ORNL/TM-11690

ornl

OAK RIDGE
NATIONAL
LABORATORY

MARTIN MARIETTA

# DCOR: A Deterministic Combat Model Code

Y. Y. Azmy



MANAGED BY
MARTIN MARIETTA ENERGY SYSTEMS, INC.
FOR THE UNITED STATES
DEPARTMENT OF ENERGY

ORNL/TM-11690

Engineering Physics and Mathematics Division

# DCOR: A Deterministic Combat Model Code

Y. Y. Azmy

DATE PUBLISHED — April 1991

Prepared for
U.S. Army Missile Command
Interagency Agreement 1875-D052-A1

Prepared by the
OAK RIDGE NATIONAL LABORATORY
Oak Ridge, Tennessee 37831
managed by
MARTIN MARIETTA ENERGY SYSTEMS, INC.
for the
U.S. DEPARTMENT OF ENERGY
under contract DE-AC05-84OR21400

# CONTENTS

# ABSTRACT

ORNL's deterministic combat model based on a system of PDEs is used to develop a flexible, user-friendly computer code called *DCOR*, for Deterministic Combat model of Oak Ridge, and previously known as WAR. The numerical solution of the PDEs is achieved via the Method of Lines (MOL) which approximates the spatial derivatives on a finite mesh yielding a set of ODEs that are solved using the Gear B Method. A general purpose software based on the MOL called PDETWO is used, with some modifications in DCOR, whereby the diffusion terms are approximated by a five-point finite-difference scheme and the convective terms are approximated by an upwind finite-difference representation. DCOR is an interactive code with graphical display capability for the solution, that permits external control by the user to simulate a wargame environment. A guide to the user of DCOR is included to help in setting up and modifying the input, either interactively or from a file. Preliminary results pertaining to the validation of the deterministic model with respect to a Monte Carlo simulation, to the accuracy of the numerical solution as the spatial mesh is refined, and to the versatility of DCOR is conducting sensitivity analyses are presented.

# 1. INTRODUCTION

Combat situations are characterized by a very complex interaction between means and objectives, decisions and plans, actions and anticipated responses, all of which encompass a multitude of fully or partially stochastic information and covering a wide spectrum of scales, from the individual combatant to an entire war. Such complexity naturally led to the widespread acceptance, and actual usage, of stochastic models to simulate the individual events and decisions, or sequences thereof, occurring during a battle.[1] Clearly, stochastic models have the advantage of being relatively nonabstract and close, in many ways, to the real-life situation they are intended to simulate. Furthermore, input parameters describing various weapon systems involved, and data representing the prevalent environment during the battle, both of which are usually stochastic in nature, are directly measurable or derivable from design specifications and field tests. The major disadvantage of stochastic models is the enormous computational resources (CPU time and memory) they require, which makes it prohibitively expensive to conduct extensive sensitivity analyses.

Alternatively, deterministic Lanchester-type models for combat based on a fully aggregated scale have been used extensively to study the temporal evolution of the attrition of combatants.[2–4] In spite of the relatively good mathematical understanding of the Lanchester model, its extreme aggregated character and lack of spatial detail sets very strict limitations on its practical utility as an attrition/maneuver model. Thus, it is used in combination with stochastic simulations to aggregate the outcome of many small, i.e., fine scale, battles into a larger, i.e., coarser scale level.

Recently, more detailed deterministic combat models have been proposed, developed, implemented and tested at ORNL.[5–9] The basic feature of all these new models is that they introduce refinements to the spatial detail that is entirely missing from Lanchester's model. Thus, the set of Ordinary Differential Equations (ODEs) that comprise the latter, are replaced in the new models by sets of Partial Differential Equations (PDEs) in which derivatives with respect to the spatial independent variables appear, to describe the movement (random or ordered) of the combatants on the battlefield.

Deterministic combat models are motivated by three specific factors. First, if the experience in particle transport computational methods provide any guidance as to the computational performance of stochastic vs. deterministic methods, we must conclude that the latter possess a higher potential for efficiency than the former. Second, there are available analytical results on the sensitivity analysis of sets of PDEs, and symbolic computer codes to perform such analysis, for the benefit of the commander interested in optimizing the battle outcome within limitations of his resources and the battle environment. Third, there exists a wealth of mathematical understanding of PDE systems, such as existence and uniqueness of solutions, bifurcations, stability analysis, chaos, catastrophes, etc., which are not directly extendible to stochastic systems. On the other hand, stochastic simulations are, in general, capable of representing complex geometrical shapes, which in deterministic models would normally require the introduction of a finite spatial mesh on which all geometric shapes must be approximated.

The purpose of the present document is to describe our implementation of the deterministic PDE combat model into a flexible, interactive computer code *DCOR* (for Deterministic Combat model of Oak Ridge) previously known as *WAR*, and some preliminary tests of the accuracy and validity of the model. Section 2 is dedicated to establishing the theoretical foundation of the deterministic model as a competitive system analogous, for example, to interacting gas mixtures. In Section 3 we briefly develop the equations for the PDE model embodied in the DCOR code. In Section 4, we describe the specific implementation of DCOR, and in particular the input and output. Some preliminary results pertaining to the method accuracy with respect to the mesh size, and to the validity of the model itself compared to stochastic simulations, and some examples of sensitivity analyses that are unique to the PDE model are presented in Section 5. Finally, Section 6 contains a brief summary of our effort and an outlook to future developments.

# 2. THEORETICAL FOUNDATION

A model is a mathematical abstraction of physical reality pertaining to a given system to be modeled, aimed at the quantitative and qualitative prediction of the behavior, or response, of that system under the influence of given external stimuli and constraints. The construction of a model is initiated by the realization that the response of a given system to a given stimulus is consistent and repeatable under the same conditions. One approach to building models is to break up the system into subsystems which are easier to model, or for which models already exist. Another possible approach is to use analogies between the system under consideration and previously studied systems sharing similar features and behavioral patterns.

The success of a model is judged by two important qualities: solubility and validity. Solubility does not necessarily require a quantitative, exact or numerical solution of the equations representing the model, but may include qualitative analysis of these equations, e.g., uniqueness proofs. The realm of soluble equations has been expanded dramatically in the last few decades by the introduction and rapid evolution of computers, and by the continuous development, analysis and implementation of numerical methods to take advantage of the latest hardware gear. Model validation is encompassed by the "experimental method" underlying all traditional fields of science, through which solutions to the equations constituting a model are compared to actually measured or observed behavior of the system being modeled.

Fundamental to the validation process are the above mentioned requirements for model construction, namely consistency and repeatability. However, the application of these requirements is greatly relaxed by using statistical methods to interpret the measurements, and sometimes in constructing the model. In particular, physical systems composed of an extremely large number of subsystems pose a very challenging problem for the modeler represented by the large number of degrees of freedom involved. Modeling such a system *exactly* requires full control (or at least knowledge) of the initial settings of *all* the variables, and the subsequent prediction of their values under the influence of applied stimuli; often this is literally impossible. Statistical averages (and higher moments) are used in these cases to simplify both the modeling and the actual measurements, thus enabling the validation process by comparing predicted and experimental values for a few macroscopic quantities. Good examples of such systems abound in traditional sciences, e.g., statistical mechanics, kinetic theory, fluid mechanics, etc.

Another complication arises in the modeling of systems involving living (human or otherwise) subsystems which have the intrinsic capability of making subjective choices among a variety of responses that are often inconsistent and/or unrepeatable. If the modeled system is composed of many such subsystems, and if none of the individual subsystems, and none of the choices available to any subsystem has an overwhelmingly dominant effect on the behavior of the system as a whole, then statistical models can be used here also. Examples of these types of models include younger fields of science, often termed "soft sciences" such as economy models, ecosystem models, attrition models, etc. One of the most profound difficulties with these models concerns their validation; not only is the size and cost of setting experiments prohibitively large, but also controlling the experiment's environment in order to duplicate it exactly is virtually impossible (for one thing, living organisms generally "learn" from previous experience and live a continuous adaptation process).

We now focus the general discussion of modeling to the specific development of combat models, which suffer from all the difficulties described above. Combat situations considered here involve a large number of combatants who make instantaneous and independent decisions while attempting to achieve a global objective determined by their immediate commander in the chain of command. Combat models are extremely difficult to construct, very expensive to solve for realistic nontrivial problems, and practically impossible to validate because of the deficiency of sufficiently detailed and accurate historical or field data. The general approach to modeling combat, until very recently, has been based on a stochastic or semi-stochastic simulation of attrition, and uses game theory analysis to simulate individual, local decisions made by the combatants.[1] The realization of such models into computer codes constitutes the extensive library of "war games",[1] whose members employ various levels of sophistication in simulating, and taking into account, all the variables determining scoring a hit, e.g., line of sight, terrain, weather conditions, etc. The only traditional deterministic model is Lanchester's,[2,3] which, in different variants, has been used primarily as a gross attrition model since it lacks much of the detail necessary in modeling the outcome of combat.

Recently, a more detailed, deterministic combat model based on PDEs has been proposed as an alternative to stochastic models.[5,6] Initial implementation of the PDE model and preliminary results indicate its versatility and its high potential.[7–9] In the remainder of this section we justify using a deterministic method to model an essentially stochastic process by drawing an analogy with mutually interacting gas mixtures, which can be represented at the microscopic level by deterministic, Boltzmann type PDEs, and at the macroscopic level by reaction diffusion equations. In addition to establishing a preliminary philosophical platform for a rigorous derivation of a kinetic theory of combat along the same line of approach used in classical statistical mechanics, this view point can be useful in providing a clearer correspondence between the measured attrition parameters of various individual weapon systems, and the attrition coefficients used in PDE combat models.

