.1 ,   54.3 ,   76.6 ,   84.8 , 146.7 }
{ 9,   58.6 ,   72.8 ,   56.0 ,   69.6 ,   83.7 , 136.7 }
{10,   59.2 ,   70.6 ,   59.8 ,   57.0 ,   93.2 , 152.1 }
{11,   59.2 ,   71.0 ,   52.1 ,   75.2 ,   82.7 , 145.9 }
{12,   59.1 ,   69.3 ,   59.0 ,   61.7 ,   88.1 , 153.0 }
{13,   59.2 ,   76.1 ,   67.9 ,   81.7 ,   84.0 , 148.9 }
{14,   59.1 ,   70.1 ,   50.4 ,   66.5 ,   89.7 , 169.9 }
{15,   58.9 ,   74.8 ,   69.3 ,   87.9 ,   85.4 , 139.3 }
{16,   59.1 ,   76.0 ,   63.7 ,  100.0 ,   83.7 , 148.0 }
{17,   59.0 ,   69.4 ,   55.7 ,   67.7 ,   86.1 , 133.8 }
{18,   58.6 ,   72.9 ,   60.8 ,   68.6 ,   79.2 , 159.1 }
{19,   59.2 ,   71.0 ,   53.0 ,   77.2 ,   86.4 , 153.0 }
{20,   59.1 ,   69.1 ,   52.5 ,   74.7 ,   81.9 , 147.5 }
{21,   59.1 ,   67.7 ,   54.4 ,   70.1 ,   86.2 , 142.7 }
{22,   59.1 ,   70.2 ,   53.6 ,   89.9 ,   78.5 , 138.1 }
{23,   58.8 ,   73.2 ,   56.7 ,   72.7 ,   89.3 , 168.0 }
{24,   58.3 ,   66.8 ,   56.4 ,   78.5 ,   95.4 , 135.0 }
{25,   59.1 ,   69.6 ,   58.2 ,   79.8 ,   89.8 , 140.2 }
{26,   58.5 ,   62.5 ,   56.2 ,   83.7 ,   93.6 , 146.9 }
{27,   57.8 ,   64.5 ,   59.9 ,   83.6 ,   75.8 , 143.8 }
{28,   59.2 ,   70.9 ,   52.8 ,   80.1 ,   76.8 , 161.4 }
{29,   59.1 ,   69.1 ,   52.5 ,   72.6 ,   92.5 , 121.6 }
{30,   59.3 ,   69.5 ,   53.6 ,   72.9 ,   87.8 , 125.5 }
{31,   59.1 ,   70.1 ,   50.9 ,   77.1 ,   97.1 , 166.7 }
{32,   58.8 ,   73.3 ,   51.2 ,   74.3 ,   94.2 , 166.8 }
{33,   59.3 ,   69.5 ,   53.6 ,   72.9 ,   99.4 , 133.9 }
{34,   59.0 ,   73.8 ,   50.9 ,   82.6 ,   89.7 , 139.5 }
{35,   58.5 ,   62.9 ,   56.8 ,   75.8 ,   84.7 , 138.1 }
{36,   59.3 ,   69.5 ,   53.6 ,   72.9 ,  107.0 , 159.5 }
```

Measure 7

Red APC Losses During Cycle -- by case number and time sample

{ 1,	788.3 ,	548.6 ,	419.0 ,	255.4 ,	330.5 ,	482.2 }
{ 2,	796.0 ,	538.7 ,	393.7 ,	228.2 ,	271.0 ,	4[illegible]7.3 }
{ 3,	766.3 ,	549.8 ,	409.0 ,	257.2 ,	281.8 ,	455.1 }
{ 4,	790.0 ,	537.1 ,	389.3 ,	224.7 ,	285.2 ,	495.8 }
{ 5,	802.7 ,	575.5 ,	354.7 ,	244.2 ,	301.0 ,	492.2 }
{ 6,	781.5 ,	552.2 ,	382.5 ,	236.4 ,	291.4 ,	471.1 }
{ 7,	754.2 ,	540.9 ,	387.3 ,	218.6 ,	293.2 ,	477.1 }
{ 8,	775.3 ,	568.8 ,	377.2 ,	249.6 ,	264.5 ,	477.7 }
{ 9,	799.3 ,	607.3 ,	371.3 ,	234.9 ,	303.2 ,	470.1 }
{10,	717.4 ,	596.0 ,	409.4 ,	226.6 ,	279.1 ,	480.3 }
{11,	796.2 ,	541.2 ,	378.1 ,	256.9 ,	283.6 ,	479.4 }
{12,	766.0 ,	549.6 ,	396.5 ,	229.0 ,	278.2 ,	458.9 }
{13,	720.7 ,	582.6 ,	402.6 ,	244.2 ,	270.1 ,	470.7 }
{14,	805.2 ,	598.7 ,	360.2 ,	268.1 ,	279.7 ,	489.7 }
{15,	782.8 ,	564.9 ,	396.1 ,	257.1 ,	298.5 ,	457.4 }
{16,	724.6 ,	611.1 ,	396.2 ,	278.9 ,	272.2 ,	475.2 }
{17,	812.5 ,	575.7 ,	377.6 ,	229.2 ,	284.1 ,	476.3 }
{18,	795.2 ,	608.0 ,	393.5 ,	223.5 ,	287.4 ,	465.2 }
{19,	796.1 ,	540.3 ,	408.4 ,	223.4 ,	288.6 ,	483.5 }
{20,	767.3 ,	563.6 ,	373.5 ,	240.5 ,	276.6 ,	463.0 }
{21,	743.3 ,	568.2 ,	402.7 ,	243.4 ,	299.0 ,	445.0 }
{22,	800.0 ,	588.8 ,	409.4 ,	224.5 ,	305.3 ,	482.9 }
{23,	805.5 ,	563.8 ,	374.2 ,	225.1 ,	285.6 ,	463.0 }
{24,	759.7 ,	575.4 ,	366.0 ,	248.0 ,	276.3 ,	418.5 }
{25,	787.3 ,	533.3 ,	396.8 ,	223.0 ,	282.1 ,	514.7 }
{26,	773.7 ,	584.4 ,	380.2 ,	222.1 ,	279.0 ,	456.9 }
{27,	772.5 ,	648.4 ,	349.6 ,	222.9 ,	273.8 ,	416.1 }
{28,	797.1 ,	553.3 ,	389.6 ,	253.7 ,	277.8 ,	470.0 }
{29,	767.2 ,	564.1 ,	374.3 ,	227.4 ,	285.3 ,	467.0 }
{30,	754.8 ,	616.2 ,	411.0 ,	216.6 ,	306.7 ,	470.4 }
{31,	804.3 ,	578.9 ,	404.1 ,	239.3 ,	272.4 ,	458.0 }
{32,	806.1 ,	565.7 ,	373.7 ,	230.8 ,	298.7 ,	467.8 }
{33,	754.8 ,	616.2 ,	411.0 ,	216.6 ,	310.4 ,	451.4 }
{34,	808.4 ,	610.6 ,	380.1 ,	227.6 ,	270.1 ,	462.6 }
{35,	770.6 ,	588.8 ,	374.6 ,	232.7 ,	275.5 ,	473.9 }
{36,	754.8 ,	616.2 ,	411.0 ,	216.6 ,	323.9 ,	459.1 }

Measure 8

Blue Aircraft Loss Rate -- by case number and time sample
(Loss Rate = Number Killed During Cycle / Number At Start of Cycle)

```
{ 1, .1899 , .1243 , .1408 , .1018 , .0415 , .0017 }
{ 2, .1599 , .1095 , .1307 , .1301 , .1043 , .0084 }
{ 3, .1245 , .0786 , .0912 , .0896 , .0876 , .0869 }
