NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

GENERALIZED WEAPON EFFECTIVENESS MODELING

by

Colin Michael Anderson

June 2004

Thesis Advisor: Morris R. Driels

Approved for public release; distribution is unlimited

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE	Form Approved OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	June 2004	Master's Thesis

4. TITLE AND SUBTITLE: Generalize Weapon Effectiveness Modeling	5. FUNDING NUMBERS
6. AUTHOR(S) Colin Michael Anderson	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A	10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT	12b. DISTRIBUTION CODE
Approved for public release; distribution is unlimited	

13. ABSTRACT (maximum 200 words)

In this thesis, we compare weapon effectiveness methods to determine if current effectiveness models provide accurate results. The United States Military currently adheres to a compilation of data and methodologies named the Joint Munitions Effectiveness Manuals (JMEM) to determine the effectiveness of air delivered weapons against a variety of ground targets. Since the time these manuals were implemented in the 1960s, progress in technology has allowed the weapon/target interaction to be more accurately modeled. This thesis investigates the differences of these high fidelity models for unguided weapons and the JMEM computations in order to determine whether the older, more simplistic, models need to be upgraded.

14. SUBJECT TERMS	15. NUMBER OF PAGES
Weaponeering, Weapon Effectiveness, Monte Carlo Simulations	129
	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	UL

NSN 7540-01-280-5500 Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited

GENERALIZED WEAPON EFFECTIVENESS MODELING

Colin M. Anderson
Ensign, United States Navy
B.S., University of Washington, 2003

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN MECHANICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
June 2004

Author: Colin M. Anderson

Approved by: Morris R. Driels

Anthony J. Healey
Chairman, Department of Mechanical and Astronautical Engineering

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

In this thesis, we compare weapon effectiveness methods to determine if current effectiveness models provide accurate results. The United States Military currently adheres to a compilation of data and methodologies named the Joint Munitions Effectiveness Manuals (JMEM) to determine the effectiveness of air delivered weapons against a variety of ground targets. Since the time these manuals were implemented in the 1960s, progress in technology has allowed the weapon/target interaction to be more accurately modeled. This thesis investigates the differences of these high fidelity models for unguided weapons and the JMEM computations in order to determine whether the older, more simplistic, models need to be upgraded.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

I would like to thank Professor Driels for his patience and guidance throughout my time at the Naval Postgraduate School. I have learned a great deal, and am glad I had the chance to work with you.

I would also like to give thanks for the support and love from my parents, Bill and Janalyn, my RMS family, and the E-Funk Ensigns. I couldn't have asked for better friends and family.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

The objective of this thesis is to compare different methodologies for determining unguided weapon effectiveness against targets in order to determine if current methods used by the United States Military provide reasonable estimates of damage. The current methods used by the military in determining weapon damage effectiveness have been used for quite some time and may be outdated. Over the last 40 years, weapon/target interaction data from weapon tests have been compiled by the military into a volume of data called the Joint Munitions Effectiveness Manuals, or JMEM. The JMEM has used this compiled data to formulate estimates of weapon effectiveness for given weapons and targets. In order to simplify weapon effectiveness calculations to meet computing speeds of the time, the JMEM needed to implement approximations of weapon effectiveness techniques. For this thesis we will compare these JMEM methods with weapon effectiveness modeling methods that do not implement approximations and that more closely represent the weapon/target interaction.

We will use Monte Carlo simulations as the basis of our comparison testing. The first Monte Carlo simulation will use a damage function called the Carleton damage function, which is based on weapon test data, to represent the damage inflicted by a weapon of given lethality. In the Monte Carlo simulations, we represent weapons with a lethal area matrix center on the weapon impact point. This lethal area matrix will be populated with damage values obtained from either the Carleton damage function, or an approximation of the Carleton damage function, depending on the method tested. We assume this Monte Carlo weapon effectiveness model representing the weapon with a lethal area matrix populated with the Carleton damage function to be the highest fidelity method in determining weapon effectiveness. This is because some JMEM methods generally rely on approximations of the Carleton damage function to calculate weapon probabilities of damage.

In order to obtain a broad array of data, we will compare the JMEM and Monte Carlo simulation results for single as well as sticks of weapons against unitary and area targets. Along with different weapon/target scenarios, we also test the different methods for various delivery accuracies to try to pinpoint the reason for discrepancies in the methods' effectiveness results. By analyzing the differences between weapon effectiveness results, we hope to determine whether the current JMEM techniques should be replaced by more complex modeling methods.

II. BACKGROUND

A. DELIVERY ACCURACY

The probability of damaging or killing a target obviously depends on how close a weapon is delivered to the target. This delivery accuracy is affected by many components, especially errors in aiming the weapon and the ballistics of the weapon used.

1. Aiming Errors

The delivery accuracy of a weapon depends heavily on the ability to aim that weapon at a desired impact point. Statistical models are used to determine to what degree these aiming errors affect the probability of impacting a desired point. First, two directions are defined in the ground plane. The range direction is the direction of the velocity vector of the aircraft releasing the weapon. The deflection direction is the direction perpendicular to the range. We next look at a sample number of weapons released from the same aircraft at the same release conditions, all aimed at the same point on the ground plane. This will result in a map of impact points on the ground plane shown in Figure 2.1.

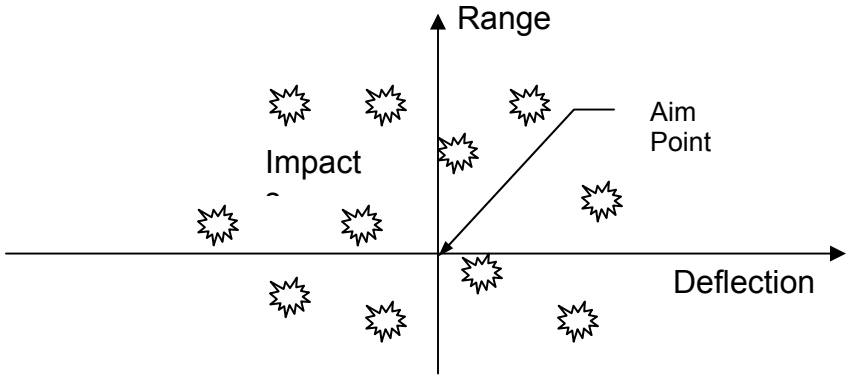

Figure 2.1: Range and deflection directions with impact points.

The distribution, or dispersion, of these impact points can be defined statistically. In order to do this, we assume that the distribution of the impact points in both range and deflection is Gaussian and independent of each other.

A Gaussian, or normal distribution, is a distribution of data that has characteristics that match a probability density function (PDF) given in equation (2.1).

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp[\frac{-(x-\mu)^2}{2\sigma^2}] \tag{2.1}$$

The mean (μ) and the variance (σ^2) are calculated from a given data set, or in our case, the weapon impact data. The mean for a normal distribution is the value of x that gives the maximum value of f(x). This is illustrated in Figure 2.2.

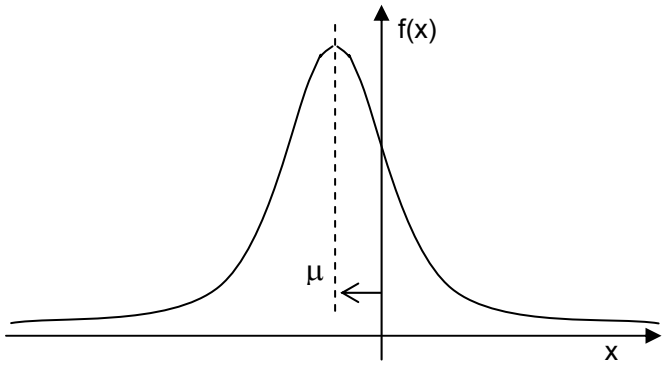

Figure 2.2: Gaussian distribution probability density function (From Ref. [1])

The variance gives an indication of the spread of the data set. A small variance, for example, would give a PDF that has a sharp peak, whereas a large variance would produce a flatter PDF. The square root of the variance (σ), or standard deviation, of the sample set is also an important quantity. In a normal distribution, approximately 68% of the data will lie in a range $\pm\sigma$. Similarly, 95.5% of data will lie in the range $\pm 2\sigma$, and 99.7% will lie within $\pm 3\sigma$.

The normal distribution assumption can be directly applied to the two dimensional impact point data. For the two dimensional case, we look at the two dimensional, bi-variate probability density function shown in equation (2.2).

$$f(x,y) = \frac{1}{2\pi\sigma_x\sigma_y}\exp-[\frac{(x-\mu_x)^2}{2\sigma_x^{\ 2}} + \frac{(y-\mu_y)^2}{2\sigma_y^{\ 2}}] \qquad (2.2)$$

The subscripts x and y denote the direction of the specified mean and standard deviation. For the weaponeering problem, the range and deflection directions will be treated as x and y respectively. A graphical idea of this concept is presented in Figure 2.3.

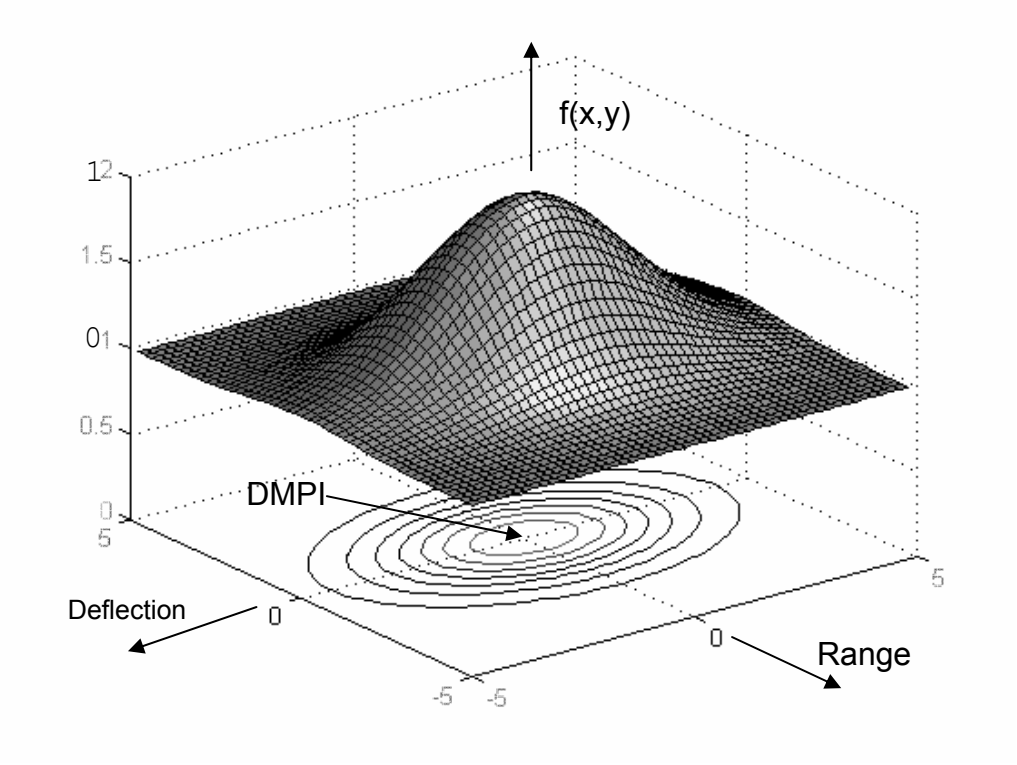

Figure 2.3: Bi-variate Gaussian distribution (From Ref. [1])

Once the impact point distribution has been assumed to be normal, we will describe the distribution in terms of a range error probable (REP) and deflection

error probable (DEP). The REP and DEP are both related to the standard deviation of the distribution in the range and deflection directions. Where the standard deviation was defined as the range such that 68% of points will lie within $\pm\sigma$, the REP is defined as the distance in the range direction such that 50% of the total impact points will lie in the range $\pm$REP. The DEP is defined similarly, but in the deflection direction. These terms are better shown in Figure 2.4.