In representing an essentially stochastic system, such as combating species or gas mixtures, by deterministic models, one makes the implicit assumption that the system is made up of a large number of non-individual subsystems, which under similar circumstances behave in a statistically consistent manner. By this we mean that the outcome of events, e.g., shooting in combat, or chemical reaction in a gas mixture, is independent of the individual subsystems involved in it, and is describable by a probability distribution. Thus a deterministic representation is sensible and accurate only when the simulated system involves a large number of subsystems of more or less equal importance, and a large number of simultaneous and sequential events of comparable effect on the entire system. The transition from stochastic reality to deterministic representation in such cases is mostly philosophical, but is operationally justifiable as follows. For the system described above, the temporal evolution of a specific initial setting of such a system depends on this initial setting, and on properties and behavior of the individual subsystems. Accordingly, there is an extremely large number, "ensemble", of possible evolutions each requiring a certain chain of events occurring in a particular order. It is neither possible nor desirable to exactly determine the initial setting or the specific member of the ensemble realized in a given experiment. Rather, just like in hard sciences, e.g., fluid mechanics, averaged quantities over the members of the ensemble are of interest, as they predict the gross behavior of the system, by "averaging-out" peculiar, or highly improbable members of the ensemble. Indeed, in stochastic models based on Monte Carlo simulations, the outcome of a single experiment has very limited value; for example, it can be useful as a training tool, but since it can produce a statistically improbable outcome, it cannot be used for planning purposes. Normally, several numerical experiments are performed in this case, and the results are averaged over the various experiments to produce "expected values" for the desired quantities. Deterministic methods, such as PDE combat models, aim at directly calculating these expected values without the expensive Monte Carlo simulations.

In view of the above discussion, the analogy between a combat situation and a mixture of interacting gases, or molecular systems on the microscopic and macroscopic scales becomes evident. Microscopically, the various types of combatants, e.g., troops, tanks, artillery, etc. are analogous to the molecular species, each represented by a distribution function; the attrition rate of one combatant by another's fire is analogous to a removal interaction cross section; the combatant's command (i.e., mission) and the terrain effects are analogous to the external force field acting on the particles in the gas mixture. Macroscopically, each type of combating unit is analogous to the corresponding gas species space and time dependent mass density; the attrition rates are analogous to chemical reaction rates; and commands and terrain effects are analogous to convective, bulk movement of the gas species. The random fluctuation in the exact position, velocity, kill effectiveness of weapons, individual local decisions, are represented by the different members in the ensemble on the microscopic scale, and averaged-out of the macroscopic model (except in the diffusive terms). Thus, it is possible to model combat using deterministic methods at the microscopic, molecular scale, or the macroscopic scale. The microscopic model is more elaborate and computationally intensive to be used at this early stage of investigating deterministic combat models, even though it is instrumental in deriving a paradigmatic analogy between combat and gas mixture interactions. Hence, only macroscopic models analogous to those used in reaction-diffusion systems are used in the remainder of this work. At a later stage, interest in microscopic models may arise because they may be capable of providing better modeling accuracy, and also to compute better approximations to the macroscopic parameters from the raw data available for various weapon systems from the manufacturing specifications and field tests.

# 3. EQUATIONS FOR THE PDE COMBAT MODEL

The macroscopic reaction-diffusion type equations for the PDE combat model have been presented and discussed in fair detail elsewhere.[5−9] However, for the sake of completeness and to introduce the variables encoded in DCOR, described in the next section on input and output specification, we briefly discuss the equations and the terms involved in them.

Let the battlefield be a rectangular region in the x-y plane, so that $(x, y) \in [0, X] \times [0, Y]$. According to the analogy derived in Section II, the spatial disposition of each combatant type, or weapon system, is represented by a density function $u_m(x, y), \quad m = 1, \ldots, M$, where M is the total number of weapon types involved in the conflict. Notice that it is not necessary to specify which side of the conflict each species belongs to, as this is determined exclusively by the attrition rates discussed below. The random, unordered, motion of each species is described by a diffusive, second derivative term of the form $-\nabla \cdot (D_m \nabla) u_m$. Normally, diffusive motion should vanish for stationary weapons, e.g., large caliber cannons, but are of small magnitude for mobile weapons, e.g., tanks, and of yet slightly larger magnitude for highly mobile systems, e.g., infantry troops. Furthermore, diffusive behavior of the species depends on human factors, such as freshness of troops, order, or lack thereof, fear, morale, etc. For example, troops retreating in defeat should exhibit more pronounced random movement, than troops charging in an ordered, apparently successful attack. Diffusion effects, in general, should also be space-dependent; troops on the front line, directly exposed to the enemy's fire should be expected to display more diffusion than protected or shielded forces.

The ordered bulk, or convective movement of the combatants on the battlefield is proportional to the product of the first order derivatives with respect to the spatial variables and the velocity component in the corresponding direction: $\bar{v}_m \cdot \nabla u_m$. Because the convective velocity $\bar{v}_m$ is practically independent of the density (but possibly space and time dependent) the convective terms are linear. The convective velocity is eventually imposed by the commander trying to achieve a specific objective but is limited beyond his control by design specification, human limitation, terrain features, enemy action, etc. All these factors can be taken into account external to the PDEs, so that linearity of these terms is preserved. The equations for the PDE combat model become,

$$\frac{\partial u_m}{\partial t} = -\nabla \cdot (D_m \nabla u_m) + \bar{v}_m \cdot \nabla u_m - A_m, \quad m = 1, \ldots, M, \tag{3.1}$$

where $A_m$ is the total attrition rate of the m-th weapon system modeled as discussed below.

The total attrition rate is composed of local attrition, $L_m$, and nonlocal attrition, $N_m$, such that $A_m = L_m + N_m$. Local attrition occurs when two or more combatants exist at the same physical location and fire at one another. Strictly speaking, no two combatants can exist at exactly the same location simultaneously; however, this is possible in the framework of the continuum PDE representation of the ensemble of discrete reality as discussed in Section II. Essentially, it represents an average over members of the ensemble in which one of the two combatants occupies the given location and the other is nearby, and members of the ensemble in which the two combatants exchange locations. Local attrition is more likely in hand-to-hand combat, such as in historical battles, small fire arms conflicts, urban warfare, etc. Also, in large-scale modern warfare, and within the framework of discretized (in space) numerical method approximations to the PDE model, the range of weapons which normally would be considered nonlocal may become small relative to the scale of the entire battlefield, or compared with the spatial mesh size. In such cases, these weapons can be approximately represented by local attrition.

In the present model, we assume that the distribution of the combatants over the battlefield is sufficiently sparse to warrant ignoring the simultaneous local attrition of one species by two or more different types of weapons; clearly such an extension would be straightforward if deemed necessary. Even though the probability of scoring a hit depends heavily on human factors for the operator of a given weapon, as well as on the characteristics of the weapon, and of the target, the randomness of the hit (or miss) event produces a probability distribution which we use to calculate a statistically expected value for the attrition rate. Since the amount of fire depends on the local density of the firing weapons, e.g., $u_n$ and the number of hits (or rather the amount of damage to the target) depends on the local density of the target weapons, e.g., $u_m$ we model the combat local attrition rate by $\alpha_{mn}u_m u_n$. The attrition parameters, $\alpha_{mn}$, are real and positive functions of space and time; they can be turned off, i.e., set to zero, by the commander and can be controlled to increase continuously up to an upper limit dictated by physical performance constraints, and by specific circumstances in the battlefield. In case combatant m is being attrited by more than one enemy weapon type, the attrition rate will be the sum $\sum_{n=1}^{M} \alpha_{mn}u_m u_n$. (For example, if $\alpha_{mm} \neq 0$, the model takes into account accidental "friendly fire", which normally should be kept very small.)

The attrition rates described above are quadratic in form. For the sake of mathematical completeness, because it is computationally inexpensive, and also based on some, maybe less, convincing arguments we include in the local attrition rates constant and linear terms. Thus,

$$L_m = u_m \sum_{n=1}^{M} \alpha_{mn}u_n + \sum_{n=1}^{M} \beta_{mn}u_n - \gamma_m, m = 1, \ldots, M \quad . \tag{3.2}$$

The constant term in Eq. (3.2) can be viewed as an external fixed source, for example, representing supplies ordered by a higher level in the chain of command. The linear term corresponding to $n = m$ can represent natural attrition due, for example, to a hostile environment, such as severe cold or heat, extreme drought, high elevations, sand or snow storms, etc. The linear terms corresponding to $n \neq m$ formally replicate the attrition terms in Lanchester's *aimed fire* model.

Modern weapons introduce nonlocal attrition into combat, whereby the firing weapon and the target can be separated by extensive distances relative to the scale of the conflict. Artillery, and missiles are good examples of such weapon systems. Nonlocal attrition is more difficult to model than local attrition, because one must take into account weapons firing from all possible locations at targets located anywhere on the battlefield. Thus,

$$N_m = u_m(x,y,t)\sum_{n=1}^{M}\int_0^X dx'\int_0^Y dy' K_{mn}(x,y,t;x',y')u_n(x',y',t), \quad m=1,\ldots,M \ , \tag{3.3}$$

where the attrition kernel, $K_{mn}$, is real and positive. The nonlocal attrition rates in Eq. (3.3) are quadratic; because the spatial integration included in Eq. (3.3) is computationally expensive, we did not include a linear nonlocal attrition term as was done in Eq. (3.2).

It is evident from Eqs. (3.1–3.3) that the combat model is composed of $M$ PDEs in the $M$ densities of the combatants. Solving these equations requires specifying initial and boundary conditions on each density. The initial condition arises normally as the spatial disposition of the opposing forces over the battlefield at the beginning of hostilities. The boundary conditions are more elusive since they are essentially a mathematical abstraction with no relation to reality, since battlefields have no absolute boundaries, only terrain features. The most general type boundary conditions that can be used for the PDE system (3.1) are of the form,

$$a_m u_m(0,y) + b_m(\partial u_m/\partial x)_{(0,y)} = c_m \ , \tag{3.4}$$

and analogous conditions on the $x = X$ and $y = 0$ and $Y$ boundaries. Unlike natural sciences, where physical constraints translate into values for $a_m, b_m$, and $c_m$ in Eq. (3.4) thus providing an unambiguous boundary condition, in combat points on the edge of the battlefield are not any different from interior points. To circumvent this problem, we assume that it is always possible to choose the battlefield large enough so that the effect of the boundary conditions on the progress of the battle is not detectable. If during the evolution of a battle situation, one or more of the combatants move too close to the battlefield boundary, it should be wise to terminate the simulation at this point, then restart it on a battlefield large enough so that the boundary effects do not influence the combatants.