{ 4, .2060 , .1394 , .1469 , .0954 , .0359 , .0017 }
{ 5, .1854 , .1362 , .1602 , .1409 , .0901 , .0034 }
{ 6, .1557 , .1083 , .1256 , .1155 , .1132 , .0619 }
{ 7, .2214 , .1471 , .1572 , .0891 , .0330 , .0017 }
{ 8, .2079 , .1577 , .1807 , .1545 , .0718 , .0035 }
{ 9, .1791 , .1308 , .1467 , .1331 , .1395 , .0286 }
{10, .1829 , .1185 , .1394 , .1036 , .0459 , .0017 }
{11, .1584 , .1057 , .1300 , .1321 , .1041 , .0084 }
{12, .1243 , .0780 , .0928 , .0896 , .0871 , .0857 }
{13, .1983 , .1342 , .1501 , .1010 , .0394 , .0017 }
{14, .1829 , .1309 , .1588 , .1429 , .0902 , .0034 }
{15, .1540 , .1185 , .1242 , .1149 , .1140 , .0624 }
{16, .2114 , .1472 , .1607 , .0917 , .0367 , .0017 }
{17, .2029 , .1517 , .1799 , .1477 , .0732 , .0017 }
{18, .1788 , .1287 , .1479 , .1323 , .1257 , .0332 }
{19, .1829 , .1279 , .1122 , .0796 , .0429 , .0083 }
{20, .1894 , .1276 , .1367 , .1087 , .0756 , .0216 }
{21, .1671 , .1049 , .1230 , .1038 , .0891 , .0606 }
{22, .2275 , .1632 , .1394 , .0778 , .0398 , .0034 }
{23, .2564 , .1875 , .1853 , .1165 , .0565 , .0068 }
{24, .2428 , .1792 , .1842 , .1419 , .1072 , .0278 }
{25, .2629 , .1749 , .1482 , .0775 , .0336 , .0017 }
{26, .3073 , .2258 , .2080 , .1094 , .0431 , .0035 }
{27, .2988 , .2301 , .2274 , .1628 , .1000 , .0169 }
{28, .1814 , .1239 , .1096 , .0855 , .0428 , .0099 }
{29, .1849 , .1294 , .1348 , .1105 , .0710 , .0200 }
{30, .0763 , .0433 , .0594 , .0602 , .0896 , .0749 }
{31, .2260 , .1589 , .1415 , .0811 , .0366 , .0017 }
{32, .2529 , .1853 , .1870 , .1131 , .0578 , .0085 }
{33, .0763 , .0433 , .0594 , .0602 , .1503 , .0751 }
{34, .2590 , .1774 , .1538 , .0732 , .0338 , .0017 }
{35, .3028 , .2237 , .2082 , .1114 , .0472 , .0017 }
{36, .0763 , .0433 , .0594 , .0602 , .1892 , .0628 }
```

Measure 9

Tank Exchange Ratio for the Cycle -- by case number and time sample (Tank Exchange Ratio = Blue Tank Losses / Red Tank Losses During Cycle)

{ 1,	1.1769 ,	1.7942 ,	1.7244 ,	1.5522 ,	0.9598 ,	2.3457 }
{ 2,	1.1604 ,	1.7915 ,	1.7181 ,	1.9038 ,	1.9228 ,	2.1429 }
{ 3,	1.2107 ,	1.6929 ,	1.7457 ,	1.4414 ,	2.0378 ,	2.2988 }
{ 4,	1.1570 ,	1.7862 ,	1.7984 ,	2.2803 ,	1.9813 ,	1.7744 }
{ 5,	1.0885 ,	1.6034 ,	1.8155 ,	2.1128 ,	1.6151 ,	1.9459 }
{ 6,	1.0058 ,	1.5556 ,	1.1422 ,	1.5449 ,	1.6371 ,	2.6675 }
{ 7,	1.2379 ,	1.7069 ,	1.7770 ,	2.4141 ,	1.4090 ,	2.5445 }
{ 8,	1.0988 ,	1.5478 ,	1.3526 ,	1.7489 ,	2.0146 ,	1.6467 }
{ 9,	0.8607 ,	1.2833 ,	1.3972 ,	1.5550 ,	1.6984 ,	2.3457 }
{10,	1.5172 ,	1.3302 ,	1.2280 ,	1.9827 ,	1.9938 ,	2.3694 }
{11,	1.1632 ,	1.7974 ,	1.8331 ,	1.1758 ,	1.9501 ,	2.1703 }
{12,	1.2132 ,	1.7016 ,	1.6699 ,	1.6141 ,	2.0392 ,	2.4766 }
{13,	1.4686 ,	1.5795 ,	1.7159 ,	1.7490 ,	1.9657 ,	2.3797 }
{14,	1.0831 ,	1.4062 ,	1.0836 ,	1.0216 ,	1.7946 ,	1.6871 }
{15,	1.0186 ,	1.5542 ,	0.9500 ,	1.8689 ,	1.5527 ,	2.7832 }
{16,	1.4268 ,	1.3886 ,	1.2214 ,	1.3069 ,	2.1467 ,	2.1809 }
{17,	1.0215 ,	1.5667 ,	1.2246 ,	1.5990 ,	1.4251 ,	1.9816 }
{18,	0.8788 ,	1.2981 ,	1.2951 ,	1.4463 ,	1.7853 ,	2.1045 }
{19,	1.1537 ,	1.7671 ,	1.5249 ,	2.1256 ,	1.9667 ,	2.5781 }
{20,	1.1678 ,	1.6973 ,	1.5656 ,	1.8157 ,	1.6651 ,	2.2872 }
{21,	1.2019 ,	1.5204 ,	1.2640 ,	1.7836 ,	1.9328 ,	2.3383 }
{22,	1.0784 ,	1.4366 ,	1.0459 ,	1.8756 ,	1.6753 ,	2.0050 }
{23,	0.9609 ,	1.4727 ,	1.3762 ,	1.9277 ,	1.9770 ,	2.7198 }
{24,	0.8157 ,	1.1191 ,	1.1210 ,	1.5842 ,	1.9254 ,	2.8139 }
{25,	1.1260 ,	1.6814 ,	1.5162 ,	1.7489 ,	2.0235 ,	1.2836 }
{26,	0.8281 ,	1.4198 ,	1.5583 ,	2.1416 ,	2.2503 ,	2.1233 }
{27,	0.6466 ,	1.0223 ,	1.1704 ,	1.7525 ,	1.7684 ,	2.9894 }
{28,	1.1517 ,	1.6978 ,	1.8901 ,	1.6259 ,	1.8676 ,	2.7279 }
{29,	1.1748 ,	1.6847 ,	1.5545 ,	1.8945 ,	1.6759 ,	1.9984 }
{30,	1.3714 ,	1.3455 ,	1.2216 ,	2.0764 ,	1.1769 ,	1.7626 }
{31,	1.0695 ,	1.4849 ,	1.1500 ,	1.8772 ,	2.0310 ,	2.6164 }
{32,	0.9660 ,	1.4769 ,	1.4174 ,	1.8535 ,	1.3894 ,	2.3446 }
{33,	1.3714 ,	1.3455 ,	1.2216 ,	2.0764 ,	0.9742 ,	2.1070 }
{34,	1.0204 ,	1.5634 ,	1.4009 ,	2.1662 ,	2.1080 ,	2.1438 }
{35,	0.8420 ,	1.4561 ,	1.3301 ,	1.7673 ,	2.0183 ,	2.4574 }
{36,	1.3714 ,	1.3455 ,	1.2216 ,	2.0764 ,	1.1087 ,	2.8420 }