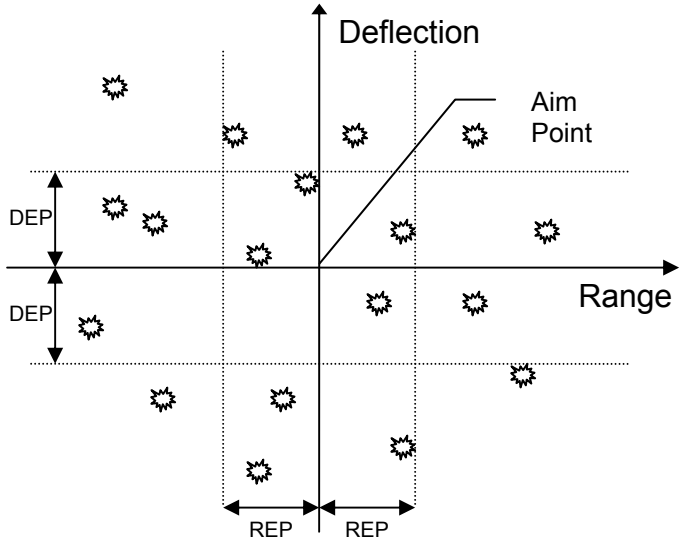

Figure 2.4: Definition of REP and DEP in ground plane.

The difference between REP and DEP and the standard deviations is the percentage of impact points that are contained within those ranges. Using tabulated values of the PDF for a normal distribution we find equations (2.3) and (2.4).

$$REP = 0.6745\sigma_x \tag{2.3}$$

$$DEP = 0.6745\sigma_y \tag{2.4}$$

Another common way of describing the dispersion of impact points involves using the circular error probable (CEP). The CEP is a radius about the desired aim point that contains 50% of the impact points, as shown in Figure 2.5.

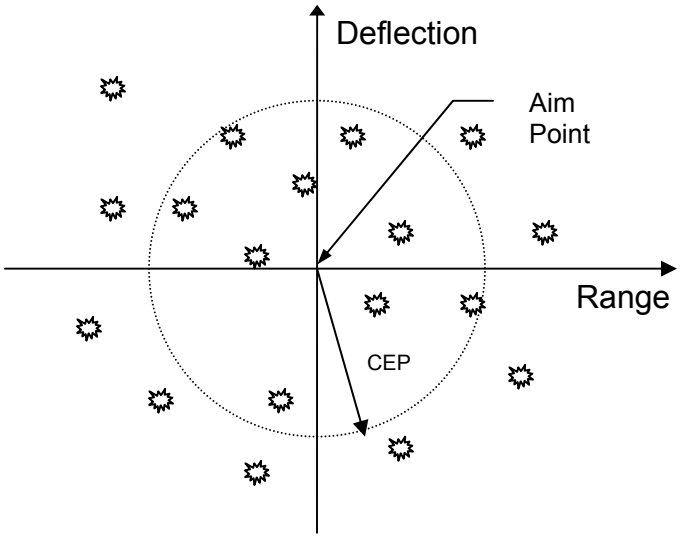

Figure 2.5: Definition of CEP in ground plane.

2. Ballistic Dispersion

The delivery accuracy of a weapon is also dependent on the manufacturing of the munitions used to attack a target. Many details may vary from one weapon to another. An example of these errors may be seen from a rifle bolted to the ground firing at a target. Since the rifle is bolted, we can assume that there are no aiming errors. The rounds fired from the rifle would create a pattern of impact points about some point on the target. These errors can be due to anything from variations in bullet shape to different weights of explosives used inside each bullet. [Ref. 1]

Usually ballistic dispersion errors are defined as a standard deviation in the normal plane of the weapon, measured in mils (milliradians). The standard deviation in the normal plane is denoted as σ_b. Figure 2.6 illustrates a typical

weapon release against a target. The normal plane can be seen as the plane perpendicular to that of the velocity vector of the weapon at its impact point.

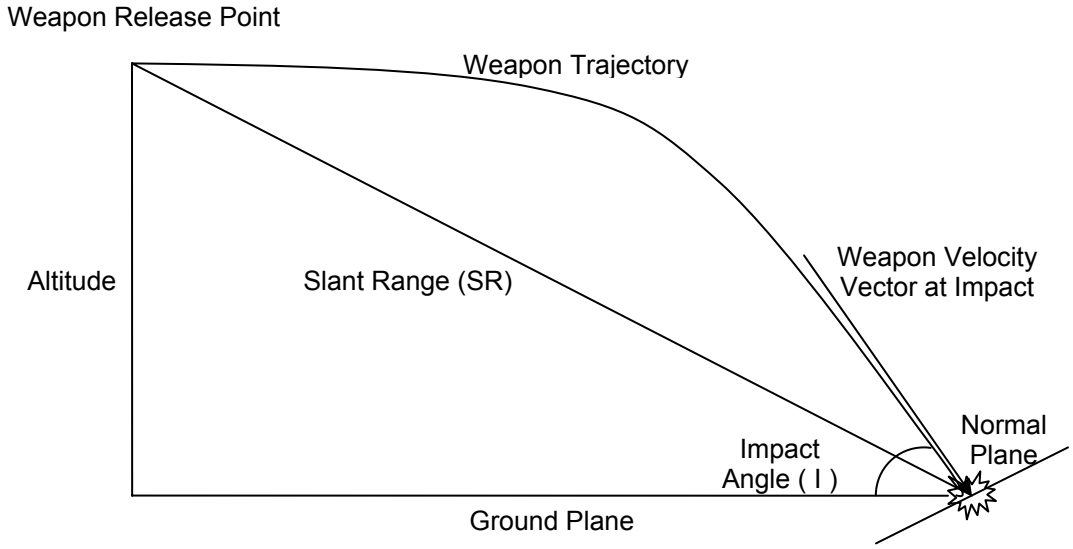

Figure 2.6: Generalized Weapon Trajectory and Impact Diagram

When aiming errors and ballistic dispersion errors are both present in a problem, it is often convenient to write the ballistic dispersion in terms of feet in the ground plane. Calculating x_{br} (ballistic dispersion error in the range direction in feet) and x_{bd} (ballistic dispersion error in deflection direction in feet) are found with in equations (2.5) and (2.6).

$$x_{br} = \frac{0.6745 \times SR \times \sigma_b}{1000 \sin(I)} \tag{2.5}$$

$$x_{bd} = \frac{0.6745 \times SR \times \sigma_b}{1000} \tag{2.6}$$

The slant range (SR) is the straight-line distance from the aircraft at the point of release to the impact point of the weapon as shown in Figure 2.6. The impact

angle I is measured in radians, and is the angle between the weapon velocity vector at impact and the ground plane.

For individually released weapons, we are able to obtain an equivalent REP and DEP that combines the ballistic dispersion errors in feet with the aiming errors in feet. To find the equivalent REP and DEP, the ballistic dispersion error and aiming errors are root sum squared, and defined as REP' and DEP',

$$REP' = \sqrt{REP^2 + x_{br}{}^2} \tag{2.7}$$

$$DEP' = \sqrt{DEP^2 + x_{bd}{}^2} \tag{2.8}$$

Ballistic dispersion is treated differently when we deal with multiple weapon releases, or sticks of weapons and will be discussed later in this thesis.

B. LETHAL AREAS

There are two major mechanisms in conventional weapons that inflict damage upon a target. These mechanisms are the blast wave produced by the high explosive inside of a weapon, and the fragments propelled from the weapon after it has exploded.

When the high explosive inside a conventional warhead detonates, the explosive material will almost instantly be converted into a gas at very high temperature and pressure. These values are typically near 200 atmospheres of pressure and 5000 degrees Celsius [Ref. 1]. This high pressure will fragment the case around the weapon and compress the air around the weapon. This compression forms a blast wave that travels into the surrounding area. Blast damage is highly effective against most targets that lie near the impact/explosion point of the weapon. Since the blast wave is a wave of pressurized and superheated air, it dissipates fairly quickly in the atmosphere, which means its

effectiveness greatly decreases with distance from the point of impact. The blast energy is roughly inversely proportional to the cube of the distance from the weapon impact.

Once a blast wave has dissipated in the atmosphere, fragmentation becomes the main source of damage. Primary fragments are the remains of the weapon after the high explosive charge has shattered the weapon casing. These fragments are normally fairly small and are propelled up to very high velocities from the initial explosion of the weapon. These velocities propel the fragments through, and eventually past the front of the pressure shock wave created by the high explosive blast. Since the fragments are only slowed by air friction, they travel further than the blast wave. This is the advantage of fragmenting weapons. Since fragments can damage targets further from the weapon impact than blast can, we can accept greater weapon miss distances, and still damage targets. Depending on the weapon, a hazardous fragment is one having impact energy of 58 ft-lb (79 joules) or greater [Ref. 1].

The damage resulting from fragmentation and blast is dependent on the size and type of weapon, as well as the kill criteria for the target it is used against. Undergoing a fairly lengthy analysis described in detail in Reference 1, weaponeers are able to create a damage matrix that gives the probability of killing (P_K) a target a given distance away from the weapon impact. An example of a damage matrix is given in Table 2.1. Each table cell shows the probability of kill given a distance in both range and deflection from the weapon impact point. Figure 2.7 shows the angles and distances associated with a weapon detonation near a target. The distance the weapon detonates from the target, which is denoted x and y in Figure 2.7, are the range and deflection distances given in Table 2.1. Note Table 2.1 shows only positive deflection distances. This is done because fragments are symmetrical about the range direction.

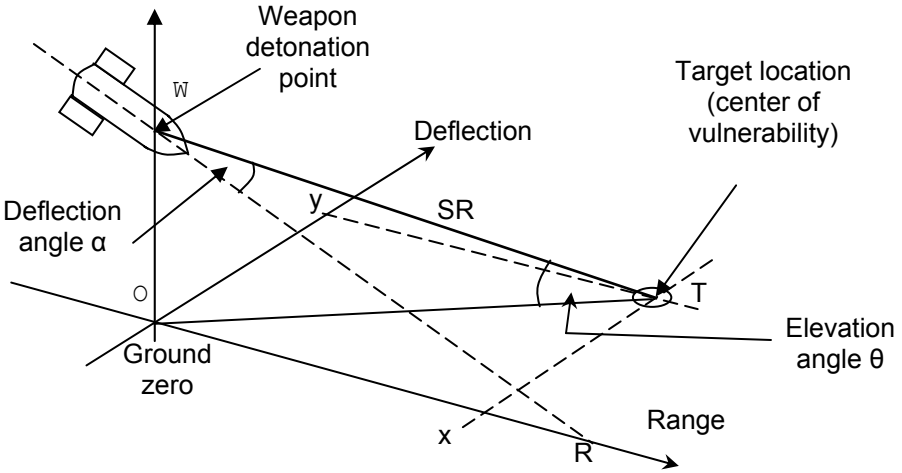

Figure 2.7: Weapon/target interaction geometry.