One aspect of combat, and indeed of most human activities, that has not been addressed intrinsically within the PDE system is decision making. Individual decisions made on a local scale by a combatant can be modeled via an interplay between stochastic events and game theory analysis. As discussed in Section II, the fluctuations in the gross progress of the battle resulting from individual decisions is averaged out in the deterministic PDE model. Higher level decisions, by the commanders for example, which have an overwhelming effect on the progress of the battle, such as charging/retreating, firing/holding fire, etc., are not modeled internally within the PDE system. Rather, limited control of the model parameters affected by these decisions is permitted to “players” commanding the combatants in a wargame environment. Limits are set on such control in order to avoid violating physical or human limitations on the weapons and their operators, respectively, such as the maximum speed of a vehicle, or the maximum range of sight of a soldier. Major decisions at this level can be analyzed using methods of sensitivity analysis and game theory in order to define and evaluate strategic options available to the commander at a given point during the battle.

# 4. IMPLEMENTATION INTO THE COMPUTER CODE DCOR

The combat model described in Section III is comprised of a set of $M$ nonlinear PDEs of the form Eq. (3.1) each with an initial condition and boundary conditions exemplified by Eq. (3.4). Numerical solution of these PDEs requires introducing approximate methods as discussed in this section. The solution algorithm has been implemented in the DCOR code, which has the following characteristics: (1) two-dimensional maneuver capability; (2) heterogeneous force representations; (3) space-time dependent local and nonlocal attrition; (4) interactive modification of parameters during engagement; (5) flexible interactive input preparation; (6) dynamic memory allocation; and (7) interactive graphical display of the evolution of the battle. DCOR has been developed, and is presently operational on ORNL's Cray X/MP running the UNICOS operating system. However, in order to enhance portability of the code, it has been written in standard FORTRAN, so that migrating it to other computers and operating systems should require only minor modifications. The only possible exception is the plotting routine which requires the availability of the DISSPLA graphics library; even then, all the graphics is performed in one routine, so that translation of DISSPLA commands to those of any other graphics software can be done with relative simplicity.

The Method of Lines[10] (MOL) has been developed and implemented for a variety of applications in order to take advantage of the Gear B[11] method for solving first order ODEs. The Gear B method is particularly useful in dealing with stiff problems, and is capable of virtually unlimited accuracy as prespecified by the user. The MOL is based on approximating the spatial differential operator on a finite mesh, thus producing a set of ODEs whose independent variable is time, and which are solved simultaneously via Gear's method. The MOL has been implemented in a general purpose software (in which the spatial operator is discretized via the finite-difference approximation) called PDETWO which we used in the DCOR code as the temporal evolution routine. This choice (as opposed to a full finite-difference discretization) was made because of the special importance of the accuracy of the time dependence in this problem, compared to that of the spatial dependence. PDETWO is slightly modified in order to accept the flexible dynamic memory allocation implemented in DCOR via the use of container arrays, and also to permit using space dependent convection speed not available in the original software. Subroutines required by PDETWO to evaluate the RHS of the set of ODEs at various points in space, and the boundary conditions on the edges are provided in the most general form of the PDE combat model. These subroutines essentially implement the finite discretization of the reaction-diffusion-convection spatial operator, by using a five-point finite-difference representation of the second order diffusion operator. Due to the space-dependent convection field feature we added to PDETWO (which is essential for modeling maneuver) it is necessary to introduce an upwind representation of the convective terms, which has been shown to be unconditionally stable in similar fluid dynamics problems.

The remaining subroutines in DCOR handle I/O, both interactive and from files, generate graphical metafiles using DISSPLA library, and call PDETWO to generate the time evolution of the battle. The code is structured to run in time batches, the length of which, and the number of time steps within, are interactively determined by the user. Before the first time batch starts the code inquires if the input data is to be fed-in interactively or from an existing file named *wari* described below. If the user chooses to enter input interactively, the code inquires about the value of each of the input variables with a brief descriptive message. All the *read* statements are unformatted so the input can be supplied in any desirable format; this is particularly useful in supplying large arrays of data. The input data is printed in the output file *waro*.

After reading-in the input data, the code compares the total size of the problem defined by that input and compares it to the hardwired size of the integer- and real-variable container arrays used in DCOR. If there is not enough room to perform the calculation, execution is terminated, and the code prints out the necessary sizes for the container arrays to accommodate the problem described by the input. Otherwise, and also for all subsequent time batches, the user is required to decide whether to start a new time batch, or stop execution at that point. Stopping execution will prompt the code to write the current status of the battle into a file *warcont* in exactly the same form in which *wari* is expected, that can be used for later continuation of the battle, and terminates the plotting device and closes the metafiles. On the other hand, if the user decides to run a time batch, the code interactively inquires about the number of time steps in the batch. If the number of time steps is negative, the code next expects the time levels at which it is to calculate the forces' disposition on the battlefield. If the number of time steps is positive, the code next expects the final time level for the batch, and divides the length of the batch (i.e., the difference between its final and initial time levels) into equal intervals. In general, the larger the time step size, the longer PDETWO takes to perform the calculation for that step; however, for a fixed time batch length, it is more efficient to use a small number of time steps.

During a time batch the code executes in uninterrupted mode. Only at the beginning of a time batch are the users allowed control over their respective forces. Thus, a time batch starts with a menu of choices allowing the user to modify the parameter settings for the new time batch as follows:

- t: Change the title of the plots; this can be useful in identifying the various stages of the battle, i.e., attack, maneuver, retreat, etc.

- m: Change the position, but not the number, of the $x-$ and $y-$mesh points; this can be useful in improving mesh resolution at selected points while the battle evolves (only partly functional).

- f: Change the spatial distribution of the forces on the battlefield; this permits the user to regroup or redeploy forces to take advantage of the present status of the battle, terrain features, *implement strategic actions*, etc. In doing so, the user can conserve, or not conserve the total number of units of each force type to simulate reinforcements arrival or gradual retreat. Also, the user is allowed to change the allegiance of each force type to simulate betrayal or mutiny.

- p: Change the diffusion constant for each type to simulate morale status, prevalence of order, lack thereof, etc. Also permits changing convection velocity components, which is necessary to simulate maneuver, attack, retreat, change in terrain features, etc.

- a: Change local and nonlocal attrition rate data for each force; this gives the user control to fire or hold fire of certain force types during various stages of the battle.

- b: Change the boundary condition coefficients for each force type.

- c: Change contour plot data, namely the contour base and constant increment between contour levels; this can be useful if a large change in force levels occurs during battle, e.g., extensive attrition, then reducing the base and increment can provide spatial detail that otherwise will be lost. Also, permits changing the increments on the $x-$ and $y-$axis labels.

- e: Each of the above entries allows the user to modify the indicated quantities then returns to the modification menu allowing further modification of other quantities. The "e" entry would allow the user to modify everything on the list without going back to the modification menu between items.

- r: Returns to execution mode, thus running the time batch with the modified variables. No further modifications permitted before the end of the time batch.

- x: To exit DCOR before running the present time batch.

Each selection in the modification menu includes a submenu that helps the user make the desired modifications, and always includes an "r" entry to allow the user to return to the modification menu. Clearly, modifications during the battle must be subject to physical and human limitations which must be strictly applied to changes as they are requested. This can be done with trivial modifications to the code, but requires knowledge of these data, which are mostly classified, and therefore unavailable to us.

For each time level within a batch, PDETWO calculates the spatial distribution of the density of each force-type over the entire battlefield, and generates a contour plot of it, with a different color assigned to each side. Also, at the end of each time step the total number of remaining units in each force type is calculated (equal to the product of the density and the area associated with a mesh point, summed over the entire grid), and the total is printed out on screen and in the output file *waro*. Once a time batch is concluded, the user is required to choose whether or not to start a new time batch, and the process continues. If the user terminates execution, the wall-clock and CPU times are printed on-screen and in the file *waro* before stopping.

There are two useful options to use when entering the elements of large arrays, either interactively or from file *wari*. The code possesses two modes for reading each large array: full and non-full; in interactive mode the user is prompted to make the choice, while in file *wari*, the line preceding each such array must start with the character "f" or "n" for full or non-full array entry. The user may choose full entry in case of uniform data, e.g., 1000*0 is the equivalent of entering 1000 zeros for the elements of some array. Also, the output file printed at the end of execution for the purpose of continuing the battle in a later run contains all arrays printed in full form since the code has no means of distinguishing uniformity patterns. In most all other cases the user would opt for the non-full mode, where it is assumed that the array is made of uniform blocks covering the entire mesh, with all uninitialized elements set to zero on the UNICOS Cray. Each block is specified by two lines of input, the first contains the *nonzero* uniform magnitude of the block (note that zero-valued blocks need not be specified as they are the default), and the second contains the initial and final $x$−mesh point indices followed by the initial and final $y$−mesh point indices. There is no limit on the number of blocks that can be specified this way, and blocks or sub-blocks can be reassigned nonzero values in the same run. Entry of blocks is terminated with a zero on the magnitude line.

Finally, we describe the structure of the input file *wari*. Every entry, array or scalar is preceded by a one-line message in which the user can describe the following input in terms most understandable to him. The content of these messages is irrelevant since they are not read by the code; all that is necessary is that each two entries be separated by one line. The novice user wishing to generate an input file is urged to try one of two approaches: use an existing sample file as a blue print for the format, and change the variable values to those specifying his input, or generate the first input file interactively then use it as a blue print as before. Following is a list of the entries in *wari* with a brief description of each variable. The italic comments preceding each entry is suggestive of the descriptive comments mentioned above, and in most cases are identical to those generated by the code in the output file to be used for battle continuation, *warcont*.

*Numbers of x-, y-mesh points, and forces:*

mx: Number of mesh points in x-direction, i.e., (mx-1) intervals.

my: Number of mesh points in y-direction, i.e., (my-1) intervals.

mfrc: Total number of separately modeled forces or weapon types on all sides of the battle.

*PDE solver parameters:*

meth: 1(2) selects Adams (backward) differencing method. Recommended choice 2.

miter: 0 selects functional iterations;

1 selects the chord method with the Jacobian provided by the user in subroutine PDB;

2 selects the chord method with a finite-difference approximation of the Jacobian; recommended;

3 selects the chord method with a diagonal approximation in place of the Jacobian; recommended for faster performance.

morder: the maximum order of the method used in Gear B, and should not exceed 12(5) if meth=1(2); maximum order recommended.

title: A string of 40 or fewer characters *always* ending with "$", to be displayed on top of each plotted frame at each specified time level. The title should start on a new line, use "(" for uppercase, ")" for lower case, which is the default; $, (, and ) are included in the 40 characters count.