Measure 10

APC Exchange Ratio for the Cycle -- by case number and time sample (APC Exchange Ratio = Blue APC Losses / Red APC Losses During Cycle)

{ 1,	0.0752 ,	0.1294 ,	0.1248 ,	0.3320 ,	0.2711 ,	0.2781 }
{ 2,	0.0744 ,	0.1318 ,	0.1283 ,	0.2940 ,	0.2919 ,	0.3318 }
{ 3,	0.0771 ,	0.1260 ,	0.1553 ,	0.2858 ,	0.3009 ,	0.3395 }
{ 4,	0.0749 ,	0.1305 ,	0.1402 ,	0.3721 ,	0.2903 ,	0.3126 }
{ 5,	0.0736 ,	0.1218 ,	0.1438 ,	0.3374 ,	0.4086 ,	0.2651 }
{ 6,	0.0754 ,	0.1239 ,	0.1370 ,	0.2885 ,	0.3874 ,	0.2566 }
{ 7,	0.0785 ,	0.1163 ,	0.1614 ,	0.3696 ,	0.3087 ,	0.2668 }
{ 8,	0.0761 ,	0.1215 ,	0.1440 ,	0.3069 ,	0.3206 ,	0.3071 }
{ 9,	0.0733 ,	0.1199 ,	0.1508 ,	0.2963 ,	0.2761 ,	0.2908 }
{10,	0.0825 ,	0.1185 ,	0.1461 ,	0.2515 ,	0.3339 ,	0.3167 }
{11,	0.0744 ,	0.1312 ,	0.1378 ,	0.2927 ,	0.2916 ,	0.3043 }
{12,	0.0772 ,	0.1261 ,	0.1488 ,	0.2694 ,	0.3167 ,	0.3334 }
{13,	0.0821 ,	0.1306 ,	0.1687 ,	0.3346 ,	0.3110 ,	0.3163 }
{14,	0.0734 ,	0.1171 ,	0.1399 ,	0.2480 ,	0.3207 ,	0.3469 }
{15,	0.0752 ,	0.1324 ,	0.1750 ,	0.3419 ,	0.2861 ,	0.3045 }
{16,	0.0816 ,	0.1244 ,	0.1608 ,	0.3586 ,	0.3075 ,	0.3114 }
{17,	0.0726 ,	0.1205 ,	0.1475 ,	0.2954 ,	0.3031 ,	0.2809 }
{18,	0.0737 ,	0.1199 ,	0.1545 ,	0.3069 ,	0.2756 ,	0.3420 }
{19,	0.0744 ,	0.1314 ,	0.1298 ,	0.3456 ,	0.2994 ,	0.3164 }
{20,	0.0770 ,	0.1226 ,	0.1406 ,	0.3106 ,	0.2961 ,	0.3186 }
{21,	0.0795 ,	0.1191 ,	0.1351 ,	0.2880 ,	0.2883 ,	0.3207 }
{22,	0.0739 ,	0.1192 ,	0.1309 ,	0.4004 ,	0.2571 ,	0.2860 }
{23,	0.0730 ,	0.1298 ,	0.1515 ,	0.3230 ,	0.3127 ,	0.3629 }
{24,	0.0767 ,	0.1161 ,	0.1541 ,	0.3165 ,	0.3453 ,	0.3226 }
{25,	0.0751 ,	0.1305 ,	0.1467 ,	0.3578 ,	0.3183 ,	0.2724 }
{26,	0.0756 ,	0.1069 ,	0.1478 ,	0.3769 ,	0.3355 ,	0.3215 }
{27,	0.0748 ,	0.0995 ,	0.1713 ,	0.3751 ,	0.2768 ,	0.3456 }
{28,	0.0743 ,	0.1281 ,	0.1355 ,	0.3157 ,	0.2765 ,	0.3434 }
{29,	0.0770 ,	0.1225 ,	0.1403 ,	0.3193 ,	0.3242 ,	0.2604 }
{30,	0.0786 ,	0.1128 ,	0.1304 ,	0.3366 ,	0.2863 ,	0.2668 }
{31,	0.0735 ,	0.1211 ,	0.1260 ,	0.3222 ,	0.3565 ,	0.3640 }
{32,	0.0729 ,	0.1296 ,	0.1370 ,	0.3219 ,	0.3154 ,	0.3566 }
{33,	0.0786 ,	0.1128 ,	0.1304 ,	0.3366 ,	0.3202 ,	0.2966 }
{34,	0.0730 ,	0.1209 ,	0.1339 ,	0.3629 ,	0.3321 ,	0.3016 }
{35,	0.0759 ,	0.1068 ,	0.1516 ,	0.3257 ,	0.3074 ,	0.2914 }
{36,	0.0786 ,	0.1128 ,	0.1304 ,	0.3366 ,	0.3303 ,	0.3474 }

Measure 11

Total BAI Kills as of Day 60 -- by case number

```
{ 1 ,  39.10}
{ 2 , 137.35}
{ 3 , 324.02}
{ 4 ,  21.61}
{ 5 ,  69.95}
{ 6 , 137.31}
{ 7 ,   3.97}
{ 8 ,  12.50}
{ 9 ,  27.86}
{10 ,  41.61}
{11 , 124.50}
{12 , 297.91}
{13 ,  25.09}
{14 ,  66.92}
{15 , 135.52}
{16 ,   4.42}
{17 ,  12.53}
{18 ,  26.55}
{19 , 161.28}
{20 , 307.30}
{21 , 461.65}
{22 ,  77.31}
{23 , 156.56}   <---this had been typed as 1156.56
{24 , 237.35}
{25 ,  10.99}
{26 ,  27.92}
{27 ,  58.52}
{28 , 163.13}
{29 , 293.20}
{30 ,  68.24}
{31 ,  83.78}
{32 , 156.03}
{33 ,  45.74}
{34 ,  13.25}
{35 ,  28.75}
{36 , 102.56}
```

Measure 12

Cumulative Blue Aircraft Killed -- by case number and time sample (Calculated from data provided by CAA)

```
{ 1, 247 , 378 , 518 , 712 , 849 , 934 }
{ 2, 320 , 504 , 709 ,1056 ,1397 ,1684 }
{ 3, 436 , 677 , 941 ,1404 ,1955 ,2790 }
{ 4, 268 , 412 , 553 , 736 , 857 , 930 }
{ 5, 371 , 593 , 830 ,1185 ,1479 ,1691 }
{ 6, 545 , 865 ,1205 ,1750 ,2344 ,3036 }
{ 7, 288 , 437 , 584 , 759 , 869 , 936 }
{ 8, 416 , 666 , 920 ,1285 ,1544 ,1702 }
{ 9, 627 ,1003 ,1380 ,1958 ,2576 ,3185 }
{10, 238 , 364 , 505 , 704 , 850 , 944 }
{11, 317 , 495 , 701 ,1053 ,1397 ,1684 }
{12, 435 , 674 , 943 ,1409 ,1958 ,2786 }
{13, 258 , 398 , 544 , 737 , 867 , 946 }
{14, 366 , 580 , 817 ,1178 ,1478 ,1693 }
{15, 539 , 858 ,1195 ,1738 ,2336 ,3034 }
{16, 275 , 426 , 578 , 759 , 875 , 948 }
{17, 406 , 648 , 904 ,1263 ,1521 ,1681 }
{18, 626 , 996 ,1377 ,1956 ,2541 ,3114 }
{19, 238 , 374 , 486 , 645 , 776 , 889 }
{20, 379 , 586 , 789 ,1087 ,1354 ,1614 }
{21, 585 , 891 ,1221 ,1725 ,2239 ,2891 }
{22, 296 , 460 , 587 , 740 , 852 , 938 }
{23, 513 , 792 ,1029 ,1315 ,1509 ,1648 }
{24, 850 ,1325 ,1739 ,2288 ,2746 ,3163 }