Deflection (ft)										
Range▼	37.9	75.8	113.7	151.6	189.5	227.4	265.3	303.2	341.1	379.0
-114.4	0	0	0	0	0	0	0	0	0	0
-100.1	0	0	0	0	0	0	0	0	0	0
-85.8	0	0	0	0	0	0	0	0	.0001	.0001
-71.5	.0001	0	0	0	0	0	.0001	.0002	.0001	.0001
-57.2	.0011	0	0	0	0	.0003	.0004	.0002	.0001	.0001
-42.9	.0028	0	0	0	.0009	.0008	.0004	.0002	.0001	.0001
-28.6	.0064	.0001	.0006	.0029	.0017	.0009	.0005	.0002	.0001	.0001
-14.3	.1402	.0059	.0099	.0042	.0019	.0009	.0005	.0002	.0001	.0001
0	.5571	.0459	.0127	.0045	.0019	.0009	.0005	.0002	.0001	.0001
14.3	.6794	.0891	.0156	.0045	.0019	.0009	.0005	.0002	.0001	.0001
28.6	.1741	.0927	.0325	.0116	.0041	.0012	.0005	.0002	.0001	.0001
42.9	.0060	.0186	.0258	.0128	.0063	.0034	.0016	.0006	.0002	.0001
57.2	.0007	.0050	.0105	.0118	.0061	.0032	.0017	.0010	.0006	.0003
71.5	0	.0024	.0015	.0072	.0056	.0031	.0017	.0010	.0006	.0004
85.8	0	.0010	.0012	.0011	.0045	.0028	.0017	.0009	.0005	.0003
100.1	0	.0003	.0009	.0005	.0012	.0025	.0015	.0009	.0005	.0003
114.4	0	0	.0006	.0004	.0002	.0011	.0014	.0009	.0005	.0003
128.7	0	0	.0003	.0003	.0002	.0001	.0009	.0007	.0004	.0003
143.0	0	0	.0001	.0003	.0001	.0001	.0001	.0006	.0004	.0003
157.3	0	0	0	.0002	.0001	.0001	0	.0002	.0004	.0002
171.6	0	0	0	.0001	.0001	.0001	0	0	.0002	.0002
185.9	0	0	0	.0001	.0001	.0001	0	0	0	.0001
200.2	0	0	0	0	.0001	0	0	0	0	0
214.5	0	0	0	0	.0001	0	0	0	0	0
228.8	0	0	0	0	0	0	0	0	0	0
243.1	0	0	0	0	0	0	0	0	0	0

Table 2.1: Sample damage matrix. (From Ref. [1])

This high fidelity modeling of the weapon target interaction is too difficult to use in more simplistic weaponeering modeling, so approximations are made. The first of these approximations is to find constant values of P_K in the damage matrix, and draw lines of contour through them. This results in a plot similar to Figure 2.8. The contours in the range and deflection directions have near Gaussian distributions, which is also shown in Figure 2.8.

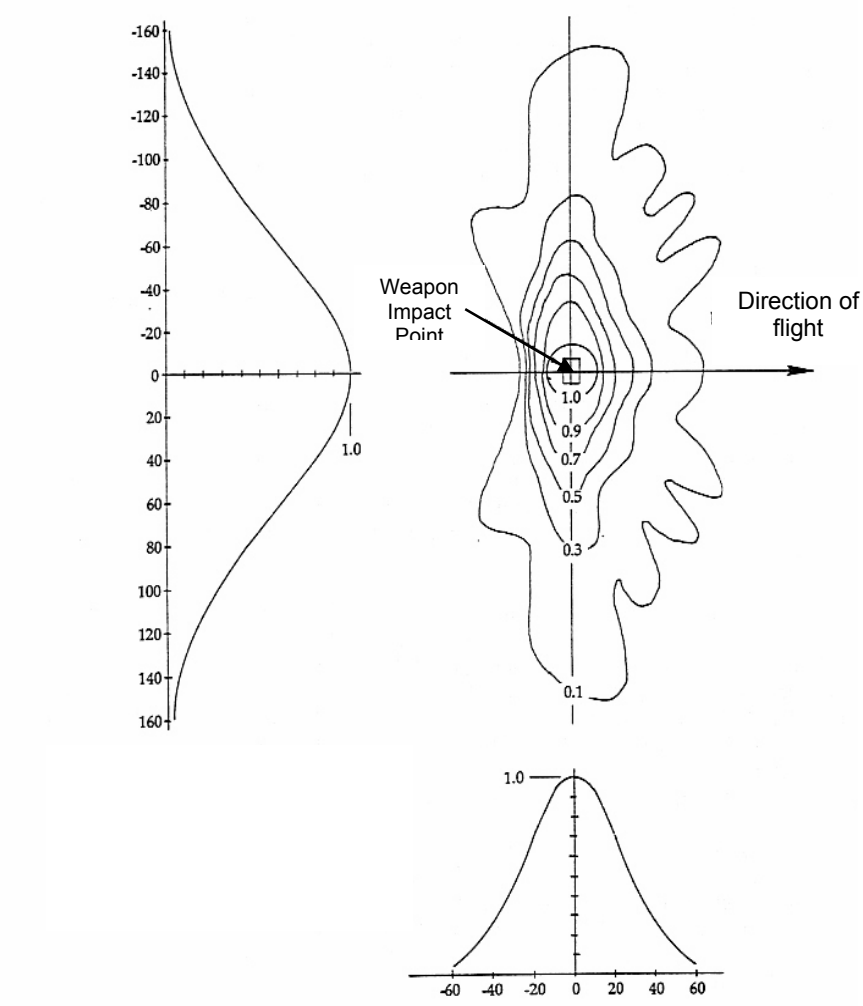

Figure 2.8: Gaussian approximations of P_K contour lines. (From Ref. [1])

The Gaussian approximation allows us to develop an expression that gives the P_K for a point (x,y), given a certain weapon. This is called the Carleton damage function which is given in equation (2.9).

$$P_K(x,y) = \exp -\left[\frac{x^2}{WR_r^{\,2}} + \frac{y^2}{WR_d^{\,2}}\right] \tag{2.9}$$

The Carleton damage function is plotted in Figure 2.9.

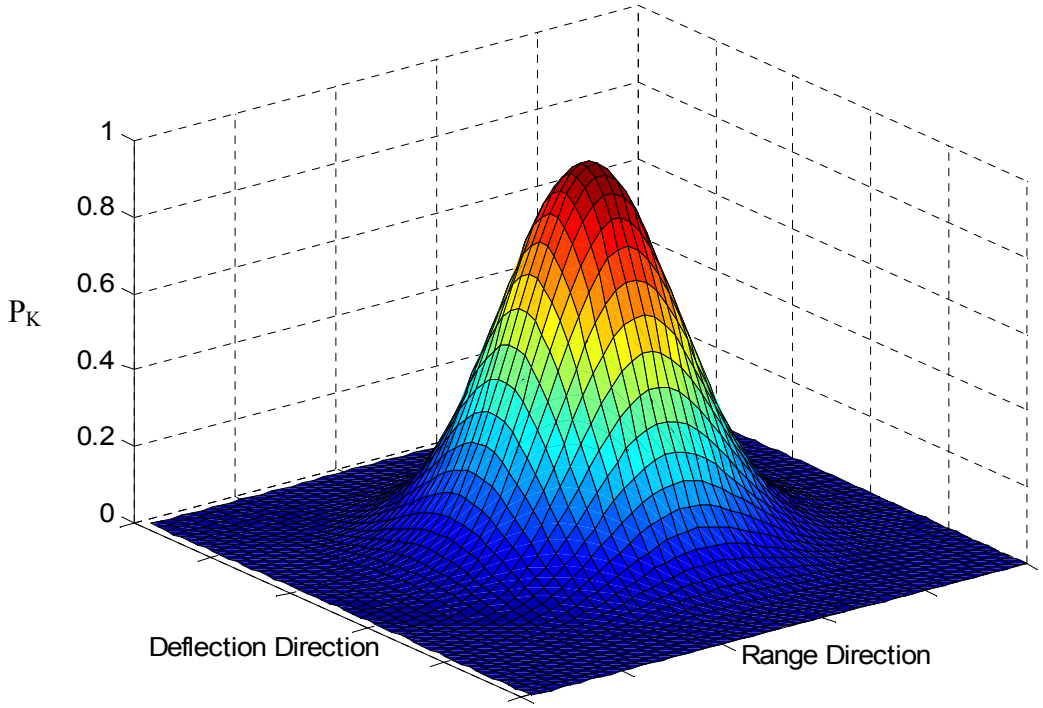

Figure 2.9: Plot of Carleton damage function.

The WR_r and WR_d, or weapon radii in the range and deflection direction are characteristics of the weapon used and help give information on the overall lethality of the weapon. This lethality is best described in a term called the lethal area, or A_L. A_L is the area under the P_K curve for all area in the ground plane.

For fragmentation weapons, we call A_L the mean area of effectiveness due to fragments, or MAE_F. For fragmentation weapons modeled with the Carleton damage function, the MAE_F is equal to the area under the Carleton damage function and given in equation (2.10).

$$MAE_F = \pi \times WR_r \times WR_d \tag{2.10}$$

Even with the numerous assumptions made up to this point, the Carleton damage function is still too difficult to use in simple modeling. We can approximate the lethal area of the Carleton damage function as the area of an ellipse with dimensions WR_r and WR_d as shown in Figure 2.10.

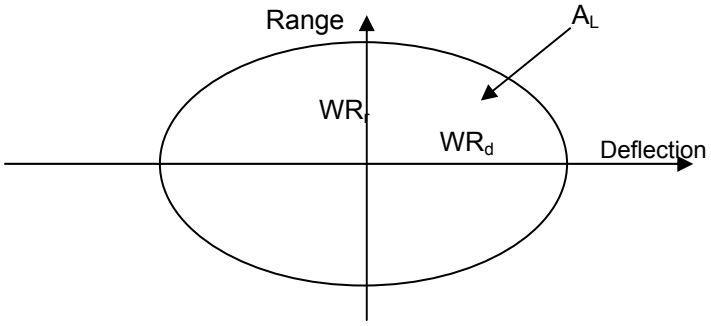

Figure 2.10: Elliptical approximation of Carleton damage function (From Ref. [1])

The P_K inside of this ellipse is equal to unity, and zero outside of the ellipse. This is done to conserve the lethality during the approximation. Since the P_K is unity inside the ellipse, and zero outside, we refer to this as an elliptical “cookie cutter”. The cookie cutter nickname pertains to the fact that a target that lies outside of the ellipse’s boundary will not be damaged, so the ellipse effectively cuts out an area of certain kill. It is also useful to define the ratio of the weapon radii WR_r and WR_d . This ratio (a) is dependent upon the impact angle of the weapon, and through the use of empirical data has been defined in equation (2.11).

$$a = \frac{WR_r}{WR_d} = MAX\left[1 - 0.8\cos(I), 0.3\right] \tag{2.11}$$

Knowing this ratio of the weapon radii allows us to rewrite the Carleton damage function in a different form. We first define an effective target length (L'_{ET}) and width (W'_{ET}) rather than the weapon radii. Equations (2.12) and (2.13) define these effective lengths

$$L'_{ET} = 2 \times WR_r = 1.128\sqrt{MAE_F \times \frac{a}{\pi}} \tag{2.12}$$

$$W'_{ET} = 2 \times WR_d = \frac{L'_{ET}}{a} \tag{2.13}$$

We can then express the Carleton damage function in terms of L'_{ET} and W'_{ET}, as shown in equation (2.14).

$$P_K(x, y) = \exp - \left[\frac{4x^2}{L'^2_{ET}} + \frac{4y^2}{W'^2_{ET}} \right] \tag{2.14}$$

The final step of the procedure is approximating the ellipse shown in Figure 2.10 into a rectangular cookie cutter. Since lethality must be conserved through this approximation, we must calculate rectangular dimensions that enclose the same amount of lethal area as the original ellipse. This approximation is shown graphically in Figure 2.11.

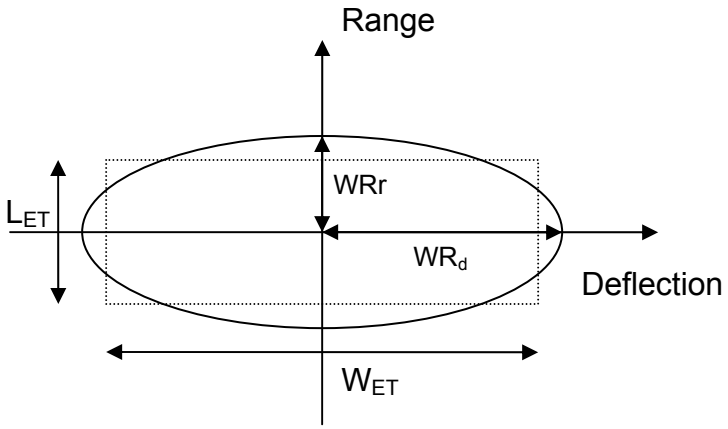

Figure 2.11: Definition of rectangular cookie cutter dimensions (After Ref. [1])

We define the rectangle's dimensions as the effective target length and width, or L_{ET} and W_{ET} respectively. These are calculated from our knowledge of the weapon radii ratio as shown in equations (2.15) and (2.16).

$$L_{ET} = \sqrt{MAE_F \times a} \tag{2.15}$$

$$W_{ET} = L_{ET} / a \tag{2.16}$$

C. EFFECTIVENESS

Suppose we are dealing with a single aircraft releasing a single bomb against a target on the ground. In a problem like this, we would be interested in calculating the single sortie probability of damage, or SSPD. The SSPD is the probability that the single released weapon will cause damage to the target, equal to the amount represented by the damage function. A sortie is single pass by a single aircraft releasing weapons upon a target. For our case, we will deal with a single released weapon.