*non-uniform x-mesh*

x(i): x-coordinate of the non-uniformly spaced mx mesh points in the x-direction; alternatively, if the mesh is uniform in the x-direction, the previous line should start with the character "u" and in this case the code will read only x(1) and x(mx) then generate the (mx-2) interior points so the spatial intervals are equal in size.

*non-uniform y-mesh*

y(j): y-coordinate of the non-uniformly spaced my mesh points in the y-direction; similar provision as above for uniform y-mesh spacing.

*distribution of forces among sides*

mdist(m): mfrc integer values specifying the side of the battle each heterogeneous, or weapon type, belongs to; this has no effect on the model or the numerical solution procedure, but affects same color plotting of forces on the same side. For clarity reasons the grid is plotted in white on a black background, and the combatants are plotted in yellow for one side and in cyan for the other.

*initial force distribution:*

*full specification of force 1 array*

u(1,i,j): mx × my array specifying the initial spatial density distribution of force type 1; non-full specification using the format described above available if the first character in the previous line is "n".

⋮

*full specification of force mfrc array*

u(mfrc,i,j): Initial condition for force type mfrc. Note that full/non-full data entry is independently selected for each force by entering an "f" or "n" on the line preceding data entry.

*diffusion coefficient*

d(m): mfrc values for each force's diffusion coefficient; these should be positive, and relatively small in magnitude. Spatial dependence of the diffusion coefficient has not been implemented yet.

*Convection velocity components:*
*full specification of x-component for force 1 array*

cx(1,i,j): mx × my array specifying the space dependent x-component of the velocity field for force 1 in full mode; non-full mode data entry available.

⋮

*full specification of x-component for force mfrc array*

cx(mfrc,i,j): mx × my array specifying the space dependent x-component of the velocity field for force mfrc.

*full specification of y-component for force 1 array*

cy(1,i,j): Analogous to cx(1,i,j).

⋮

*full specification of y-component for force mfrc array*

cy(mfrc,i,j).

*full specification of external source for force 1 array*

s(1,i,j): mx × my array specifying the space dependent external, or fixed source for force 1 in full mode (non-full mode available). This is the same as $\gamma_1$ in Eq. (3.2).

⋮

*full specification of external source for force mfrc array*

s(mfrc,i,j).

*local linear interactions for force 1*

al(1,m): vector of length mfrc specifying the *spatially constant* local attrition rate of force 1 by all other forces, designated the symbol $\beta_{1n}$ in Eq. (3.2); only full mode input available.

⋮

*local linear interactions for force mfrc*

al(mfrc,m).

*local quadratic interactions for force 1*

aq(1,m,n): mfrc × mfrc array specifying the *spatially constant* local attrition rate of force 1 due to the coexistence of force types m and n locally; only full mode input available.

⋮

*local quadratic interactions for force mfrc*

aq(mfrc,m,n).

*nonlocal interactions for force 1*

an(1,1): The nonlocal, quadratic, *spatially constant* attrition rate of force 1 at a given point by force 1(!) at another point.

r0(1,1): The minimum distance between the attriting and attrited forces for *an* to be effective.

r1(1,1): The maximum distance between the attriting and attrited forces for *an* to be effective; an, r0, and r1 can be entered on the same line.

an(1,2),r0(1,2),r1(1,2).

⋮

an(1,mfrc),r0(1,mfrc),r1(1,mfrc).

⋮

*nonlocal interactions for force mfrc*

an(mfrc,1),r0(mfrc,1),r1(mfrc,1).

⋮

an(mfrc,mfrc),r0(mfrc,mfrc),r1(mfrc,mfrc).

*BCs for force 1*

bc(1,1,1), bc(1,2,1), bc(1,3,1): The coefficients for the top boundary condition for force 1, denoted in Eq. (3.4) by $a_1, b_1$, and $c_1$, respectively.

bc(1,1,2), bc(1,2,2), bc(1,3,2): The coefficients for the right boundary condition for force 1.

bc(1,1,3), bc(1,2,3), bc(1,3,3): The coefficients for the bottom boundary condition for force 1.

bc(1,1,4), bc(1,2,4), bc(1,3,4): The coefficients for the left boundary condition for force 1.

⋮

*BCs for force mfrc*

⋮

bc(4,1,4), bc(4,2,4), bc(4,3,4).

*contour parameters*

zbase: base (or minimum) value at which contour lines are plotted.

dcont: increment between contour lines; the smaller the magnitude of *dcont*, the closer the contour lines are to one another. These parameters do not influence the calculation itself, and can be entered on the same line.

*x- and y-axis increments*

delx: Distance (in units of x) between x- labels on the contour plot.

dely: Same as *delx* for y-axis. These parameters do not influence the calculation and can be entered on the same line.

This concludes the description of the input file *wari*. The DCOR Code, and hence all input data, has been written in dimensionless units, so that the only restriction on input data is that it all have consistent units.

# 5. SOME PRELIMINARY RESULTS ON VALIDITY, ACCURACY, AND SENSITIVITY

In this section, we present some preliminary numerical results we obtained from DCOR and discuss their impact on validating the code (and model), and on the accuracy of the spatial approximation. Also, we demonstrate the utility of the code in performing sensitivity studies that the field commander should use when making decisions on the battlefield.

Validation of combat models and wargame codes against actual measured data is practically impossible due to the lack of detail, and often large uncertainties in historic battles data. Validation of such codes, therefore, require the development of special validation philosophies, and possibly defining validity norms specially designed to take into consideration the peculiarities of combat and the lack of detail in historic data. However, this is beyond the scope of this work. A simpler approach which we have adopted[12,13] is to validate our deterministic model code against the results of the Monte Carlo simulation, wargame code, JANUS.[1] The justification for such an approach is that the two approaches represent vastly differing solution philosophies to the same problem, so that the agreement of their respective results can loosely be interpreted as validation of each against the other. Alternatively, one can think of the Monte Carlo simulation as the mathematical twin of an actual battle that indeed does not require validation, as long as the individual events and attrition rates are valid. From this point of view, the comparison with JANUS results would constitute a validation of the deterministic model.

The problem of validation was made more difficult for us by the fact that the data, i.e., attrition coefficients for current weapons, and even results of wargame simulations are mostly classified. One exception was an engagement scenario in which a Red Force consisting of a tank regiment executes a frontal attack on a Blue Force consisting of M60A3 tanks and M901 vehicles carrying TOW antitank missiles in a prepared defensive position.[14] The JANUS results for this battle simulation have been published, even though the attrition data for each of the participating weapon systems is not available. Hence, we reduced our validation attempt to showing that for properly tuned attrition coefficients the deterministic model is capable of displaying a temporal behavior of the total number of remaining units in each weapon type as the Monte Carlo simulation. Clearly, a much stronger validation is necessary, and will be performed, when attrition data for this case are available, or when a completely artificial battle fought with imaginary (thus unclassified) weapon systems is solved by the two approaches, and the results are compared.

The purpose of the JANUS example, called the *Staggered Defense Scenario*, which we use for our validation process, is to evaluate the human performance and casualty criteria of the Blue Force as a function of various nuclear weapons laydown conditions. The simulation, i.e., force types, number ratios, etc., is tailored so that the blue force is the consistent winner in the absence of nuclear detonation. Solutions to this situation then provides the base reference case, and the effect of the nuclear element is determined by comparing solutions to the various nuclear laydown conditions to this reference solution. Clearly, our interest lay completely in the conventional arms reference case, for which the initial disposition of the various weapon types is shown in Fig. 1 adapted from Ref. 14. The interested reader will find more details on the specifications of this battle scenario in Ref. 14. For our purposes, all that matters is the number of remaining units for each weapon type as a function of time, which has been obtained in Ref. 14 by averaging the results of three independent JANUS runs to produce an approximate expected value for these quantities. A comparison between the JANUS and DCOR results obtained on a 15 × 9 mesh for the Staggered Defense Scenario is shown in Fig. 2. The comparison indicates a very good agreement especially in the early stages of the engagement. The relatively larger disagreement between the two methods' results towards the end of the battle can be blamed on two factors: unreliability of the stochastic simulations when only a few units are left on Red's side; and inaccuracy of the deterministic method solution at large time levels due to numerical diffusion. Needless to say, this attempt at validating the DCOR code and the deterministic model is very crude and preliminary. The availability of unclassified, even if unreal data and battle simulations, will immediately prompt a more rigorous, in-depth comparison between the two methods.