{25, 342 , 514 , 641 , 788 , 886 , 953 }
{26, 615 , 928 ,1166 ,1421 ,1568 ,1660 }
{27,1046 ,1611 ,2057 ,2596 ,2977 ,3272 }
{28, 236 , 368 , 478 , 645 , 782 , 899 }
{29, 370 , 581 , 782 ,1082 ,1343 ,1587 }
{30, 267 , 407 , 595 , 948 ,1531 ,2506 }
{31, 294 , 454 , 584 , 743 , 854 , 930 }
{32, 506 , 783 ,1024 ,1308 ,1501 ,1647 }
{33, 267 , 407 , 595 , 948 ,1769 ,3013 }
{34, 337 , 508 , 641 , 787 , 882 , 949 }
{35, 606 , 918 ,1158 ,1419 ,1575 ,1672 }
{36, 267 , 407 , 595 , 948 ,1901 ,3263 }
```

Measure 13

Cumulative Blue Tanks Killed -- by case number and time sample (Calculated from data provided by CAA)

```
{ 1,  197 ,  438 ,  631 ,  992 , 1412 , 2613 }
{ 2,  197 ,  447 ,  596 ,  858 , 1346 , 2540 }
{ 3,  177 ,  429 ,  602 ,  915 , 1368 , 2565 }
{ 4,  197 ,  447 ,  607 ,  921 , 1353 , 2538 }
{ 5,  196 ,  413 ,  560 ,  921 , 1479 , 2980 }
{ 6,  175 ,  400 ,  587 ,  877 , 1336 , 2676 }
{ 7,  176 ,  424 ,  587 ,  965 , 1451 , 2639 }
{ 8,  176 ,  389 ,  569 ,  895 , 1357 , 2485 }
{ 9,  174 ,  412 ,  614 ,  936 , 1440 , 2635 }
{10,  190 ,  419 ,  600 ,  896 , 1354 , 2572 }
{11,  197 ,  449 ,  608 ,  986 , 1450 , 2486 }
{12,  177 ,  429 ,  603 ,  868 , 1299 , 2594 }
{13,  189 ,  451 ,  652 , 1013 , 1473 , 2579 }
{14,  196 ,  436 ,  592 ,  917 , 1324 , 2456 }
{15,  176 ,  419 ,  617 , 1006 , 1522 , 2784 }
{16,  189 ,  467 ,  691 , 1126 , 1696 , 2821 }
{17,  195 ,  445 ,  602 ,  865 , 1284 , 2329 }
{18,  174 ,  412 ,  653 ,  967 , 1407 , 2526 }
{19,  197 ,  437 ,  594 ,  885 , 1299 , 2697 }
{20,  177 ,  430 ,  605 ,  912 , 1268 , 2292 }
{21,  174 ,  409 ,  587 ,  963 , 1488 , 2610 }
{22,  195 ,  424 ,  613 ,  939 , 1376 , 2445 }
{23,  179 ,  414 ,  581 ,  921 , 1401 , 2829 }
{24,  169 ,  431 ,  624 ,  958 , 1448 , 2832 }
{25,  181 ,  427 ,  619 ,  943 , 1424 , 2596 }
{26,  167 ,  401 ,  623 , 1074 , 1666 , 2771 }
{27,  165 ,  404 ,  582 ,  904 , 1333 , 2534 }
{28,  197 ,  449 ,  630 ,  935 , 1338 , 2802 }
{29,  177 ,  430 ,  601 ,  910 , 1363 , 2333 }
{30,  205 ,  431 ,  640 ,  965 , 1357 , 2244 }
{31,  195 ,  417 ,  642 , 1022 , 1542 , 2832 }
{32,  179 ,  413 ,  597 ,  895 , 1378 , 2649 }
{33,  205 ,  431 ,  640 ,  965 , 1399 , 2459 }
{34,  195 ,  473 ,  637 ,  963 , 1438 , 2575 }
{35,  167 ,  401 ,  564 ,  842 , 1259 , 2463 }
{36,  205 ,  431 ,  640 ,  965 , 1565 , 3079 }
```

Measure 14

Cumulative Red Tanks Killed -- by case number and time sample (Calculated from data provided by CAA)

```
{ 1,  167 ,  302 ,  413 ,  640 , 1007 , 1744 }
{ 2,  170 ,  309 ,  396 ,  538 ,  793 , 1379 }
{ 3,  146 ,  295 ,  394 ,  601 ,  854 , 1400 }
{ 4,  170 ,  310 ,  399 ,  546 ,  754 , 1397 }
{ 5,  180 ,  315 ,  396 ,  573 ,  891 , 1720 }
{ 6,  174 ,  319 ,  483 ,  691 ,  977 , 1595 }
{ 7,  143 ,  288 ,  379 ,  548 ,  834 , 1438 }
{ 8,  160 ,  298 ,  431 ,  632 ,  874 , 1508 }
{ 9,  202 ,  388 ,  532 ,  746 , 1051 , 1648 }
{10,  125 ,  298 ,  445 ,  622 ,  852 , 1404 }
{11,  169 ,  309 ,  396 ,  693 ,  998 , 1497 }
{12,  146 ,  294 ,  398 ,  560 ,  787 , 1351 }
{13,  129 ,  294 ,  412 ,  619 ,  864 , 1367 }
{14,  181 ,  352 ,  495 ,  810 , 1106 , 1764 }
{15,  172 ,  329 ,  537 ,  797 , 1108 , 1698 }
{16,  133 ,  333 ,  516 ,  855 , 1185 , 1705 }
{17,  191 ,  351 ,  479 ,  658 ,  943 , 1557 }
{18,  198 ,  382 ,  567 ,  794 , 1058 , 1629 }
{19,  171 ,  307 ,  409 ,  561 ,  766 , 1357 }
{20,  151 ,  300 ,  413 ,  589 ,  796 , 1295 }
{21,  145 ,  300 ,  441 ,  672 ,  952 , 1476 }
{22,  181 ,  340 ,  521 ,  735 ,  986 , 1561 }
{23,  187 ,  346 ,  468 ,  661 ,  906 , 1495 }
{24,  207 ,  441 ,  613 ,  849 , 1121 , 1695 }
{25,  160 ,  307 ,  433 ,  627 ,  877 , 1650 }
{26,  202 ,  367 ,  509 ,  739 , 1007 , 1513 }
{27,  256 ,  489 ,  641 ,  850 , 1094 , 1590 }
{28,  171 ,  320 ,  415 ,  595 ,  822 , 1425 }
{29,  151 ,  301 ,  411 ,  584 ,  844 , 1373 }
{30,  150 ,  317 ,  489 ,  681 ,  959 , 1566 }
{31,  183 ,  332 ,  527 ,  768 , 1031 , 1580 }
{32,  186 ,  344 ,  474 ,  650 ,  973 , 1668 }
{33,  150 ,  317 ,  489 ,  681 , 1046 , 1787 }