We will first only consider the range direction in our analysis. The accuracy distribution relative to the target is assumed to be a Gaussian normal distribution, and is called g(x). Also, the damage function of the weapon is modeled with the Carleton damage function, or c(x). To find the SSPD in the range direction, the distance the weapon lands from the target needs to be known. In this example we will call this distance u. Figure 2.12 shows a graphical representation of these assumptions.

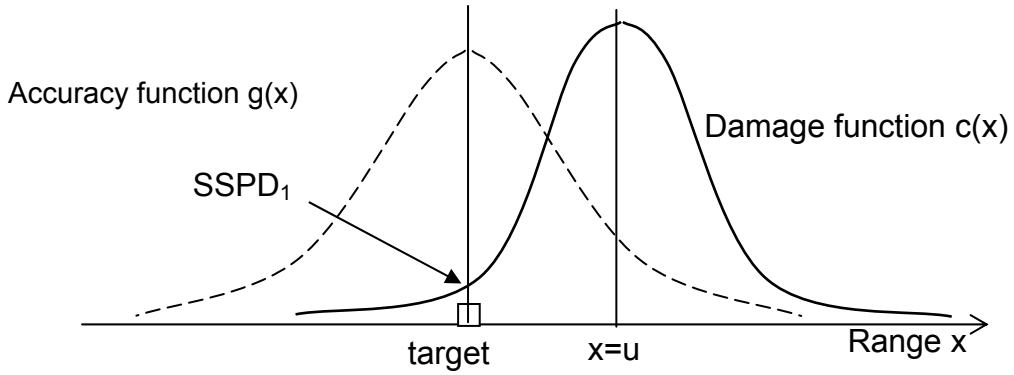

Figure 2.12: SSPD for a single target against a unitary target.
(From Ref. [1])

The point $SSPD_1$ marks the probability of damaging the target with a single sortie. This is the value of the damage function c(x) at the target. From our knowledge of delivery accuracy models, we may assume that the value u has a normal distribution for a large number of independent trial sorties. If we average the SSPD value over this large number of trials it will tend toward an expected value. Using knowledge of statistics and incorporating the damage function and accuracy distribution, we end up with equation (2.17).

$$SSPD_x = \int_{-\infty}^{\infty} c(x)g(x)dx \tag{2.17}$$

We can then substitute in the one-dimensional forms of the Carleton damage function and the accuracy function. These expressions are given in equations (2.18) and (2.19).

$$g(x) = \frac{1}{\sigma_x \sqrt{2\pi}} \exp\left[-\frac{x^2}{2\sigma_x{}^2} \right] \tag{2.18}$$

$$c(x) = \exp\left[-\frac{4x^2}{L'_{ET}{}^2} \right] \tag{2.19}$$

Substituting these equations into equation (2.17) and solving yields an expression for $SSPD_x$, which is given in equation (2.20).

$$SSPD_x = \frac{L'_{ET}}{\sqrt{8\sigma_x{}^2 + L'_{ET}{}^2}} \tag{2.20}$$

We have been only dealing in one dimension so far, but the results in the deflection direction are similar as shown in equation (2.21).

$$SSPD_y = \frac{W'_{ET}}{\sqrt{8\sigma_y{}^2 + W'_{ET}{}^2}} \tag{2.21}$$

Lastly, we find the total SSPD by combing the SSPD in range and in deflection.

$$SSPD = SSPD_x \times SSPD_y \tag{2.22}$$

Calculating the single sortie probability of damage is useful for determining the damage inflicted by a given weapon against a single target. However, this

method in determining the effectiveness of a weapon is not used for all weapon/ target interactions. These different effectiveness methods will be discussed later in this thesis.

THIS PAGE INTENTIONALLY LEFT BLANK

III. SINGLE WEAPON VERSUS UNITARY TARGET

A. JMEM METHOD

The Joint Munitions Effectiveness Manual (JMEM) uses a methodology similar to the case discussed in the Effectiveness section of Chapter II. Option 4 of the MATLAB m-file code provided in Appendix A reproduces the JMEM methodology for the case of single weapon released against unitary targets. The JMEM method uses the Carleton damage function to calculate the probability of damage to a target. This method first root sum squares the aiming errors and ballistic dispersion errors, to find a REP' and DEP' in feet, in the ground plane as shown in equations (3.1) and (3.2).

$$REP' = \sqrt{REP^2 + x_{br}{}^2} \tag{3.1}$$

$$DEP' = \sqrt{DEP^2 + x_{bd}{}^2} \tag{3.2}$$

From the user inputted values of the weapon MAE_F and the weapon impact angle, we can calculate the weapon radii WR_r and WR_d with equations (3.3) and (3.4).

$$WR_r = \sqrt{MAE_F \times \frac{a}{\pi}} \tag{3.3}$$

$$WR_d = \frac{WR_r}{a} \tag{3.4}$$

Where a is the aspect ratio of the weapon radii calculated from the weapon impact angle I, and shown in equation (3.5).

$$a = \frac{WR_r}{WR_d} = MAX\left[1 - 0.8\cos(I), 0.3\right] \tag{3.5}$$

The JMEM method for a single weapon against a unitary target calculates the effective target length L'_{ET} and effective target width W'_{ET} of the weapon lethal area based on the weapon radii. This is shown in equations (3.6) and (3.7).

$$L'_{ET} = 2 \times WR_r = 1.128 \sqrt{MAE_F \times \frac{a}{\pi}} \tag{3.6}$$

$$W'_{ET} = 2 \times WR_d = \frac{L'_{ET}}{a} \tag{3.7}$$

In order to calculate the single sortie probability of damage (SSPD), we use equation (3.8). This gives the probability of damage for a single weapon with user inputted lethality and accuracy against a unitary target.

$$SSPD = \frac{L'_{ET} \times W'_{ET}}{\sqrt{(17.6(REP')^2 + L'^2_{ET})(17.6(DEP')^2 + W'^2_{ET})}} \tag{3.8}$$

B. MONTE CARLO SIMULATIONS

A Monte Carlo simulation is a way of finding the probability of damaging a target by analyzing the results of a large number of sample weapon versus target trials. For example, to find the probability of damaging a target using a Monte Carlo approach, we look at a large number of independent trials of a weapon aimed at a desired impact point. The difference between iterations is a small random variation in the delivery accuracy of the weapon. This is shown in Figure 3.1.

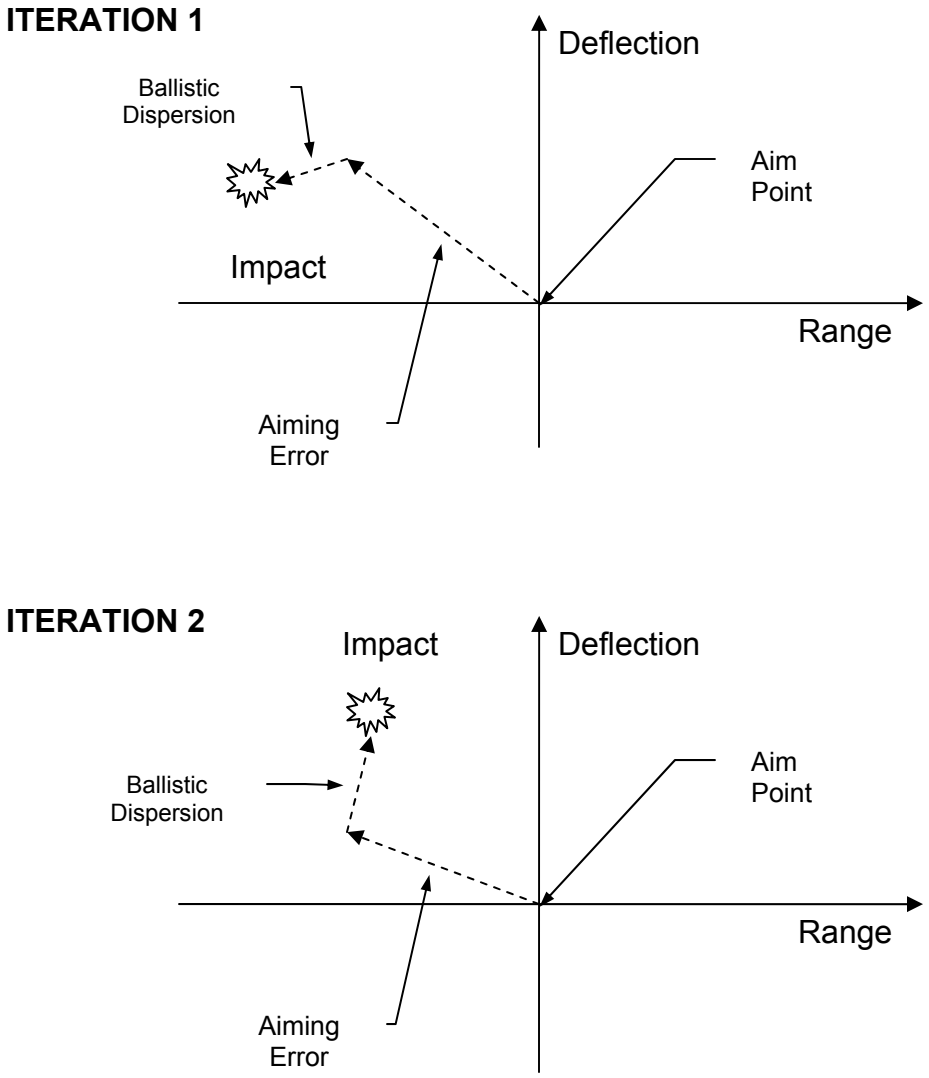

Figure 3.1: Two sample iterations of a Monte Carlo simulation.

Figure 3.1 shows two sample iterations of a Monte Carlo simulation. By slightly changing the aiming error and ballistic dispersion error of a weapon, the impact point of the weapon will change.

The variations in delivery accuracies are obtained by first calculating the standard deviations of the aiming error and ballistic dispersion in the ground plane. Given the user inputted REP and DEP, we obtain the standard deviation

of the aiming error in the ground plane for both the range and deflection directions (σ_{aim_range} and $\sigma_{aim\_deflection}$ respectively) through equations (3.9) and (3.10)

$$\sigma_{\text{aim\_range}} = \frac{\text{REP}}{0.6745} \tag{3.9}$$

$$\sigma_{\text{aim\_deflection}} = \frac{\text{DEP}}{0.6745} \tag{3.10}$$

Given the user inputted slant range (SR), impact angle (I), and ballistic dispersion measured in mils in the normal plane (σ_b), we calculate the standard deviation of the ballistic dispersion in the ground plane for both the range and deflection directions (x_{br} and x_{bd}) through equations (3.11) and (3.12).

$$x_{br} = \frac{0.6745 \times SR \times \sigma_b}{1000 \sin(I)} \tag{3.11}$$

$$x_{bd} = \frac{0.6745 \times SR \times \sigma_b}{1000} \tag{3.12}$$

All four of the delivery accuracy standard deviations (σ_{aim_range}, $\sigma_{aim\_deflection}$, $x_{br,}$ and x_{bd}) are then multiplied by a random number, and the standard deviations are added together in the range and deflection directions. This results in the range and deflection location of the impact point, and is shown in equations (3.13) and (3.14).

$$\text{Impact Point}_{\text{RANGE}} = (\sigma_{\text{aim\_range}} \times random\#) + (x_{br} \times random\#) \tag{3.13}$$

$$\text{Impact Point}_{\text{DEFLECTION}} = (\sigma_{\text{aim\_deflection}} \times random\#) + (x_{bd} \times random\#) \tag{3.14}$$

The random# term in the previous equations is a random number from a normal distribution. This normal distribution has a mean value of zero and a standard deviation of one. Using this normal distribution allows us to assume that the

average probability of damage for a large number of weapon trials will equal the expected value of the damage function at the desired aim point.