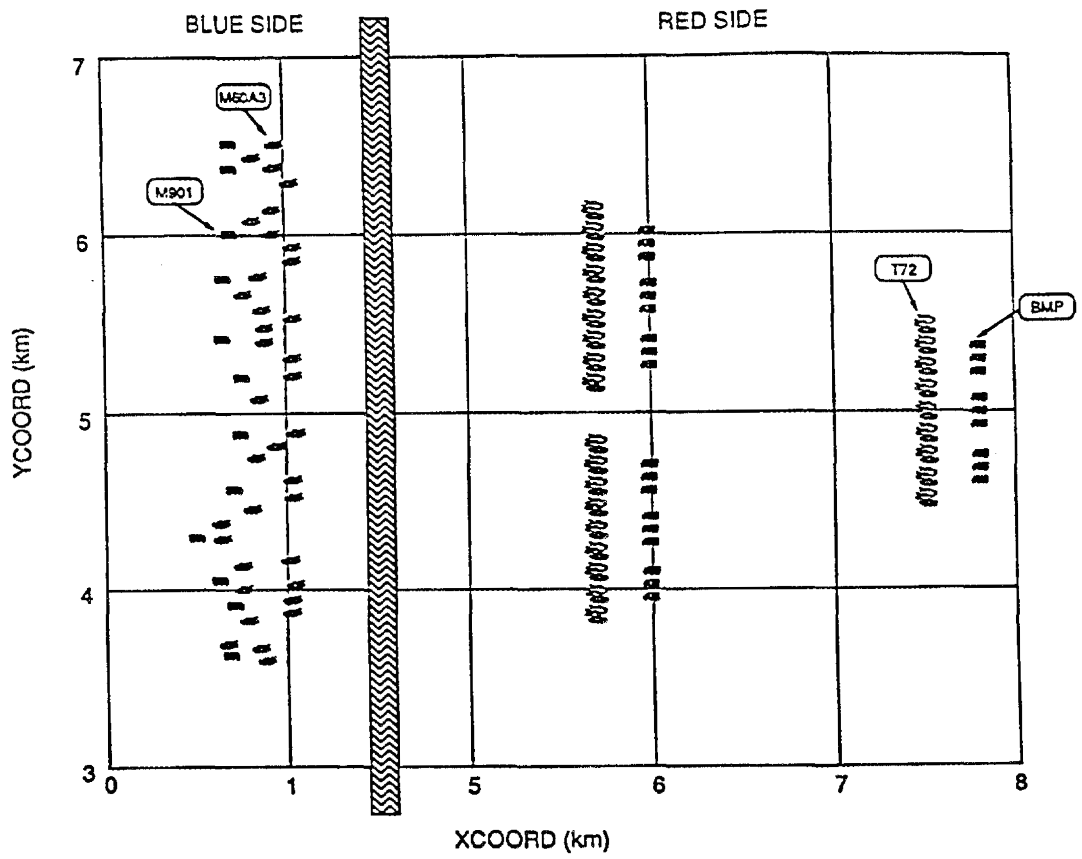



**Fig. 1. The Staggered Defense Scenario; adapted from Ref. 14**

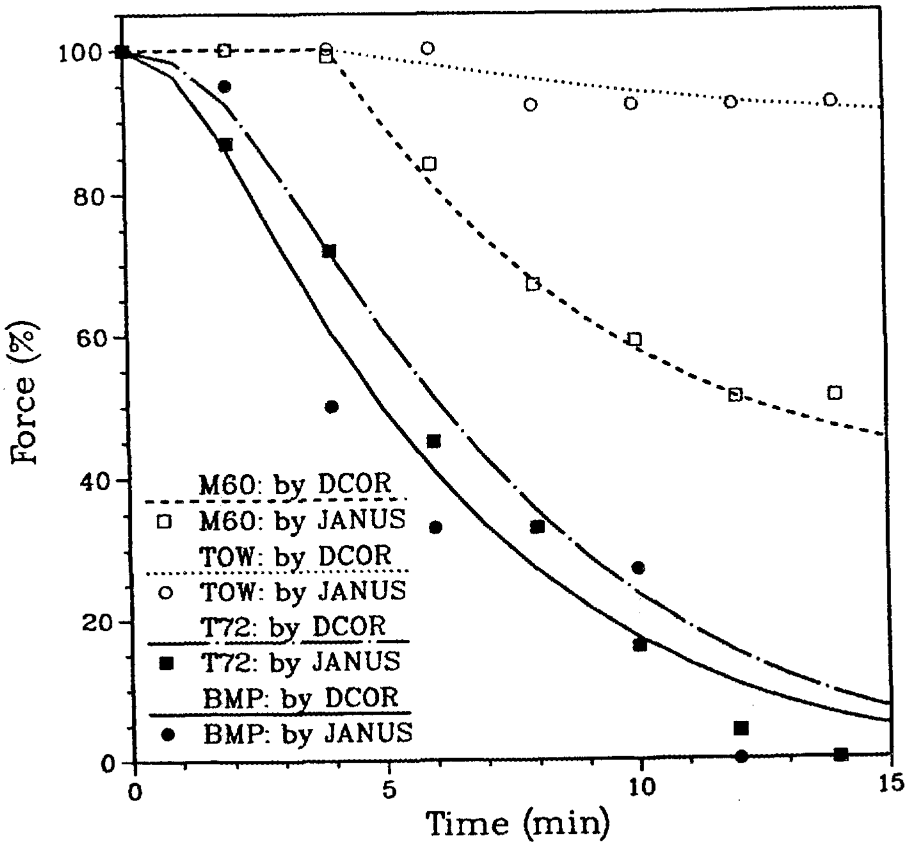
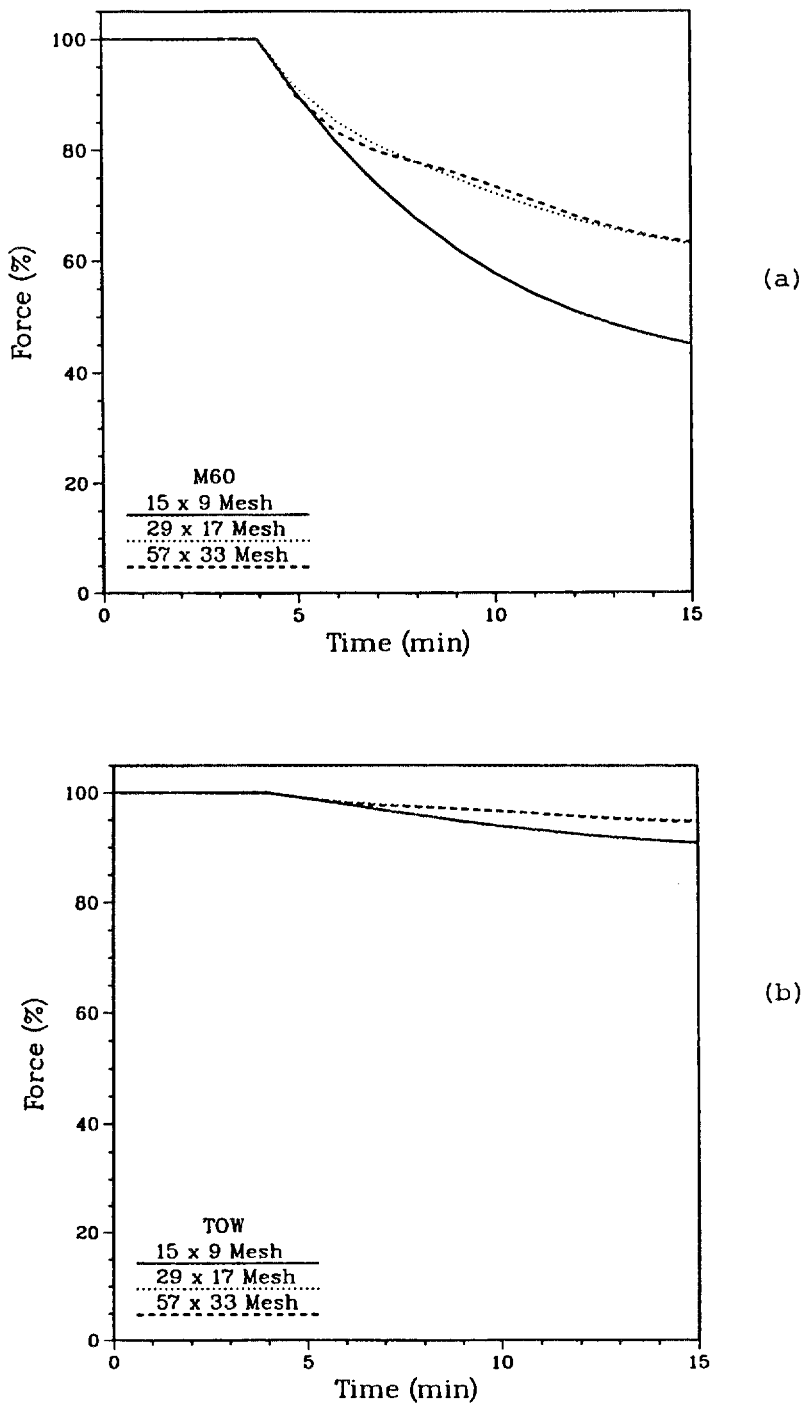
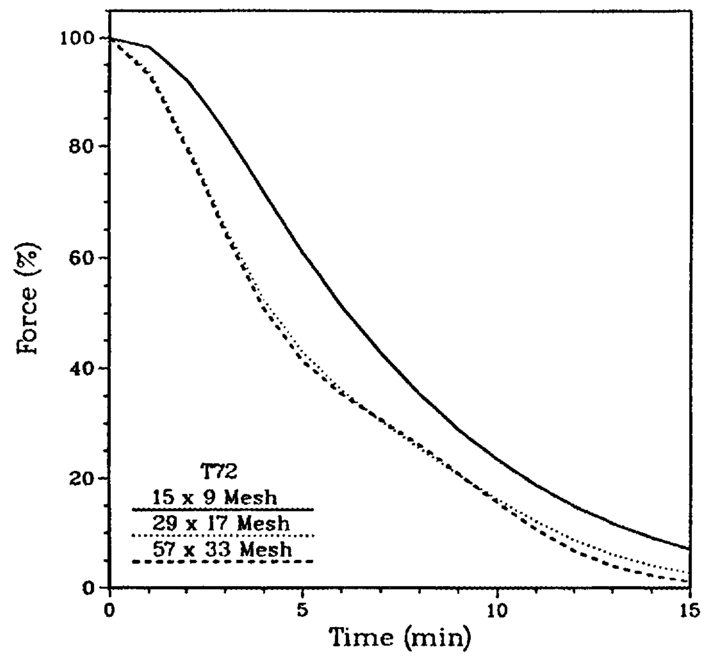


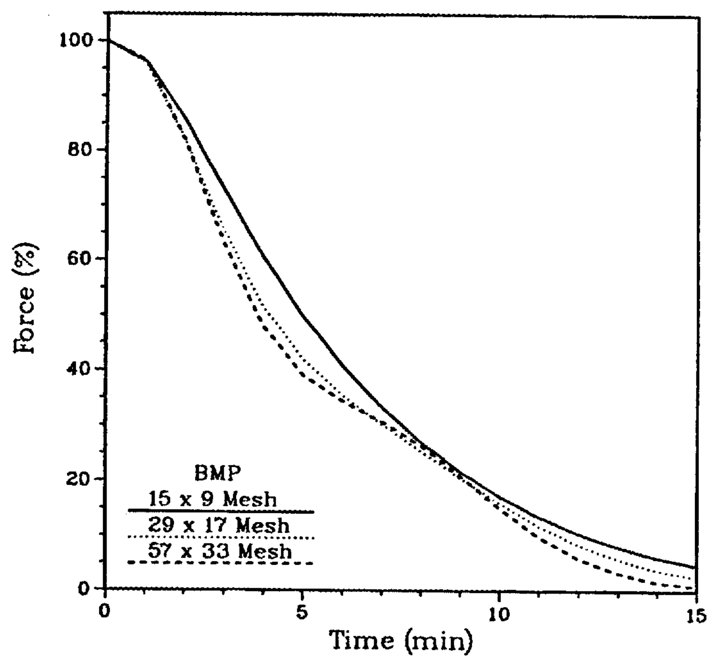

**Fig. 2. Percent remaining forces as a function of time for the Staggered Defense Scenario as calculated by DCOR and by JANUS.[14]**