{34,  191 ,  369 ,  486 ,  657 ,  881 , 1414 }
{35,  199 ,  359 ,  482 ,  654 ,  870 , 1398 }
{36,  150 ,  317 ,  489 ,  681 , 1160 , 2066 }
```

Measure 15

Cumulative Blue APCs Killed -- by case number and time sample (Calculated from data provided by CAA)

```
{ 1,   59 ,  130 ,  183 ,  336 ,  600 , 1405 }
{ 2,   59 ,  130 ,  181 ,  307 ,  532 , 1363 }
{ 3,   59 ,  128 ,  192 ,  334 ,  577 , 1449 }
{ 4,   59 ,  129 ,  185 ,  338 ,  587 , 1455 }
{ 5,   59 ,  129 ,  180 ,  329 ,  658 , 1549 }
{ 6,   59 ,  127 ,  180 ,  308 ,  602 , 1425 }
{ 7,   59 ,  122 ,  185 ,  337 ,  599 , 1380 }
{ 8,   59 ,  128 ,  182 ,  324 ,  571 , 1412 }
{ 9,   59 ,  131 ,  187 ,  320 ,  557 , 1355 }
{10,   59 ,  130 ,  190 ,  305 ,  548 , 1436 }
{11,   59 ,  130 ,  182 ,  321 ,  562 , 1393 }
{12,   59 ,  128 ,  187 ,  309 ,  547 , 1424 }
{13,   59 ,  135 ,  203 ,  360 ,  609 , 1457 }
{14,   59 ,  129 ,  180 ,  305 ,  550 , 1499 }
{15,   59 ,  134 ,  203 ,  370 ,  628 , 1442 }
{16,   59 ,  135 ,  199 ,  381 ,  648 , 1491 }
{17,   59 ,  128 ,  184 ,  314 ,  553 , 1347 }
{18,   59 ,  132 ,  192 ,  326 ,  553 , 1427 }
{19,   59 ,  130 ,  183 ,  326 ,  576 , 1447 }
{20,   59 ,  128 ,  181 ,  319 ,  558 , 1393 }
{21,   59 ,  127 ,  181 ,  314 ,  556 , 1386 }
{22,   59 ,  129 ,  183 ,  345 ,  591 , 1379 }
{23,   59 ,  132 ,  189 ,  326 ,  577 , 1517 }
{24,   58 ,  125 ,  182 ,  327 ,  597 , 1423 }
{25,   59 ,  129 ,  187 ,  336 ,  595 , 1425 }
{26,   59 ,  121 ,  177 ,  331 ,  602 , 1470 }
{27,   58 ,  122 ,  182 ,  338 ,  573 , 1375 }
{28,   59 ,  130 ,  183 ,  329 ,  563 , 1439 }
{29,   59 ,  128 ,  181 ,  316 ,  573 , 1337 }
{30,   59 ,  129 ,  182 ,  319 ,  567 , 1332 }
{31,   59 ,  129 ,  180 ,  321 ,  593 , 1551 }
{32,   59 ,  132 ,  183 ,  320 ,  583 , 1533 }
{33,   59 ,  129 ,  182 ,  319 ,  590 , 1424 }
{34,   59 ,  133 ,  184 ,  333 ,  595 , 1422 }
{35,   59 ,  121 ,  178 ,  320 ,  566 , 1372 }
{36,   59 ,  129 ,  182 ,  319 ,  605 , 1564 }
```

Measure 16

Cumulative Red APCs Killed -- by case number and time sample (Calculated from data provided by CAA)

```
{ 1,  788 , 1337 , 1756 , 2349 , 3265 , 6185 }
{ 2,  796 , 1335 , 1728 , 2268 , 3038 , 5640 }
{ 3,  766 , 1316 , 1725 , 2315 , 3136 , 5802 }
{ 4,  790 , 1327 , 1716 , 2248 , 3043 , 5882 }
{ 5,  803 , 1378 , 1733 , 2277 , 3123 , 5995 }
{ 6,  782 , 1334 , 1716 , 2262 , 3081 , 5840 }
{ 7,  754 , 1295 , 1682 , 2204 , 3009 , 5797 }
{ 8,  775 , 1344 , 1721 , 2284 , 3063 , 5767 }
{ 9,  799 , 1407 , 1778 , 2316 , 3157 , 5947 }
{10,  717 , 1313 , 1723 , 2267 , 3052 , 5811 }
{11,  796 , 1337 , 1716 , 2290 , 3114 , 5882 }
{12,  766 , 1316 , 1712 , 2254 , 3039 , 5709 }
{13,  721 , 1303 , 1706 , 2274 , 3058 , 5751 }
{14,  805 , 1404 , 1764 , 2346 , 3174 , 5972 }
{15,  783 , 1348 , 1744 , 2328 , 3182 , 5907 }
{16,  725 , 1336 , 1732 , 2348 , 3172 , 5889 }
{17,  813 , 1388 , 1766 , 2298 , 3096 , 5853 }
{18,  795 , 1403 , 1797 , 2329 , 3127 , 5850 }
{19,  796 , 1336 , 1745 , 2284 , 3085 , 5885 }
{20,  767 , 1331 , 1704 , 2252 , 3046 , 5727 }
{21,  743 , 1312 , 1714 , 2281 , 3122 , 5799 }
{22,  800 , 1389 , 1798 , 2340 , 3175 , 6022 }
{23,  806 , 1369 , 1744 , 2268 , 3065 , 5773 }
{24,  760 , 1335 , 1701 , 2256 , 3057 , 5560 }
{25,  787 , 1321 , 1717 , 2250 , 3038 , 5943 }
{26,  774 , 1358 , 1738 , 2262 , 3042 , 5706 }
{27,  773 , 1421 , 1771 , 2280 , 3050 , 5536 }
{28,  797 , 1350 , 1740 , 2315 , 3125 , 5838 }
{29,  767 , 1331 , 1706 , 2234 , 3032 , 5756 }
{30,  755 , 1371 , 1782 , 2312 , 3142 , 5944 }
{31,  804 , 1383 , 1787 , 2348 , 3132 , 5782 }
{32,  806 , 1372 , 1746 , 2279 , 3107 , 5874 }
{33,  755 , 1371 , 1782 , 2312 , 3150 , 5887 }
{34,  808 , 1419 , 1799 , 2331 , 3098 , 5759 }
{35,  771 , 1359 , 1734 , 2270 , 3054 , 5776 }
{36,  755 , 1371 , 1782 , 2312 , 3177 , 5985 }
```

APPENDIX C

AUTOMATIC DETECTION OF KEY SENSITIVITIES

CONTENTS, APPENDIX C

A. INTRODUCTION 1

B. GEOMETRIC MEASURES OF MULTIDIMENSIONAL DATA 2

1. Purpose 2
2. List of Measures 2
3. Descriptions 3