This thesis will use two approaches in modeling the weapon lethal area in a Monte Carlo simulation. In each simulation, the ground plane and the lethal area are divided into 1ft by 1ft cells. The first weapon model will use a weapon lethal area matrix with each matrix cell populated with a probability of damage obtained by the Carleton damage function. An example of a lethal area matrix populated by the Carleton damage function is given in Figure 3.2.

0.0000	0.0003	0.0013	0.0036	0.0059	0.0059	0.0036	0.0013	0.0003	0.0000
0.0003	0.0022	0.0098	0.0266	0.0439	0.0439	0.0266	0.0098	0.0022	0.0003
0.0013	0.0098	0.0439	0.1194	0.1969	0.1969	0.1194	0.0439	0.0098	0.0013
0.0036	0.0266	0.1194	0.3247	0.5353	0.5353	0.3247	0.1194	0.0266	0.0036
0.0059	0.0439	0.1969	0.5353	0.8825	0.8825	0.5353	0.1969	0.0439	0.0059
0.0059	0.0439	0.1969	0.5353	0.8825	0.8825	0.5353	0.1969	0.0439	0.0059
0.0036	0.0266	0.1194	0.3247	0.5353	0.5353	0.3247	0.1194	0.0266	0.0036
0.0013	0.0098	0.0439	0.1194	0.1969	0.1969	0.1194	0.0439	0.0098	0.0013
0.0003	0.0022	0.0098	0.0266	0.0439	0.0439	0.0266	0.0098	0.0022	0.0003
0.0000	0.0003	0.0013	0.0036	0.0059	0.0059	0.0036	0.0013	0.0003	0.0000

Figure 3.2: Example of lethal area matrix populated by Carleton damage function.

The dimensions of this lethal area matrix are dictated by the weapon radii. For the Monte Carlo simulations in this thesis, the size of the lethal area matrix was set to $4WR_r$ by $4WR_d$ because outside of these dimensions, the probability of damage from the Carleton damage function is negligible.

The second Monte Carlo simulation will use the rectangular cookie cutter approximation of the Carleton damage function. This will also use a lethal area matrix, but the lethal area matrix cells will all be populated by a probability of damage equal to one. The dimensions of this lethal area are L_{ET} and W_{ET} as explained in Chapter II Section B Lethal Areas. These dimensions are calculated from the user inputs of MAE_F and weapon impact angle (which determines the aspect ratio a as in equation (3.5)) as shown in equations (3.15) and (3.16).

$$L_{ET} = \sqrt{MAE_F \times a} \tag{3.15}$$

$$W_{ET} = L_{ET} / a \tag{3.16}$$

Again, the probability of damage inside of the lethal area matrix defined by L_{ET} and W_{ET} is equal to one, and the probability of damage outside of these dimensions is zero.

The details of how each Monte Carlo simulation determines a probability of damage will be explained in the following two sections.

1. Modeling Weapons with Carleton Damage Function

The first Monte Carlo simulation models a single weapon with a lethal area matrix with 1ft by 1ft cells populated with a probability of damage from the Carleton damage function. The first step in this simulation is to create the weapon lethal area matrix from user inputs. The result of which gives a lethal area matrix similar to the one shown in Figure 3.2. This weapon matrix is held at a constant throughout the Monte Carlo simulation. The location of the target in which we are aiming is what varies in each iteration of the simulation. The process of varying the target location was explained above with equations (3.9) through (3.14).

Examples of iterations of the Monte Carlo simulation are shown in Figure 3.3. Here we have a lethal area matrix populated by the Carleton damage function fixed in the ground plane. For this example the location of the target, which is based on the delivery accuracy of the weapon, is 3 feet past the aimpoint in the range direction, and 2 feet past in the deflection direction. The corresponding cell of this target location is shaded in dark grey. This 0.1194 value is the probability of damaging the target for this iteration. The next iteration finds the target location to be 4 feet short of the aim point in the range direction, and 3 feet short in the deflection direction. This target location is shaded in light grey and has a probability of damage equal to 0.0098.

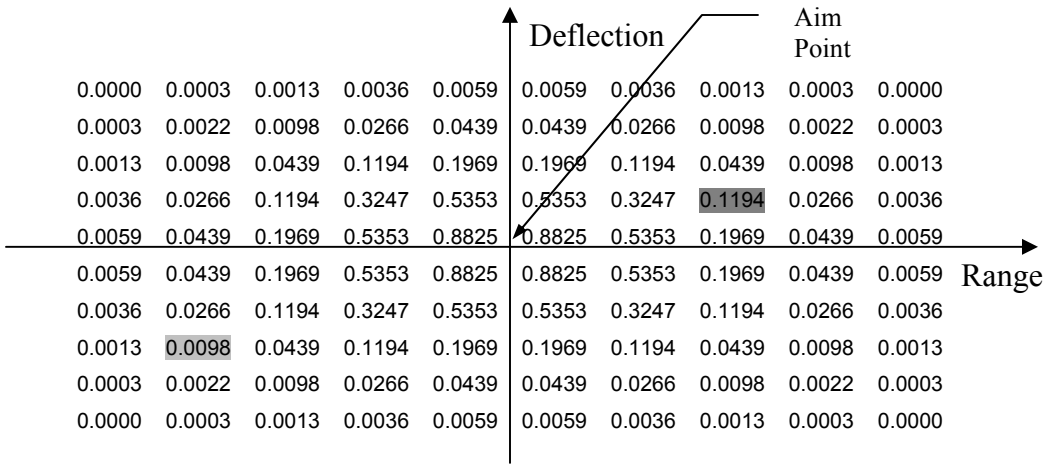

Figure 3.3: Example iteration of Monte Carlo simulation using Carleton damage function to fill lethal area matrix.

For the full Monte Carlo simulation, we would repeat this for n iterations, each individual iteration (i) resulting in its own probability of damage $P_{d/i}$. The value n is the total number of iterations used in the Monte Carlo simulation. The final probability of damage for the Monte Carlo simulation is the average of all of the $P_{d/I}$ shown in equation (3.17).

$$P_d = \frac{\sum_{i=1}^{n} P_{d/i}}{n} \tag{3.17}$$

The number of iterations (n) used in the Monte Carlo simulation used to compare the Monte Carlo and JMEM methods was set to ten million. In running the Monte Carlo simulation multiple times, we ensure that the resulting P_d from each simulation did not change. This shows that the simulation was converging to a single value, which we would expect. After testing different values of n, it was found that ten million provides for this convergence.

A Monte Carlo simulation with a weapon modeled with a lethal area matrix fill by the Carleton damage function is presented in Appendix A, Option 1.

2. Modeling Weapons with Rectangular Cookie Cutter Approximation

The second Monte Carlo simulation models a single weapon as a lethal area matrix rectangular cookie cutter. The dimensions of this lethal area matrix are determined from user inputs as described earlier in this chapter in equations (3.5), (3.15), and (3.16). As in the previous Monte Carlo simulation, we place this lethal area matrix on the ground plane, and for each iteration of the simulation, the location of the target changes due to variations in the delivery accuracy of the weapon.

Examples of iterations of the Monte Carlo simulation are shown in Figure 3.4. Here we have a lethal area matrix with dimensions L_{ET} and W_{ET} fixed in the ground plane. Since this is a rectangular cookie cutter approximation method, we have probability of damage equal to one inside of the lethal area matrix, and zero outside of the lethal area matrix. In this example we assume that the user inputted values of the weapon effectiveness result in L_{ET} = 6ft and W_{ET} = 6ft. This 6ft by 6ft rectangular cookie cutter is shown In Figure 3.4. For this example, the location of the target, which is based on the delivery accuracy of the weapon, is 3 feet past the aim point in the range direction, and 2 feet past in the deflection direction. The corresponding cell of this target location is shaded in dark grey. Since this target location is inside the weapon lethal area, its probability of damage is 1.0. Say the next iteration finds the target location to be 4 feet short of the aim point in the range direction, and 3 feet short in the deflection direction. This target location is shaded in light grey and has a probability of damage equal to 0.0.

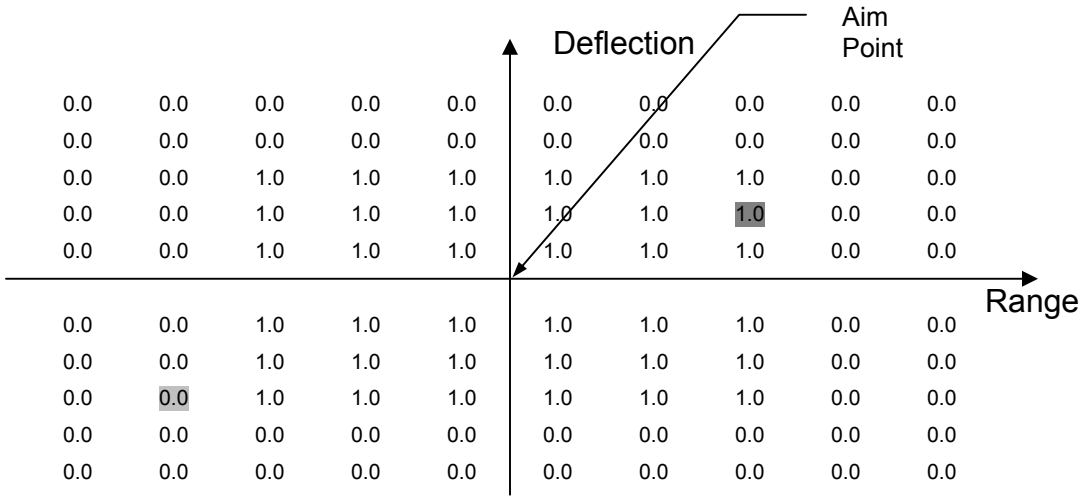

Figure 3.4: Example iteration of Monte Carlo simulation using a rectangular cookie cutter lethal area matrix.

For a full Monte Carlo simulation, the total P_d is found as it was before with equation (3.17). For this simulation however, we will only sum ones and zeros because those are the only values $P_{d/I}$ can have. Again, the Monte Carlo simulation using a rectangular cookie cutter lethal area uses ten million iterations to achieve a convergence of results.

C. RESULTS OF UNITARY TARGET COMPARISONS

The comparisons done this section are to find the differences between weapon effectiveness methodologies for the case of a single weapon against unitary targets. In order to get a wide array of results each comparison was done for different user inputted values of MAE_F, REP & DEP and ballistic dispersion in mils, in the normal plane. The weapon impact angle and release slant range were kept at a constant 45° and 10,000 ft respectively for each comparison.

1. JMEM versus Carleton Damage Function

The first comparison to the JMEM methodology of calculating the effectiveness of a single weapon against a unitary target is to compare it to a Monte Carlo simulation using the Carleton damage function. This is a particularly useful comparison because the current JMEM method is based on the Carleton damage function. Since this is the case, we would expect the Monte Carlo simulation representing the weapon as the Carleton damage function to produce similar results to the JMEM. Tables 3.1 and 3.2 present the results for the P_K found for both the JMEM method and Monte Carlo simulation. The shaded values in the Tables are obtained from the JMEM method, while the unshaded values are those of the Monte Carlo simulation.