The accuracy of the spatial, finite-difference, approximation is an easier, though computationally nontrivial task. In order to estimate the accuracy, we solved the Staggered Defense Scenario on $29 \times 17$ and $57 \times 33$ uniform meshes, which correspond to one-half and one-fourth the computational cell size per dimension for the $15 \times 9$ mesh case discussed above. The initial disposition of forces in the $29 \times 17$ and $57 \times 33$ cases is identical, but is slightly different from that used in the $15 \times 9$ mesh case, where the coarseness of the mesh obscures some of the details of the initial condition. The time evolution of the total number of remaining units for each of the four weapon systems (two Blue and two Red) as calculated on the three meshes described above are presented in Figs. 3.a–d. Again, the results agree considerably well at the beginning and deteriorate towards the end of the battle due to numerical diffusion and accumulation of errors as time evolves. It is interesting to note, however, that on the two finer meshes, with the more accurate representation of the initial condition, the end-of-battle total number of T72 tanks and BMPs is closer to that predicted by JANUS. As a sample of the graphical display sequence depicting the time evolution of the spatial distribution of the density of each of the four participating force types at t=1, 4, 6, and 11 minutes into the battle, calculated on the 57 × 33 mesh is shown in Figs. 4.a–d, respectively.






(c)




(d)

**Fig. 3. Accuracy of the temporal evolution of the percent remaining force for each weapon type in the Staggered Defense Scenario as calculated by DCOR on three meshes. The large discrepancy between the 15 × 9 mesh results and the fine mesh results is mainly due to the difference in initial conditions dictated by the coarseness of the former.**

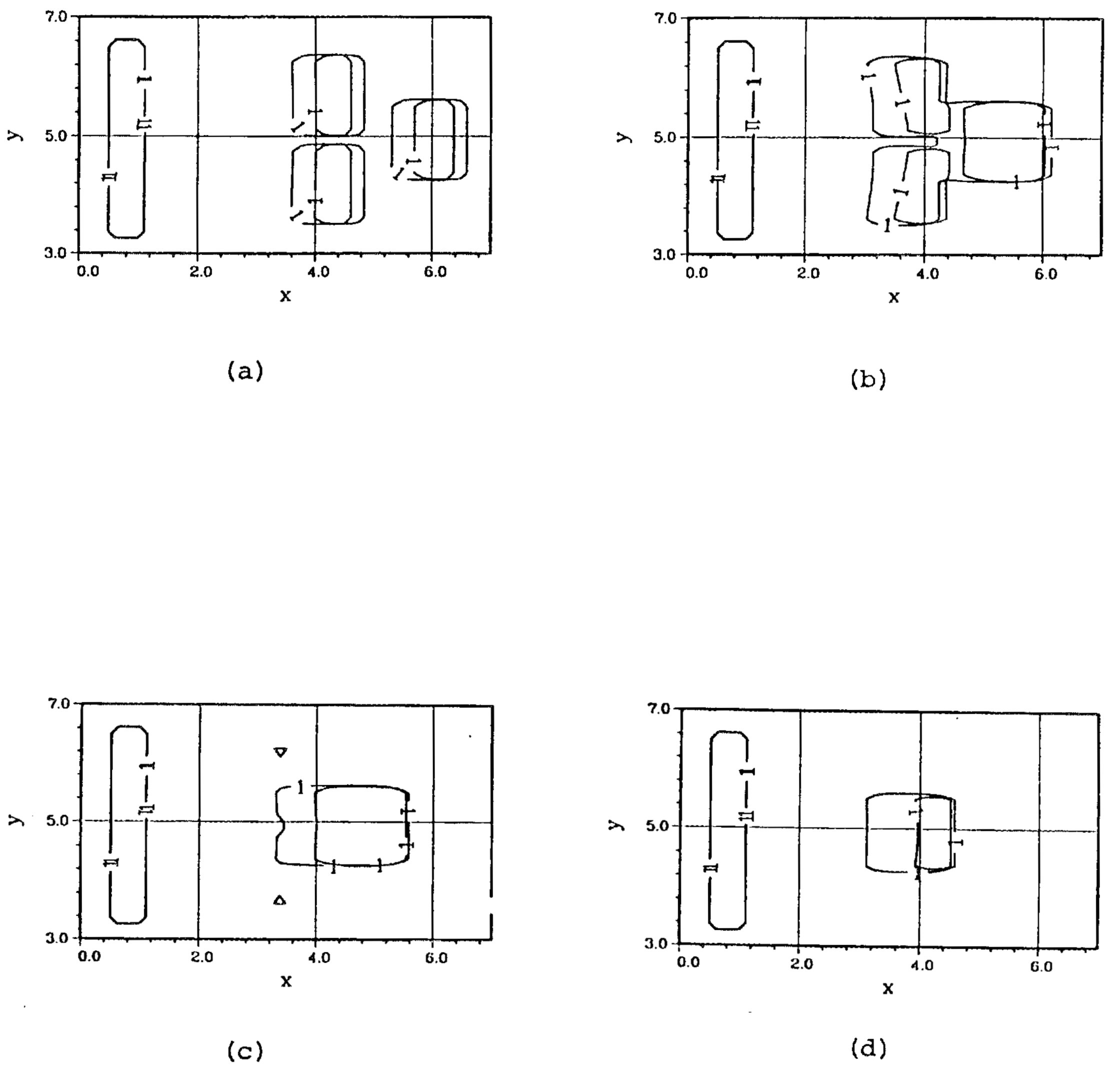



**Fig. 4. Contour map depiction of the spatial distribution of the four weapon types over the battlefield in the Staggered Defense Scenario as calculated by DCOR on a 57 × 33 uniform mesh, after: (a) 1, (b) 4, (c) 6, and (d) 11 minutes into the battle.**

In colored graphics the Blue side appears in cyan, while the red side is yellow. Only one contour line is drawn per force type where the density equals 1 $km^{-2}$; the contour lines for the M60s and the TOW missiles overlap over most of the battle's duration.

Finally, to demonstrate the utility of the DCOR code to the commander planning an attack on a defended site, or planning defense against an impending attack, we perform a study of the sensitivity of the battle outcome with respect to the variables under his control. All sensitivity studies presented below are performed on the $15 \times 9$ mesh with the same attrition rates that have been validated against JANUS.

The first step in strategy development in a situation where one side wins consistently is to find out the source of strength of that side. The commander of the side possessing this strength would then build his strategy on the premise of preserving his edge in that area, and in anticipating and aborting any attempt by his opponent to reduce his advantage. On the other hand, the opposing commander would direct his resources towards eliminating the source of strength of the enemy, and if sacrifice is absolutely necessary to accomplish his objective, try to reduce losses to a minimum. In searching for the source of strength of the Blue side in this battle, in spite of the overwhelming initial Red to Blue Force ratio advantage (2.5:1) one can attempt to increase the initial Red Force so that the ratio is 4:1 and 6:1, for example. The resulting time dependent attrition profiles are shown in Figs. 5 and 6, respectively. From these figures, it is clear that the larger initial Red Force

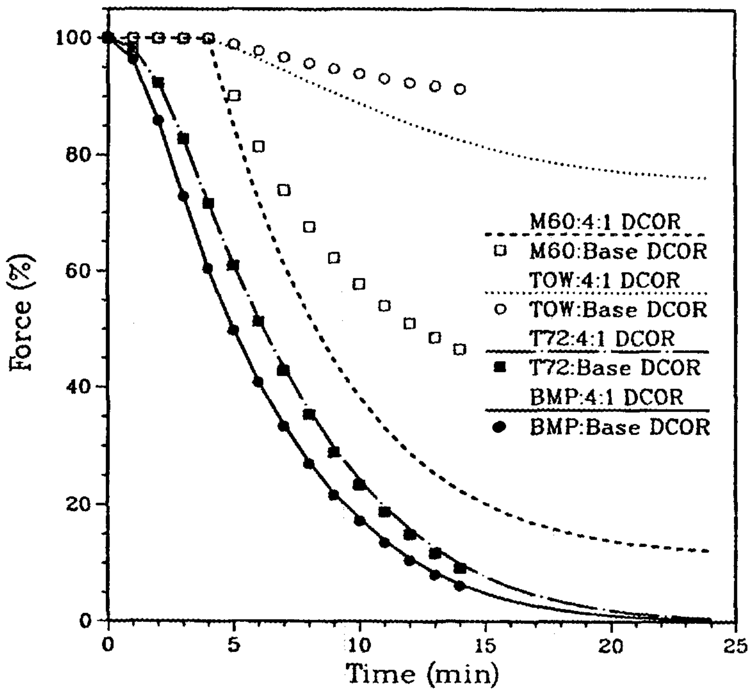



**Fig. 5. Comparison between the reference case (initial Red to Blue ratio of 2.5:1) and an initial ratio 4:1 case, both calculated by DCOR on a $15 \times 9$ uniform mesh.**

advantage lengthens the battle duration, and annihilates the M60s completely in the 6:1 case, but they are insufficient to demolish the TOW missiles, and make them ineffective. Indeed in the 6:1 initial ratio case, the TOW missiles solely hold back the Red attacking forces starting from the fifteenth minute of the battle, and are responsible for demolishing the Red Force at the end of the battle, albeit at great losses to the Blue side.