C. CURV PROGRAM 4

1. Overview 4
2. Structure of the Input File.....4
3. Fiber and Animation Directions 5
4. Mathematical Fundamentals 6
5. Program Algorithms.....8
6. Sample Output.....9

A. INTRODUCTION

Analysts are often overwhelmed by their data. Modern technology is capable of generating and transmitting information (data) at ever increasing rates of speed. This is both a blessing and a curse. While gaining access to more and more information, analysts are confronted with the increasingly complex problem of absorbing (i.e., understanding) the data to which they have access.

Computer graphics have, to an enormous degree, provided a means of coping with this problem. By displaying curves and surfaces in a graphical (as opposed to tabular) format, one gains immediate insights into the nature of the presented data. Regions of greatest interest become apparent. Anomalies and errors jump out. General interest in the point being made is increased simply by the manner in which results are displayed.

Nonetheless, in a data-rich environment, the basic question still faces the analyst: which results are important? Paraphrased, how does the analyst choose most wisely the data to display (with graphics or any other medium)? Obviously, if the adjective "data-rich" were dropped from the discussion, the analyst would visually examine, and possibly display, all results. The focus of this section, by contrast, is on the situation where the sheer volume of data precludes this possibility.

The material described in this chapter is a methodology, implemented in Fortran code, for analyzing a data set and producing a collection of measures that clearly indicate the presence of anomalous data or complicated relationships. The data to be analyzed are read by the computer program, then subdivided in a natural fashion according to an input parameter set. These subdivisions are examined for their geometric content and compared quantitatively with one another.

The subdivisions correspond to the various ways that one could display cross-sections of a multidimensional data set as a plot containing multiple surfaces. Several measures have been developed that may be applied to these plots, or series of such plots. These measures are designed to address the following questions:

For each individual plot:

1. Are the surfaces in the plot close to one another?
2. Are any of the surfaces affine, or nearly so?
3. Are any of the successive differences (e.g., surface 2 minus surface 1, etc.) affine or nearly so?

4. Are there any distinguished data outliers in the plot?

For each animation sequence (i.e., a series) of plots:

5. If each plot in the series of plots has the same number of surfaces, are the successive differences (from plot to plot) of collections of surfaces close, constant or affine?

This remainder of this appendix is organized into a brief description of the measures, followed by an in-depth discussion of the underlying computer program. The motivation behind, and interpretation of, each measure will be made clear.

B. GEOMETRIC MEASURES OF MULTIDIMENSIONAL DATA

1. Purpose

The CURV program produces a collection of five measures that describe a given input data set. These measures are intended to portray global characteristic of the input: the "flatness" or lack thereof, the general trend, and the presence of anomalies. The point in computing these measures is to quickly convey to the analyst the degree to which there exists interesting relationships among the data at hand. Viewed slightly differently, these measures act like a filter that allows the analyst to discard the portions of a data set that provide no valuable insights or contain no information relevant to present needs.

Graphical displays convey all the above information. However, the question of which relationships to display when large multidimensional data sets are to be analyzed must first be resolved. It is precisely that question these measures are geared to answer.