$MAE_F=5000ft^2$	Ballistic Dispersion (mils in normal plane)		
	0	3	5
REP=25ft & DEP=50ft	0.2236	0.1787	0.1363
	0.2239	0.1477	0.0978
REP=25ft & DEP=25ft	0.3384	0.2621	0.1905
	0.3386	0.2091	0.1284
REP=50ft & DEP=25ft	0.1834	0.1616	0.1335
	0.1837	0.1419	0.101
REP=50ft & DEP=50ft	0.1211	0.1103	0.0956
	0.1215	0.1002	0.077

Table 3.1: Results of JMEM and Monte Carlo for $MAE_F=5000ft^2$

MAE_F=1000ft^2	Ballistic Dispersion (mils in normal plane)		
	0	3	5
REP=25ft & DEP=50ft	0.0542	0.042	0.031
	0.0546	0.0339	0.0217
REP=25ft & DEP=25ft	0.0999	0.0719	0.0483
	0.1003	0.0543	0.0304
REP=50ft & DEP=25ft	0.0509	0.0427	0.0333
	0.0511	0.036	0.0237
REP=50ft & DEP=50ft	0.0276	0.0249	0.0213
	0.0278	0.0225	0.0169

Table 3.2: Results of JMEM and Monte Carlo for MAE_F=1000ft^2

It can be seen from Tables 3.1 and 3.2 that the Monte Carlo simulation does not produce the same results as the JMEM. This is especially true for the case of a large ballistic dispersion. The lethal area as described by the MAE_F does not seem to alter the error between the JMEM and Monte Carlo result, which is expected since both methodologies use the MAE_F to find the weapon radii used to describe the lethality of the weapon. This leaves the delivery accuracy as the prime reason for the differences in results. These differences are especially seen for the case of large ballistic dispersion. As the ballistic dispersion is lowered down to zero, we see a decrease in the error between the two methods. In the cases of no ballistic dispersion for any of the four aiming error combinations, we obtain results that are within a tenth of a percent of each other for the two methods. This leads us to believe that the handling of the ballistic dispersion is the main reason for differences in the results. There are two viable explanations for this. One possibility is that the Monte Carlo simulation does not correctly represent the ballistic dispersion by varying it by a random number in each iteration of the code. Another reason may be from the fact that the JMEM method root sums squares the aiming error with the ballistic dispersion (measured in feet in the ground plane). It may be that these ballistic errors cannot be combined with the aiming error probables as originally thought.

It is also interesting to note that the Monte Carlo result is constantly greater that the JMEM result for non-zero ballistic dispersion cases, which shows that the JMEM is a more conservative estimate of damage.

2. JMEM versus Rectangular Cookie Cutter Approximation

The last comparison for single weapons against unitary targets is between the JMEM method and a Monte Carlo simulation which represents the weapon as a rectangular cookie cutter. Though interesting, the results of this comparison have very little practical meaning. JMEM weaponeering methodologies make use of the rectangular cookie cutter approximation in cases where the Carleton damage function itself to too complicated to utilize. The case we are looking at now of a single weapon versus and unitary target does not need to use an approximation, and the JMEM method uses the Carleton damage function as was explained in Chapter III Section A. Modeling the weapon as a rectangular cookie cutter area would only lessen the fidelity of the JMEM method for this weapon/target case. The results of comparing the Monte Carlo simulation and the JMEM method are given in Tables 3.3 and 3.4. The shaded values in the Tables are obtained from the JMEM method, while the unshaded values are those of the Monte Carlo simulation.

$MAE_F=5000ft^2$	Ballistic Dispersion (mils in normal plane)		
	0	3	5
REP=25ft & DEP=50ft	0.2443	0.1921	0.1438
	0.2239	0.1477	0.0978
REP=25ft & DEP=25ft	0.3937	0.2975	0.2108
	0.3386	0.2091	0.1284
REP=50ft & DEP=25ft	0.2062	0.1801	0.146
	0.1837	0.1419	0.101
REP=50ft & DEP=50ft	0.1218	0.1161	0.0997
	0.1215	0.1002	0.077

Table 3.3: Results of JMEM and Monte Carlo for $MAE_F=5000ft^2$

$MAE_F=1000ft^2$	Ballistic Dispersion (mils in normal plane)		
	0	3	5
REP=25ft & DEP=50ft	0.0528	0.0408	0.0298
	0.0546	0.0339	0.0217
REP=25ft & DEP=25ft	0.1007	0.0713	0.0475
	0.1003	0.0543	0.0304
REP=50ft & DEP=25ft	0.0509	0.0422	0.0326
	0.0511	0.036	0.0237
REP=50ft & DEP=50ft	0.0266	0.024	0.0206
	0.0278	0.0225	0.0169

Table 3.4: Results of JMEM and Monte Carlo for $MAE_F=1000ft^2$

As expected, the results from the Monte Carlo simulation and the JMEM method do not match. This is due mainly to the reasons discussed before concerning the fact that the rectangular cookie cutter method makes approximations in the Carleton damage function that the JMEM method does not make. Along with this reason we suspect that the ballistic dispersion accounts for some of the error between the methods, as it did for the Monte Carlo simulation with the Carleton damage function. Also similar to the previous comparison is that the Monte Carlo simulation consistently gives greater damage results than the JMEM method, leading to the conclusion that the JMEM is more conservative than the Monte Carlo for all delivery accuracy inputs.

A better way of comparing the rectangular cookie cutter Monte Carlo simulation and JMEM method is to look at the JMEM for single weapons versus area targets. The concept of single weapons versus area target will be discussed in detail Chapter IV. When the JMEM calculates a probability of damage for a single weapon against a unitary target, the target dimensions inputted into OEM 5.0 (Reference 2) are a 0 ft by 0 ft target. When these dimensions are inputted, the OEM program uses the Carleton damage function as described earlier in this chapter for a single weapon against a unitary target.

When anything other than 0 ft by 0 ft target is inputted, the OEM program uses the JMEM method for single weapons against area targets. This methodology for area targets uses an approximation of the Carleton damage function to find the probability of damage. For the comparison given below in Tables 3.5 and 3.6, a target dimension of .01 ft by .01 ft was inputted into OEM 5.0 to represent a unitary target with a small area target. Now we are able to compare results of more similar methods, rather than comparing a rectangular cookie cutter Monte Carlo simulation with the Carleton damage function.

$MAE_F=5000ft^2$	Ballistic Dispersion (mils in normal plane)		
	0	3	5
REP=25ft & DEP=50ft	0.2443	0.1921	0.1438
	0.2503	0.1601	0.104
REP=25ft & DEP=25ft	0.3937	0.2975	0.2108
	0.402	0.2379	0.1407
REP=50ft & DEP=25ft	0.2062	0.1801	0.146
	0.2109	0.1595	0.1103
REP=50ft & DEP=50ft	0.1218	0.1161	0.0997
	0.1313	0.1073	0.0815

Table 3.5: Results of JMEM (small area) and Monte Carlo for $MAE_F=5000ft^2$

$MAE_F=1000ft^2$	Ballistic Dispersion (mils in normal plane)		
	0	3	5
REP=25ft & DEP=50ft	0.0528	0.0408	0.0298
	0.0546	0..0347	0.022
REP=25ft & DEP=25ft	0.1007	0.0713	0.0475
	0.1071	0.0567	0.0313
REP=50ft & DEP=25ft	0.0509	0.0422	0.0326
	0.0541	0.0375	0.0244
REP=50ft & DEP=50ft	0.0266	0.024	0.0206
	0.0285	0.023	0.0172

Table 3.6: Results of JMEM (small area) and Monte Carlo for $MAE_F=1000ft^2$

We can see from Tables 3.3, 3.4, 3.5 and 3.6 that the results for the Monte Carlo simulation match somewhat better with the JMEM method for a small area target than they do with the unitary target JMEM model. This was expected due to the similarities between the rectangular cookie cutter method and the JMEM method for area targets in approximating the Carleton damage function. Differences in the results are from the sources previously discussed in this section, as well the fact that we are dealing with different target areas. The Monte Carlo uses a 1ft by 1 ft cell to represent the unitary target, while we have to use a .01 ft by .01 ft area to calculate the JMEM method probability of damage.

THIS PAGE INTENTIONALLY LEFT BLANK

IV. SINGLE WEAPON VERSUS AREA TARGET

The area target is best described as an area of targets, or a certain grouping of target elements. An area target could contain anything from ground troops to tanks to components of a surface to air missile site. The key to analyzing area targets is that we must assume that the target elements are uniformly distributed within the prescribed area.

When determining the amount of damage done to an area target, we do not calculate the single sortie probability of damage as done with unitary targets. For area targets we measure a quantity called the expected fractional damage, or EFD. Since we are dealing with an area of target elements, we are concerned with what fraction of these target elements are damaged for a given weapon impact. This is shown in Figure 4.1 below. Figure 4.1 shows an area target consisting of twelve target elements (denoted by stars), which are uniformly distributed inside the target area dimensions.

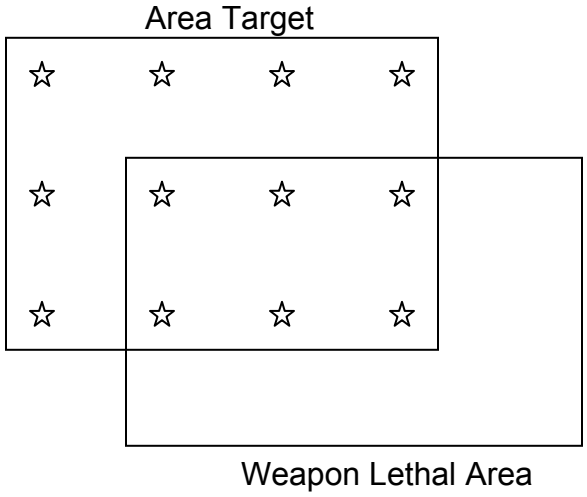

Figure 4.1: Area target partially covered by weapon lethal area. (After Ref. [1])

In this case, 50% of the target elements are covered by the weapon lethal area rectangle. If we assume the probability of damage inside of the weapon lethal area, or P_{CD}, is 1.0, we know the fractional damage done in Figure 4.1. This will

remain true for any case in which the weapon lands at a point such that half of the area target elements are covered by the weapon lethal area. The EFD is calculated from the expected fractional coverage of the weapon lethal area on the area target for a large number of trials, shown in equation (4.1)

$$EFD = E(F_C) \times P_{CD} \tag{4.1}$$

The same comparisons done for unitary target situations earlier will be done for area targets. The methods analyzed are Monte Carlo simulations with two ways of modeling weapon effectiveness, and the JMEM method of calculating EFD.

A. JMEM METHOD

The first step in the JMEM method for area targets is dealing with the delivery accuracy of the weapon. The delivery accuracy is handled in the same fashion as it is for unitary targets by root sum squaring the aiming errors and ballistic dispersion to find REP' and DEP'.

We will first look at only at the problem in the range direction. The area target is assigned length and width dimensions of L_A and W_A respectively. A sample weapon/target interaction is presented in Figure 4.2. The figure shows a target length of L_A with a weapon impacting the ground a distance u from the center of the area target. The weapon is represented by the rectangular equivalent of the Carleton damage function, where the probability of damage inside of the rectangular area is equal to unity. As before when calculating SSPD, the accuracy function of the weapon is represented by g(x), and the rectangular damage function has length of L_{ET} in our one dimensional case.

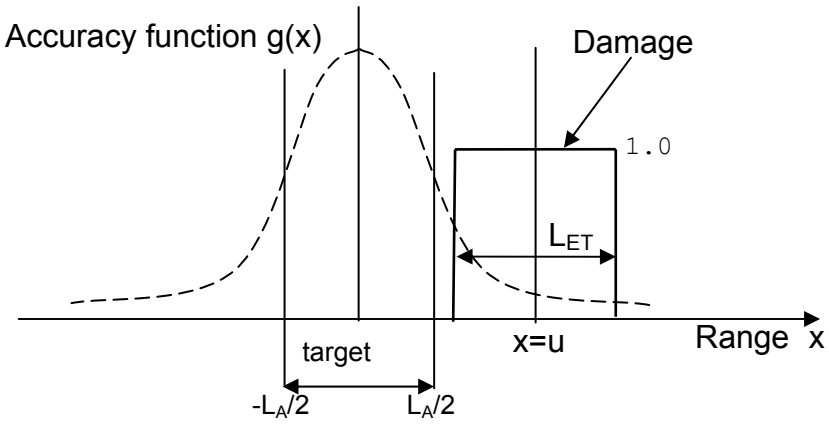

Figure 4.2: Rectangular lethal area against an area target. (From Ref. [1])

The example shown in Figure 4.2 would yield no damage to the area target because the weapon lethal area does not overlap the area target. The next weapon drop however, may have an impact point that causes the lethal area to partially cover the area target, which we have defined as fractional coverage. In order to determine the damage caused by a weapon, we will need to know the details of the fractional coverage.