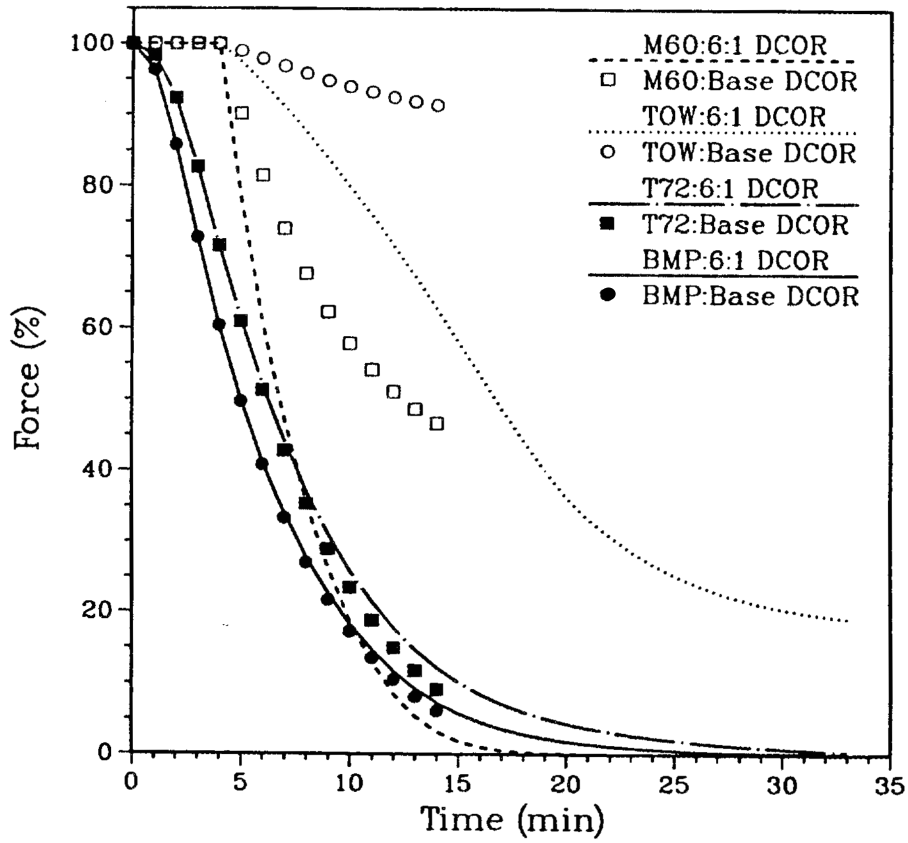



**Fig. 6. Comparison between the reference case (initial Red to Blue ratio of 2.5:1) and an initial ratio 6:1 case, both calculated by DCOR on a 15 × 9 uniform mesh.**

These initial experiments seem to suggest the crucial role the TOW missiles play in guaranteeing Blue's victory. To confirm this conclusion, we replace all of Blue's TOW missiles in the 2.5:1 initial ratio case with M60 tanks and repeat the calculation. The results of this experiment as shown in Fig. 7, clearly indicate the truth of this hypothesis as it shows Blue's defeat, at a great cost to Red.

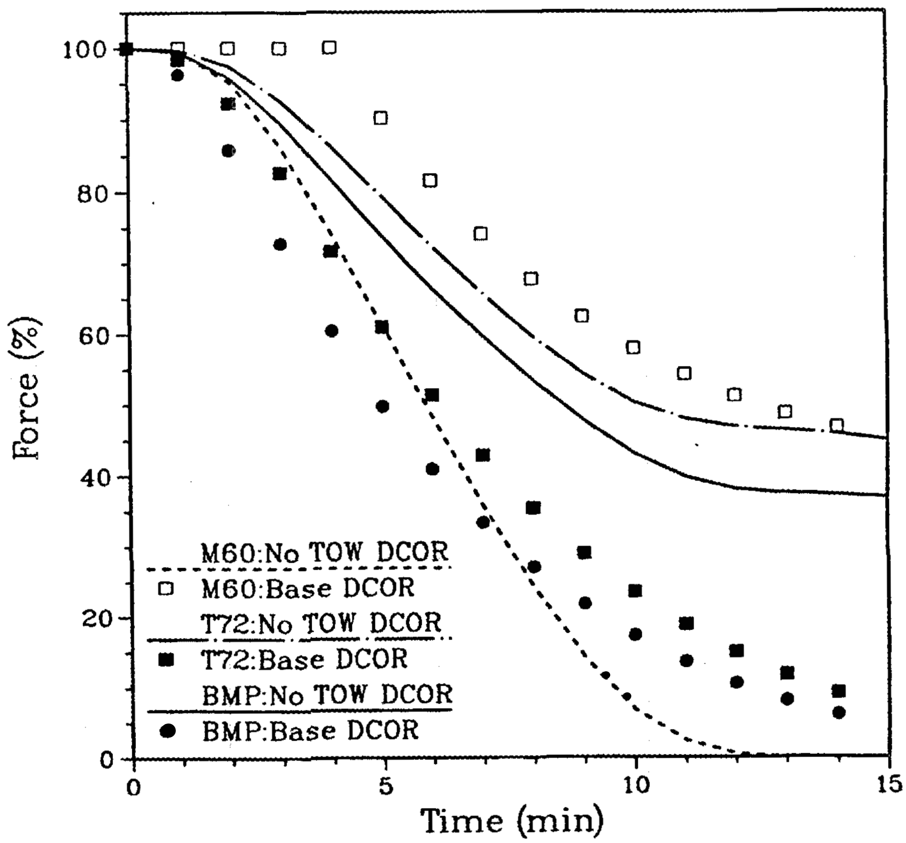



**Fig. 7. Comparison between the reference case (Blue has 36 M60 tanks and 12 TOWs) and a case where the TOWs are replaced by M60s (i.e., Blue has 48 M60s), both calculated by DCOR on a 15 × 9 uniform mesh.**

At this point the commander of the Red side must weigh the importance of achieving his objective, which at most can be to soften but not demolish the Blue defense, against the grave cost of annihilation of his forces. If attacking while Blue possesses the TOW missiles is inevitable, the Red commander should consider alternative, possibly unconventional, initial force deployment schemes in order to maximize his gain, e.g., maximum softening of Blue force. We tried three such schemes: a single formation massed at $x$=4.5, $y$=5; a single skirmish line at $x$=4.5; and uniformly distributed over the rectangle $4.5 \leq x \leq 6.5$, $3.5 \leq y \leq 6.5$. The resulting attrition evolution profiles are shown in Figs. 8, 9, and 10, respectively. It is clear from these figures that the single massed formation is worst, followed by the single line skirmish, then the uniformly distributed initial deployment. However, out of the three experiments, only the last produces better results for Red than the two echelons, two forward and one following battalions, reference case depicted in Fig. 1.

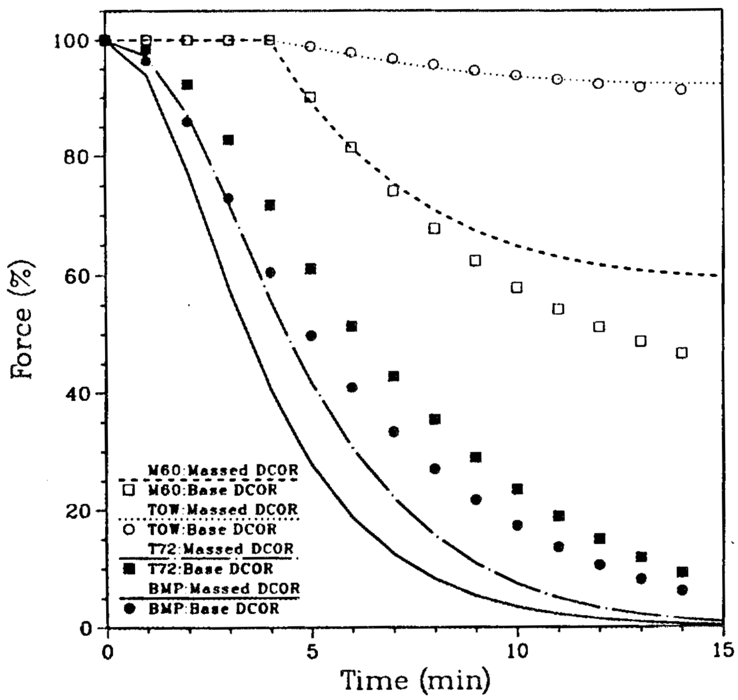



**Fig. 8. Comparison between the reference case initial Red Force deployment, and a case where Red is massed in a single formation at x=4.5, y=5, both calculated by DCOR on a 15 × 9 uniform mesh.**

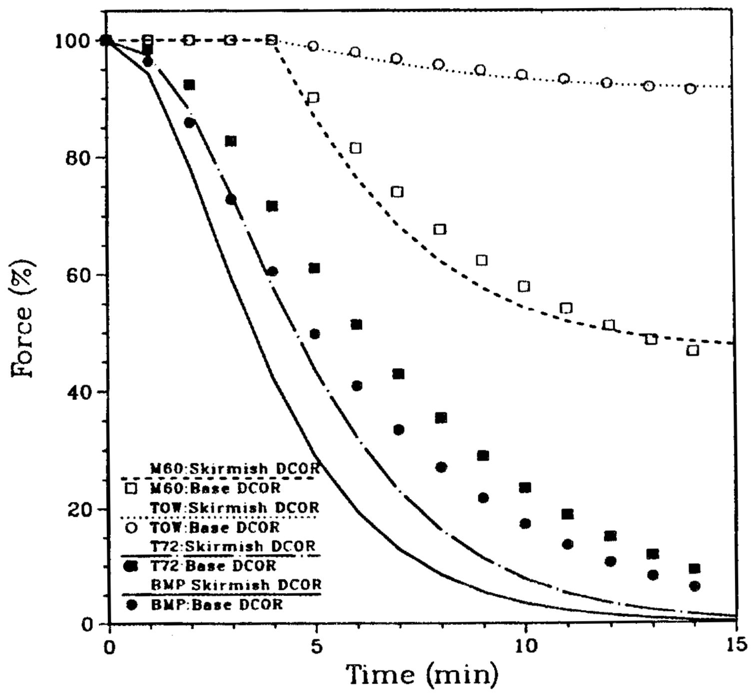



**Fig. 9. Comparison between the reference case initial Red Force deployment, and a case where Red initially forms a single skirmish line at x=4.5, both calculated by DCOR on a 15 × 9 uniform mesh.**