2. List of Measures

M1 absolute value of greatest departure of a surface (i.e., family member) from the XY plane. A sequence of surfaces shown in one plot is termed a family or bundle. The variation of the 'fiber' dimension determines the sequence.

M2 absolute value of greatest difference between two successive family members (i.e., greatest one step jump in the fiber direction).

M3 sum of the M2s over a complete family.

M4 the average variation in the normal vector over one surface.

M5 the average normal vector to a surface.

3. Descriptions

The first measure, M1, is the greatest departure of a given surface from the XY plane. Clearly, a small value for M1 indicates little variation in the data. The surface is essentially flat and geometrically uninteresting in this case. On the other hand, a large value of M1 says very little about the surface's shape; instead, the implication is that some points wander far above or below the XY plane.

For this reason, M1 must often be used in conjunction with other measures in order to form a cogent description of the surfaces.

The second measure, M2, is the greatest difference between two successive surfaces (that is, two surfaces in the fiber direction). Thus, a small value for M2 indicates that little or no change takes place as one steps from a specific family member to the next. The two surfaces measured show very little variation with respect to the parameter that defines the individual family members. Conversely, large values for M2 indicate a high degree of sensitivity to this parameter.

The third measure, M3, is the sum of the M2 taken over one entire family. A small M3 indicates that the parameter distinguishing individual family members has little or no importance. The data are insensitive to it. On the other hand, a substantial value of M3 indicates a qualitative difference among the surfaces as one cuts across all members of a given family.

The two remaining measures differ in nature from the previous three. Roughly speaking, CURV generates unit normal vectors at various points on the surface. CURV then measures the variation of the unit normal vectors as one moves from vector to vector. If, for example, the surface is a plane, then this measure of variation will be zero. In fact, this is the only case in which the measure is zero. In comparison, if the surface is a hemisphere of any radius, the measure will be approximately 0.904.

The fifth and final measure is the surface's "average" unit normal vector. The 'average' is computed in the following sense: all of the unit normal vectors are added together (each is given a weight determined by the area of the 'patch' it represents). Since each vector has a non-zero Z-component, the vector sum is non-zero. The normalization of this vector is the fifth measure, or "average" vector.

Information about the surface can be gathered by comparing measures 4 and 5. For example, if the surface is flat and horizontal, then measure 4 will be zero and measure 5

will be vertical. The converse is also true. Similarly, if measure 4 is zero and measure 5 is not vertical, then an examination of the components determines the surface's behavior. Specifically, the surface increases in the directions determined by the negative components of this vector.

C. CURV PROGRAM

1. Overview

CURV in an interactive Fortran program. It operates on one input data set and produces one output data set. The user must supply a name for each. CURV reads all the input data into a large array. The user is then requested to specify which dimensions will serve as the X and Y directions in all subsequent analyses. By specifying these directions, the user determines how the original data set will be partitioned into surfaces against which measures are applied.

Thus, in some sense, the input to CURV can be visualized as an n-dimensional array. The current upper limit on n, the number of array dimensions, is 7; however, the program could easily be enlarged to 10 or more if necessary. The choice of X and Y directions determines a natural partitioning, or foliation, of the data into two dimensional lattices or "surfaces". More precisely, by fixing all but two indices of the input array, a lattice is generated by allowing the free indices to vary over their domain. A surface is constructed by interpolating in two dimensions between the lattice points. The height (or Z-coordinate) of a surface over a particular point in the XY plane is the array value for the corresponding set of indices.

CURV applies measures to the surfaces generated from the input data set. A discussion of the order in which surfaces are generated is contained below. Mathematical fundamentals and algorithms appear subsequently.

2. Structure of the Input File

At the beginning of program execution, the user is asked to supply the name of an input file. This name is supplied from the terminal and can be any string not exceeding 31 characters (it must be enclosed in single quotes).

The structure of the input file is rather simple. All records are free format, and all entries within a record must be separated by a space or comma. The first record contains

the number of dimensions and the size of each dimension. For example, a first record equal to

5 4 4 3 6 2

indicates that the data array has five dimensions of length 4, 4, 3, 6 and 2, respectively.

The length of all subsequent data records is fixed and equal to the second field in the first record (in this case, the number 4). That is, the first dimension determines the length of all subsequent data records. Thus, the data contained in the first record correspond to array entries (1,1,1,1,1) through (4,1,1,1,1). Those contained in the first four records correspond to entries (1,1,1,1,1) through (4,4,1,1,1), etc.

Any data set organized in the above manner will be acceptable to CURV. Due to the free formatting feature, no additional structure is necessary.

3. Fiber and Animation Directions

Recall that a surface is generated by fixing the indices in n-2 of the array dimensions and allowing the remaining two (X and Y) to vary freely. Successive surfaces are constructed by choosing different values for the n-2 indices in the fixed vector. By varying one dimension at a time in the fixed vector, a series of surfaces is generated which differ from one another by only one parameter. Thus, a 'family' of surfaces is constructed. That is, a family of surfaces share n-1 indices in their respective fixed vectors. The index or dimension that parameterizes a given family is called the 'fiber direction.' Generalizing a little bit, collections of families can be parameterized by allowing one of the remaining n-3 indices of the fixed vector to vary. The index selected for this role is referred to as the 'animation direction.'

For example, suppose the basic data array had five dimensions. Assume the user specified that dimensions 2 and 5 were to serve as X and Y, respectively. Then CURV would initially choose dimension 1 as the fiber direction. By fixing coordinates in dimensions 1, 3, and 4 and by allowing X and Y to vary freely, a two dimensional lattice is generated. Now, by stepping to the next coordinate value in dimension 1, but keeping coordinates fixed in dimensions 3 and 4, a second lattice is generated. In this manner, a family of 'surfaces' is constructed.

An easy way to describe the procedure outlined in the example is to say this: CURV varies indices fastest in the the X and Y directions; the next fastest variation is in the fiber

direction. If more than three dimensions exist, then the fourth fastest direction is called the animation direction. Here the idea is this: If the fiber bundle of surfaces (i.e., a family of surfaces) were to be drawn in three dimensions, then allowed to vary as one stepped through the fourth index set, a set of images that could be likened to an animation would result. This animation would appear as a series of surfaces moving up and down (possibly) in the fiber direction.

4. Mathematical Fundamentals

One goal of the procedures outlined here is to develop a set of measures that would quickly convey attributes of an input data set. These measures would have certain predetermined values only in cases where the data could be imbedded in planar surfaces. Any departure from these values would indicate the presence of 'curvature' or, possibly, wide variations in the data with respect to certain parameters.