With a weapon dropped against an area target, there are an infinite number of ways in which the weapon lethal area can either partially overlap, totally cover, or not overlap the area target. This creates a problem when dealing with weapon lethal areas that have different area target dimensions. Through a detailed explanation provided in Reference 1, we are able to simplify the problem of weapon overlap scenarios. This is done first by defining a parameter called the effective pattern length and width. These are found in equations (4.2) and (4.3).

$$L_{EP} = \max(L_{ET}, L_A) \tag{4.2}$$

$$W_{EP} = \max(W_{ET}, W_A) \tag{4.3}$$

The next finding is that there is a way to summarize the fractional coverage of a weapon lethal area onto an area target as a function of x, or the distance from the weapon impact point to the center of the area target. This summary is shown graphically in Figure 4.3.

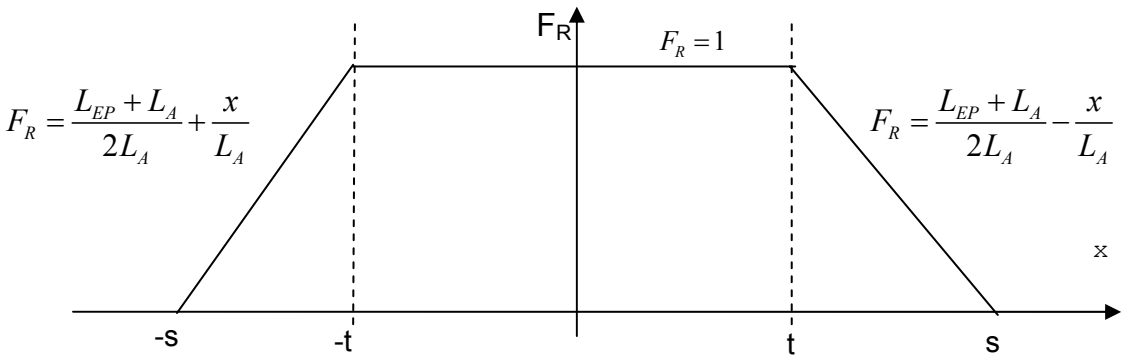

Figure 4.3: Fractional coverage in the range direction. (From Ref. [1])

The values s and t are shorthand notations given in equations (4.4) and (4.5).

$$s = \frac{L_{EP} + L_A}{2} \tag{4.4}$$

$$t = \frac{L_{EP} - L_A}{2} \tag{4.5}$$

We are then able to find the expected value of the fractional coverage since we know the fractional coverage given a value of x (from Figure 4.3) and that the value x itself is a product of the normally distributed aiming error g(x). We can then define the expected fractional coverage $E(F_R)$ in equation (4.6)

$$E(F_R) = \int_{-\infty}^{\infty} F_R(x) g(x) dx \tag{4.6}$$

Equation (4.6) can be rewritten b\y substituting in the value of F_R from Figure 4.3 and using the probability density function of the aiming error g(x) for a normal distribution with zero mean (given by equation (4.7))

$$g(x) = \frac{1}{\sigma_x \sqrt{2\pi}} \exp\left[\frac{-x^2}{2\sigma_x}\right] \tag{4.7}$$

We can then expand equation (4.6) to equation (4.8)

$$E(F_R) = \int_{-\infty}^{+\infty} F_R(x) g(x) dx = \frac{1}{\sigma_x \sqrt{2\pi}} \left\{ \begin{array}{l} \int_{-t}^{t} \exp\left(\frac{-x^2}{2\sigma_x{}^2}\right) dx \\ + \int_{-s}^{-t} \left[\frac{L_{EP} + L_A}{2L_A} + \frac{x}{L_A}\right] \exp\left(\frac{-x^2}{2\sigma_x{}^2}\right) dx \\ + \int_{t}^{s} \left[\frac{L_{EP} + L_A}{2L_A} - \frac{x}{L_A}\right] \exp\left(\frac{-x^2}{2\sigma_x{}^2}\right) dx \end{array} \right\} \tag{4.8}$$

This may also be written as equation (4.9)

$$E(F_R) = \frac{1}{\sigma_x \sqrt{2\pi}} \left\{ \begin{array}{l} \int_{-t}^{t} \exp\left(\frac{-x^2}{2\sigma_x{}^2}\right) dx + \left[\frac{L_{EP} + L_A}{2L_A}\right] \int_{-s}^{-t} \exp\left(\frac{-x^2}{2\sigma_x{}^2}\right) dx \\ + \left[\frac{L_{EP} + L_A}{2L_A}\right] \int_{t}^{s} \exp\left(\frac{-x^2}{2\sigma_x{}^2}\right) dx + \int_{-s}^{-t} \frac{x}{L_A} \exp\left(\frac{-x^2}{2\sigma_x{}^2}\right) dx \\ - \int_{t}^{s} \frac{x}{L_A} \exp\left(\frac{-x^2}{2\sigma_x{}^2}\right) dx \end{array} \right\} \tag{4.9}$$

Equation (4.9) is easier solved in parts, so a shorthand notation is used to split it into five smaller integrals. This shorthand is shown in equation (4.10).

$$E(F_R) = \{I_1 + I_2 + I_3 + I_4 - I_5\} \tag{4.10}$$

The first three integrals in equation (4.10) involve the normal cumulative distribution given in equation (4.7). In order to compute the normal cumulative distribution function (CDF) we use the MATLAB function normcdf.m. This function takes the user inputs of x, μ (mean of the miss distance), and σ

(standard deviation of the miss distance) and gives the value of the CDF. In our case, each integral has the same zero mean and the same standard deviation calculated from the delivery accuracy of the weapon. Using this function as well as our previously defined shorthand notation, we solve the first three integrals in equations (4.11), (4.12), and (4.13).

$$I_1 = [normcdf(t) - normcdf(-t)] \tag{4.11}$$

$$I_2 = \frac{L_{EP} + L_A}{2L_A}[normcdf(-t) - normcdf(-s)] \tag{4.12}$$

$$I_3 = \frac{L_{EP} + L_A}{2L_A}[normcdf(s) - normcdf(t)] \tag{4.13}$$

By rearranging elements of the fourth and fifth integrals, we can solve the difference between the integrals analytically as done in Reference 1. The result of this is given in equation (4.14).

$$I_4 - I_5 = \frac{2\sigma_x}{L_A\sqrt{2\pi}}[e^{-\left(\frac{s}{\sigma_x\sqrt{2}}\right)^2} - e^{-\left(\frac{t}{\sigma_x\sqrt{2}}\right)^2}] \tag{4.14}$$

The same procedure is used for the deflection direction. Equations (4.15) through (4.21) summarize the equations to find $E(F_D)$.

$$s = \frac{W_{EP} + W_A}{2} \tag{4.15}$$

$$t = \frac{W_{EP} - W_A}{2} \tag{4.16}$$

$$E(F_D) = \{I_1 + I_2 + I_3 + I_4 - I_5\} \tag{4.17}$$

$$I_1 = [normcdf(t) - normcdf(-t)] \tag{4.18}$$

$$I_2 = \frac{W_{EP} + W_A}{2W_A}[normcdf(-t) - normcdf(-s)] \tag{4.19}$$

$$I_3 = \frac{W_{EP} + W_A}{2W_A}[normcdf(s) - normcdf(t)] \tag{4.20}$$

$$I_4 - I_5 = \frac{2\sigma_y}{W_A\sqrt{2\pi}}[e^{-\left(\frac{s}{\sigma_y\sqrt{2}}\right)^2} - e^{-\left(\frac{t}{\sigma_y\sqrt{2}}\right)^2}] \tag{4.21}$$

Once we have the expected fractional coverage in both the range and deflection direction we combine them in equation (4.22) to find the total expected fractional coverage.

$$E(F_C) = E(F_R) \times E(F_D) \tag{4.22}$$

Given the fractional coverage of the weapon, we multiple it by the reliability of the weapon (R) and the conditional probability of damage (P_{CD}) inside of the weapon lethal area as shown in equation (4.23). The P_{CD} term is equal to unity when the weapon lethal area is larger than the target area. When the target area is larger, the P_{CD} must go down in order to conserve the lethality of the weapon. This is explained in detail in Reference 1.

$$EFD = E(F_C) \times P_{CD} \times R = E(F_C) \times \left[\frac{A_{ET}}{L_{EP} \times W_{EP}}\right] \times R \tag{4.23}$$

B. MONTE CARLO SIMULATIONS

For the case of single weapons versus area targets, we run two Monte Carlo simulations to determine the probability of damaging the target. The first Monte Carlo represents the weapon as a lethal area matrix consisting of 1 ft by 1 ft cells populated with probability of damage values populated from the Carleton damage function. The second Monte Carlo simulation represents the weapon with a lethal area matrix which has dimensions of a rectangular cookie cutter approximation of the Carleton damage function. The details of these two simulations are explained in detail in Chapter 3 Section B. This section also explains the process of varying the delivery accuracy of the weapon in each iteration of the Monte Carlo Simulation.

1. Modeling Weapons with Carleton Damage Function

The first Monte Carlo simulation models the weapon as a lethal area matrix populated with the Carleton damage function. This process begins with the user inputting values of the MAE_F and impact angle (I) of the weapon. This information provides us the means to calculate the aspect ratio (a) of the weapon radii which gives us the weapon radii (WR_r and WR_d), as shown in equations (4.24), (4.25), and (4.26).

$$a = \frac{WR_r}{WR_d} = MAX\left[1 - 0.8\cos(I), 0.3\right] \tag{4.24}$$

$$WR_r = \sqrt{MAE_F \times \frac{a}{\pi}} \tag{4.25}$$

$$WR_d = \frac{WR_r}{a} \tag{4.26}$$

Now that the weapon radii are calculated, the lethal area matrix can be filled with values from the Carleton damage function in equation (4.27). The values x

and y are the distances in the range and deflection direction respectively from the weapon impact point. For our Monte Carlo simulation, the weapon lethal area matrix is fixed in the ground plane and we will look at the distance from the target to the center of the weapon impact point in order to find the probability of damaging the target.

$$P(x,y) = \exp -\left[\frac{x^2}{WR_r^{\,2}} + \frac{y^2}{WR_d^{\,2}} \right] \tag{4.27}$$

An example iteration of a Monte Carlo simulation modeling the weapon as a lethal area matrix filled with the Carleton damage function are shown in Figure 4.4. The target we are dealing with is 3 ft by 3 ft for this example. We first look at a target location where the center of the target is placed at 3 ft past the aim point in the range direction, and 1 ft short of the aim point in the deflection direction. Again, this target placement based on the delivery accuracy of the weapon. By knowing the location of the center of the target, we can then superimpose the 3 ft by 3ft target onto the lethal area matrix on the ground plane. We superimpose the target area onto the lethal area by dividing the target area into 1ft by 1ft cells. This way, each division of the target area will overlap a specific value of the lethal area matrix. This is shown by the dark grey 3 ft by 3ft target area in Figure 4.4.