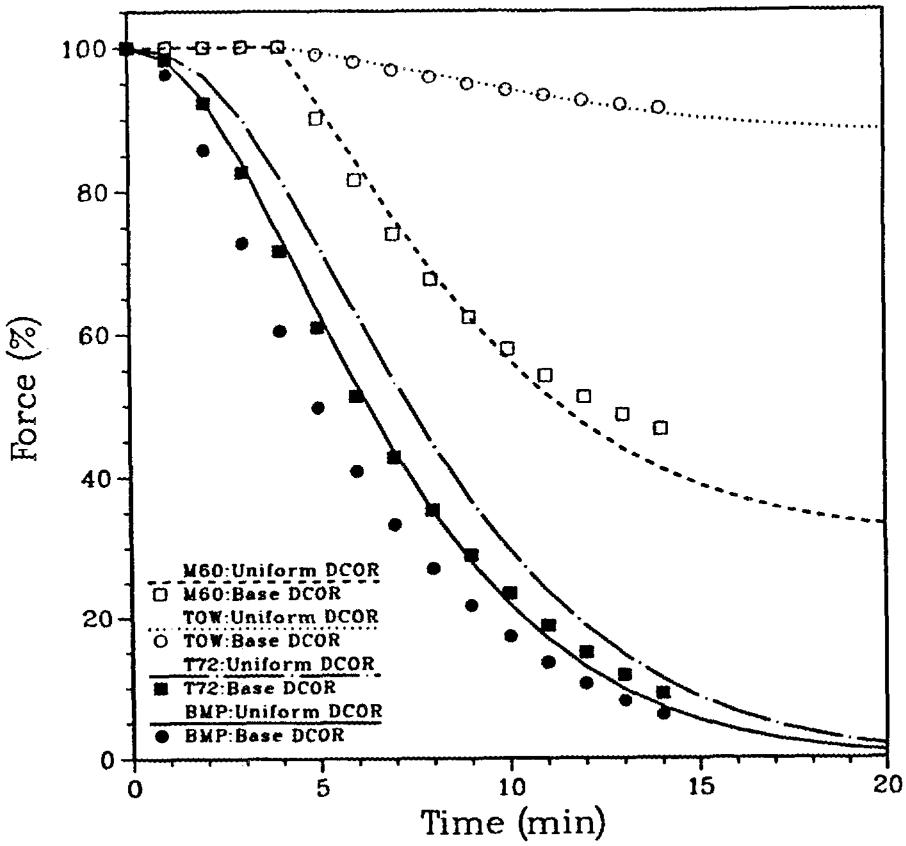



**Fig. 10. Comparison between the reference case initial Red Force deployment, and a case where Red is initially uniformly spread over the rectangle $4.5 \leq x \leq 6.5$, $3.5 \leq y \leq 6.5$, both calculated by DCOR on a $15 \times 9$ uniform mesh.**

Red's last resort to avoid annihilation at Blue's hands would be to find a way to eliminate, or at least diminish the number of TOW missiles before beginning the attack. This can be accomplished by requesting support from his superiors of weapon systems not at his disposal, that are capable of selectively striking the TOWs, e.g., air support, smart weapons,[15] etc. Clearly, it is Blue commander's job to anticipate and neutralize such attempts by Red's commander to preserve his battlefield superiority represented by the TOWs.

The example we used in this section and the arguments we used in developing strategies and counterstrategies based on sensitivity studies conducted via the DCOR code lack realistic military insight, and would probably appear to be naive to the military expert. For example, no commander fights a battle up to complete annihilation of his force. However, the emphasis here has been on flexibility, versatility, and computational robustness of the DCOR code in performing a wide variety of modeling tasks that can provide the commander with invaluable predictions of the "expected outcome" of the battle. Such predictions can complement intuition, expertise, and military doctrine in realistic situations to develop better strategic evaluations of battle outcomes as a function of options and actions available to the commanders of the two sides in the battle.

# 6. CONCLUSIONS

We have developed the first version of a flexible, user-friendly computer code, DCOR (previously known as WAR), implementing the deterministic PDE combat model recently proposed and tested by ORNL. We realize that the DCOR code is still in its infancy and requires substantial development, testing, and validation. However, the results we have accumulated so far, some of which are reported here, and others still emerging at the time of this writing, all seem to suggest the usefulness of this code, and its future progeny, in performing various combat modeling tasks. Thus, it provides a very powerful tool for predicting battle outcome, determining initial deployment of forces, optimizing use of resources to maximize gain from battle, predict the opponent's actions by exploring his options, and developing strategies and counterstrategies to achieve battle objectives. Just like in other areas where neither Monte Carlo nor deterministic methods totally dominate the field, we conjecture that stochastic battle simulations and PDE combat models will coexist in the future, most probably complementing one another.

# REFERENCES


1. For example, J. E. Pimper and L. A. Dobbs, *Janus Algorithms Document Version 4.0*, Lawrence Livermore National Laboratory, Livermore, CA (1988).
2. F. W. Lanchester, "Aircraft in Warfare, The Dawn of the Fourth Arm—No. V., The Principle of Concentration," *Engineering* **98**, 422 (1914).
3. J. G. Taylor, *Lanchester Models of Warfare*, Vols. 1 and 2, MAS, Operation Research Society of America, Virginia (1983).
4. D. Willard, "Lanchester Attrition of Interpreting Forces," *Naval Research Logistics* **37**, 31 (1990).
5. V. Protopopescu, R. T. Santoro, J. Dockery, R. L. Cox, and J. M. Barnes, "Combat Modeling with Partial Differential Equations," ORNL/TM-10636, Oak Ridge National Laboratory, November 1987.
6. V. Protopopescu, R. T. Santoro and J. Dockery, "Combat Modeling with Partial Differential Equations," *Eur. J. Oper. Res.* **38**, 178 (1989).
7. P. Rusu, "Two-Dimensional Combat Modeling with Partial Differential Equations," ORNL/TM-10973, Oak Ridge National Laboratory, December 1988.
8. R. T. Santoro, P. Rusu and J. M. Barnes, "Mathematical Descriptions of Offensive Combat Maneuvers," ORNL/TM-11000, Oak Ridge National Laboratory, January 1989.
9. V. Protopopescu, R. T. Santoro, R. Cox, P. Rusu and J. Dockery "Combat Modeling with Partial Differential Equations: The Bidimensional Case," ORNL/TM-11343 (1989).
10. J. M. Hyman, "The Method of Lines Solution of Partial Differential Equations," Courant Institute of Mathematical Sciences, Mathematics and Computing Laboratory Report COO-3077-139 (1976).
11. A. C. Hindmarsh, "GEARB: Solution of Ordinary Differential Equations Having Banded Jacobian," Lawrence Livermore Laboratory Report UCID-30059, Rev. 1 (1975).
12. V. Protopopescu, R. T. Santoro and Y. Y. Azmy, "Recent Advances in Analytic Combat Simulation: From Modeling to Validation and Beyond," accepted for publication in the *Proc. of MORS.*
13. R. T. Santoro, Y. Y. Azmy and V. Protopopescu, "Oak Ridge Cross Validates Modeling Versus War Gaming," *Signal* **44**, p. 49, July 1990.
14. G. A. Keller and A. P. Deverill, "Performance Assessment of Combat Forces After Irradiation," DNA 89-68, Defense Nuclear Agency, February 1989.
15. V. Protopopescu, R. T. Santoro, Y. Y. Azmy and B. L. Kirk, "Recent Advances in Validating Analytic Combat Models," accepted for publication in the *Proc. of the MORS Minisimposium on Simulation–Validation*, Albuquerque, NM, October 15–18, 1990.

## INTERNAL DISTRIBUTION

1. B. R. Appleton
2–6. Y. Y. Azmy
7. D. E. Bartine
8. T. J. Burns
9. R. M. Davis
10. W. Fulkerson
11. C. M. Haaland
12. D. S. Hartley III
13. D. T. Ingersoll
14. B. L. Kirk
15. A. S. Loebl
16. F. C. Maienschein
17. J. R. Merriman
18. V. Protopopescu
19. R. T. Santoro
20. A. Zucker
21. EPMD Reports Office
22–23. Laboratory Records Department
24. Laboratory Records, ORNL–RC
25. Document Reference Section
26. Central Research Library
27. ORNL Patent Section

## EXTERNAL DISTRIBUTION

28. Robert Alongi Sr. , U.S. Army Missile Command , ATTN: AMSMI-RD-AC, Redstone Arsenal, Alabama 35898-5282
29. Lois Aymett, U.S. Army Missile Command, ATTN: AMSMI-RD-AC, Redstone Arsenal, Alabama 35898-5282
30. David Baker, U.S. Army Material Systems Analysis Activity, UK RARDE, Aberdeen Proving Grounds, Maryland 21005
31. J. J. Dorning, Department of Nuclear Engineering and Physics, Thornton Hall, McCormick Road, University of Virginia, Charlottesville, Virginia 22901
32. Bruce W. Fowler, U.S. Army Missile Command, ATTN: AMSMI-RD-AC, Redstone Arsenal, Alabama 35898-5282
33. R. M. Haralick, Department of Electrical Engineering, University of Washington, Seattle, Washington 98195
34. R. L. Helmbold, Concepts Analysis Agency, 8120 Woodmont Avenue, Bethesda, Maryland 20814
35. P. B. Hemmig, Safety and Physics Branch, Office of Technology Support Programs, Department of Energy, Washington, D.C. 20545
36. Joe Lacetera, U.S. Army Laboratory Command, ATTN: AMCLD-PA, 2800 Powder Mill Road, Adelphi, Maryland 20783-1145
37. James E. Leiss, 13013 Chestnut Oak Drive, Gaithersburg, Maryland 20878
38. Neville Moray, Department of Mechanical and Industrial Engineering, University of Illinois, 1206 West Green Street, Urbana, Illinois 61801
39. Jerry Smith, U.S. Army Missile Command, ATTN: AMSMI-RD-AC, Redstone Arsenal, Alabama 35898-5282
40. E. B. Vandiver III, Director, Concepts Analysis Agency, 8120 Woodmont Avenue, Bethesda, Maryland 20814
41. Sally Van Nostrand, U.S. Army Laboratory Command, 2800 Powder Mill Road, Adelphi, Maryland 20783-1145
42. Joseph Wald, U.S. Army Ballistics Research Laboratory, Aberdeen Proving Grounds, Maryland 21005

43. Mary F. Wheeler, Department of Mathematical Sciences, Rice University, P.O. Box 1892, Houston, Texas 77251
44. Office of Assistant Manager for Energy Research and Development, Department of Energy, Oak Ridge Operations, P.O. Box 2001, Oak Ridge, Tennessee 37831

45–54. Office of Scientific and Technical Information, P.O. Box 62, Oak Ridge, Tennessee 37830