One such measure, the "L-infinity" norm (or "essential supremum"), measures the essential magnitude of a member of a special class of functions. This norm was the model for measures M1 and M2, above. In our application, this measure was applied to the height of individual surfaces and to the difference in height between two surfaces of the same family.

Another measure, the variation of the normal vector over the entire data surface, is zero if and only if the surface is flat. The discussion in this section develops the notion of variation of the normal vector and provides some basis for its utility in measuring curvature.

To begin, the data are partitioned into two dimensional lattices in a manner described above. For purposes of discussion, the points on such a lattice will be indexed as a matrix, (f_{ij}). Thus one may refer to the data point in the first row and first column as f_{11}. Since we will be imbedding these data in surfaces, it will be of particular interest to closely examine surface points that occur between actual data values. More concretely, we have the values f_{11}, f_{12}, f_{21}, and f_{22}; these can be envisioned as the heights above the points (1,1), (1,2), (2,1), and (2,2) in the XY plane. The question that now arises is how high is the surface at points not in the original lattice (i.e., at intermediate points)? In order to keep everything simple, these intermediate values are computed by two dimensional linear interpolation. Thus, if s and t are both between 0 and 1, then the surface height above $(1+s,1+t)$ is

$$(1-s)(1-t)f_{11} + (1-s)tf_{12} + s(1-t)f_{21} + stf_{22} \quad (1)$$

In particular, the surface height above (1.5,1.5), halfway between the nearest lattice points, is

$$0.25(f_{11} + f_{12} + f_{21} + f_{22}) \quad (2)$$

The surface is built up by constructing these "patches" from four neighboring lattice points, then attaching these patches to one another at their common borders.

The next step is to construct unit vectors normal to the surface at the center of each patch. Once this has been done, one moves from patch to patch and measures the difference between these vectors and an "average" normal vector. This difference is a measure of the extent to which the surface differs from a plane. For, only in the planar case, all normal vectors are identical and the difference is zero.

Normal vectors are computed by taking the cross products of vectors whose Z-coordinates are the partial derivatives of (1) with respect to s and t. The particular vectors in question are

$$(1, 0, (1-t)(f_{21} - f_{11}) + t(f_{22} - f_{12}))$$

and

$$(0, 1, (1-s)(f_{12} - f_{11}) + s(f_{22} - f_{21}))$$

Their cross product is

$$[-(1-t)(f_{21} - f_{11}) - t(f_{22} - f_{12}),$$
$$-(1-s)(f_{12} - f_{11}) - s(f_{22} - f_{21}), 1] \quad (3)$$

This expression reduces to

$$[1/2(-f_{22} + f_{12} - f_{21} + f_{11}),$$
$$1/2(-f_{22} + f_{21} - f_{12} + f_{11}), 1] \quad (4)$$

when evaluated at the center $(s = t = 1/2)$ of the patch. It is of interest to note that one also obtains (4) by integrating (3) over the entire patch; in this sense, (4) is representative of all vectors of the form (3).

Clearly, the magnitude of (3) and, a fortiori (4), is at least 1. The procedure for measuring the surface curvature calls for calculating the difference between the unit normal vectors at each patch and some average vector for the entire surface. Thus, the next step is

simply to convert (4) into a unit vector by dividing each of its components by the magnitude,

$$[1 + 1/4(-f_{22} + f_{12} - f_{21} + f_{11})^2 + 1/4(-f_{22} + f_{21} - f_{12} + f_{11})^2]^{1/2} \qquad (5)$$

Finally, the average vector for the entire surface is calculated by adding together all unit normal vectors, weighted by the area of their respective patches, and normalizing the resulting sum. Clearly, this "average" vector will be identical to all unit normal vectors only if the surface is a plane. The degree to which it differs is a measure of the surface's curvature.

5. Program Algorithms

The CURV program has three principal subprograms (functions or subroutines). Each contains an algorithm that relates directly to one or more measures discussed above; all are written in Fortran. These subprograms are:

(1) ESSUP1

(2) AVEVEC

(3) VECDIF

ESSUP1, a function, is the most elementary of the three. Its parameter list contains a two-dimensional array whose elements are compared in absolute value. ESSUP1 returns the largest absolute value. Other ESSUP1 parameters are the array's two dimensions, a tolerance level, and a counter. The tolerance is specified by the user to measure the span of the array data. The counter increments each time the absolute value of array element exceeds the tolerance. Thus ESSUP1 yields both the magnitude of the data and the frequency with which the data go beyond a prescribed limit.

Subroutine AVEVEC computes the surface's average unit normal vector. Averaging is performed with respect to the area of the surface patches in the sense described above. AVEVEC's parameters include the two-dimensional array that defines the surface in question, the dimensions of the array, and two vectors that are of use only when grid points are unevenly spaced.

AVEVEC uses the above equations to compute the unit normal vector at each patch. It then call a subfunction to approximate the area of the patch by adding together the area of four triangles that share the center of the patch as a common vertex. Since each normal vector has a non-zero Z component (see equation 4, above), their weighted sum is non-

zero. AVEVEC converts this sum to a unit vector, then returns its components in the output parameter list.

Function VECDIF measures the extent to which each unit normal vector differs from the surface's average vector. At each patch, it computes the square of the magnitude of the vector difference between the local unit normal and the average vector. The weighted sum of squares (again by area) is computed and returned as the function value.

While not one of the principal algorithms per se, subroutine DRIVER is responsible for partitioning the input data and applying the measures to the resulting surfaces. All "bookkeeping" functions reside in this subroutine. In particular, the process of constructing families of surfaces whose members are indexed by the fiber dimension is performed in DRIVER. Similarly, DRIVER coordinates the "animation" of these families and properly sequences the various data dimensions into the fiber and animation roles. The DRIVER input parameters are the array containing the entire data base, a vector of array dimensions, and a scalar equal to the number of components of this vector.

6. Sample Output

The following is a typical output data set from CURV. Each record, after the title record, consists of the following data:

(1) X, Y, fiber, and animation directions. (fields 1-4)
(2) indices of the fiber and animation directions. (5-6)
(3) components of the average vector, M5. (7-9)
(4) variation of the unit normal, M4. (10)
(5) L-infinity norm for a given surface, M1. (11)
(6) L-infinity norm of the difference between two successive surfaces, M2. (12)
(7) running sum of the M2 for a family of surfaces, M3. (13)
(8) count of number of points on surface that exceed prescribed tolerance in absolute value. (14)
(9) array indices that are fixed for current family of surfaces, fiber and animation directions. (15-20)

The measures provide the analyst with a concise description of his or her data. Some measures are scalars that quantify the change in behavior as certain input parameters are varied. Others are vectors that measure global characteristics of the data set. All are simply described and easily visualized.