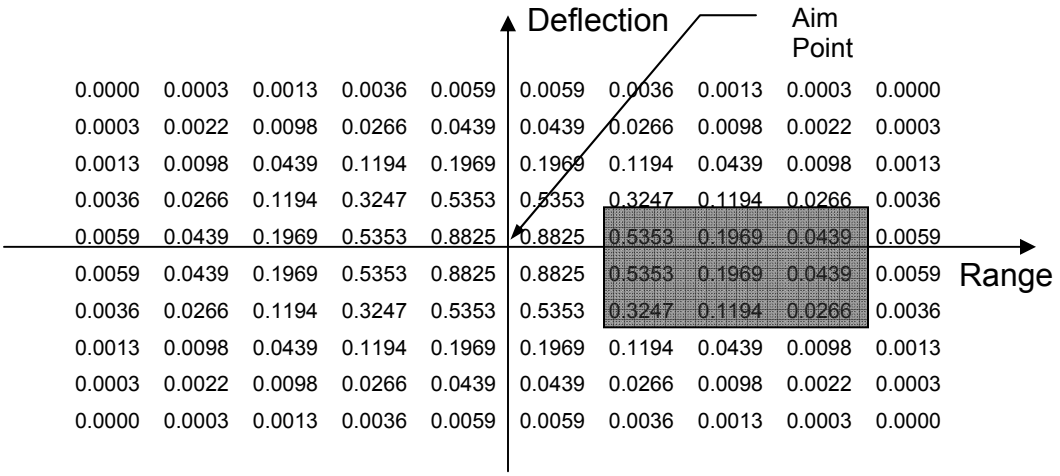

Figure 4.4: Example iteration of Monte Carlo simulation using Carleton damage function to fill lethal area matrix against an area target.

Finding the probability of damaging the area target for each iteration, we need to account for the probability of damage in each 1ft by 1 ft division of the target area. The total probability of damage is obtained by averaging the individual target cells. This is shown in equation (4.28).

$$P_{d/i} = \frac{\sum_{j=1}^{j} P_{d/cell(j)}}{j} \tag{4.28}$$

Equation (4.28) sums the probability of damage $P_{d/cell(j)}$ from each of the j total number of 1 ft by 1 ft cells in the area target. It then divides this sum by the total number of cells j to find the average $P_{d/i}$, which is the probability of damaging the area target for a given iteration i. For our example this would give results shown in equation (4.29).

$$P_{d/i} = \frac{0.5353 + 0.1969 + 0.0439 + 0.5353 + 0.1969 + 0.0439 + 0.3247 + 0.1194 + 0.0266}{9} = .2247 \tag{4.29}$$

For the full Monte Carlo simulation, we repeat this for n iterations, each individual iteration (i) resulting in its own probability of damage $P_{d/i}$. The value n is the total number of iterations used in the Monte Carlo simulation. The final probability of damage for the Monte Carlo simulation is the average of all of the $P_{d/I}$ shown in equation (4.30)

$$P_d = \frac{\sum_{i=1}^{n} P_{d/i}}{n} \tag{4.30}$$

A Monte Carlo simulation with a weapon modeled with a lethal area matrix fill by the Carleton damage function is presented in Appendix A, Option 1. This is the same option for the single weapon versus unitary target case, so the user must input an area target size.

2. Modeling Weapons with Rectangular Cookie Cutter Approximation

The second Monte Carlo simulation to find the probability of damaging an area target models a single weapon as a lethal area matrix rectangular cookie cutter. The dimensions of this lethal area matrix are determined from user inputs as described earlier in Chapter III, equations (3.5), (3.15), and (3.16). As in the previous Monte Carlo simulation, we place this lethal area matrix on the ground plane, and for each iteration of the simulation, the location of the target changes due to variations in the delivery accuracy of the weapon.

An example iteration of the Monte Carlo simulation representing the weapon as a rectangular cookie cutter approximation of the Carleton damage function is given in Figure 4.5. For this sample case, we assume that the user inputted values of weapon lethality produce a L_{ET} = 6ft and a W_{ET} = 6ft. These are the dimensions of the rectangular cookie cutter lethal area. This lethal area matrix is then fixed to the ground plane. For this example, the center of the 3 ft

by 3 ft area target is 3 ft short of the aim point in the range direction, and 2 ft short of aim point in the deflection direction. This location is based on the delivery accuracy of the weapon. This is shown by the dark grey 3 ft by 3ft target area in Figure 4.5.

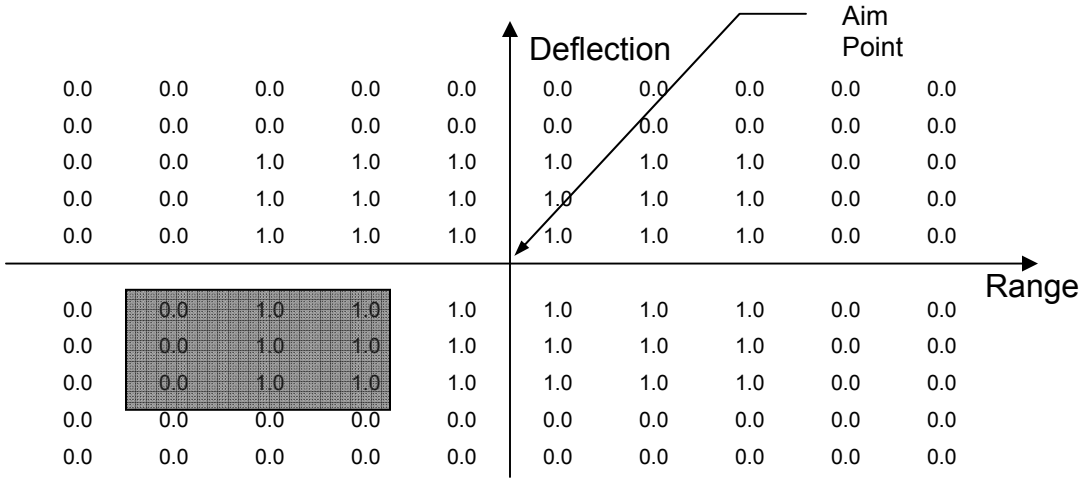

Figure 4.5: Example iteration of Monte Carlo simulation using a rectangular cookie cutter lethal area matrix against an area target.

Again, finding the probability of damaging the area target for each iteration, we need to account for the probability of damage in each 1ft by 1 ft division of the target area. The total probability of damage is obtained by averaging the individual target cells. This is shown above in equation (4.28). For our example iteration we would compute $P_{d/I}$ = .6667 as shown in equation (4.31).

$$P_{d/i} = \frac{0.0+1.0+1.0+0.0+1.0+1.0+0.0+1.0+1.0}{9} = .6667 \qquad (4.31)$$

As before, for the full Monte Carlo simulation, we repeat this for n iterations, each individual iteration (i) resulting in its own probability of damage $P_{d/i}$. The final probability of damage for the Monte Carlo simulation is the average of all of the $P_{d/I}$ shown in equation (4.30).

A Monte Carlo simulation with a weapon modeled by a lethal area matrix filled by a rectangular cookie cutter approximation of the Carleton damage function is presented in Appendix A, Option 2. This is the same option for the single weapon versus unitary target case, so the user must input an area target size.

C. RESULTS OF AREA TARGET COMPARISONS

The comparisons done this section are to find the differences between weapon effectiveness methodologies for the case of a single weapon against area targets. In order to get a wide array of results each comparison was done for different user inputted values of MAE_F, REP & DEP and ballistic dispersion in mils, in the normal plane. The weapon impact angle and release slant range were kept at a constant 45° and 10,000 ft respectively for each comparison. The size of the area target used for all comparisons was a 50 ft by 50 ft area.

1. JMEM versus Carleton Damage Function

The first comparison done was the JMEM method results for computing damage probabilities of single weapons versus area targets against a Monte Carlo simulation modeling a weapon with the a lethal area matrix populated with the Carleton damage function. We expect these methods to give different results since the JMEM approximates the fractional coverage and the effectiveness of the weapon against the target using equations (4.8) through (4.22) while the Monte Carlo simulation uses a lethal area matrix from the Carleton damage function. Tables 4.1 and 4.2 present the results for the P_K found for both the JMEM method and Monte Carlo simulation. The shaded values in the Tables are obtained from the JMEM method, while the unshaded values are those of the Monte Carlo simulation.

$MAE_F=5000ft^2$	Ballistic Dispersion (mils in normal plane)		
	0	3	5
REP=25ft & DEP=50ft	0.2079	0.1694	0.131
	0.2285	0.1521	0.1007
REP=25ft & DEP=25ft	0.3089	0.2445	0.181
	0.3596	0.2226	0.135
REP=50ft & DEP=25ft	0.1743	0.1551	0.1288
	0.1987	0.152	0.1066
REP=50ft & DEP=50ft	0.1168	0.1073	0.0929
	0.1263	0.1039	0.0794

Table 4.1: Results of JMEM and Monte Carlo for $MAE_F=5000ft^2$

$MAE_F=1000ft^2$	Ballistic Dispersion (mils in normal plane)		
	0	3	5
REP=25ft & DEP=50ft	0.0499	0.0394	0.0297
	0.0487	0.0321	0.021
REP=25ft & DEP=25ft	0.0877	0.0655	0.0454
	0.0881	0.0511	0.0295
REP=50ft & DEP=25ft	0.0471	0.04	0.0317
	0.0487	0.0349	0.0232
REP=50ft & DEP=50ft	0.0265	0.0242	0.0207
	0.0269	0.0219	0.0166

Table 4.2: Results of JMEM and Monte Carlo for $MAE_F=1000ft^2$

It can be seen in Tables 4.1 and 4.2 that the probability of damage results for the two methodologies does not match. The lethal area as described by the MAE_F does not seem to alter the error between the JMEM and Monte Carlo result, which is expected since both methodologies use the MAE_F to find the weapon radii used to describe the lethality of the weapon. We can imply then

that the delivery accuracy of the weapon is the root of the error. This is especially true for cases involving weapon ballistic dispersion. This was also case for the single weapon versus unitary target cases. From this, we came make one of two assumptions. One possibility is that the Monte Carlo simulation does not correctly represent the ballistic dispersion by varying it by a random number in each iteration of the code. Another reason may be that the root sum squaring of the weapon ballistic dispersion with the aiming error as done in the JMEM method may not be a realistic approximation of the weapon target interaction.

Unlike the comparison between the JMEM and Monte Carlo simulation methods performed for single weapons against unitary targets, from Table 4.1 and 4.2 we see an appreciable difference in the results for cases of zero ballistic dispersion. This error could be due to the fact that the JMEM method approximates the fractional coverage of the weapon on the target in terms of Figure 4.3. This differs from the Monte Carlo simulation, in that the Monte Carlo simulation computes the fractional coverage directly from the ground plane, where no approximations are used.

It is also interesting to note that the Monte Carlo result is constantly greater that the JMEM result for non-zero ballistic dispersion cases, which shows that the JMEM is a more conservative estimate of damage. However, the results for zero ballistic dispersion cases are generally more conservative in the Monte Carlo simulation.

2. JMEM versus Rectangular Cookie Cutter Approximation

The last comparison for single weapons against area targets is between the JMEM method and a Monte Carlo simulation which represents the weapon as a rectangular cookie cutter. This is a particularly interesting comparison because both methods represent the weapon lethality in terms of a rectangular lethal area. The Monte Carlo simulation always represents the weapon lethal area as a rectangular cookie cutter approximation of the Carleton damage

function, which is based off of the user inputted values of weapon lethality. Again, this rectangular cookie cutter has dimensions of L_{ET} and W_{ET}. The JMEM method, however, uses a rectangular cookie cutter dimensions based on the sizes of the target and L_{ET} and W_{ET} as shown in equations (4.2) and (4.3). Table 4.3 and 4.4 present the results for the P_K found for both the JMEM method and Monte Carlo simulation. The shaded values in the Tables are obtained from the JMEM method, while the unshaded values are those of the Monte Carlo simulation.

$MAE_F=5000ft^2$	Ballistic Dispersion (mils in normal plane)		
	0	3	5
REP=25ft & DEP=50ft	0.2253	0.181	0.1381
	0.2285	0.1521	0.1007
REP=25ft & DEP=25ft	0.3564	0.2756	0.1994
	0.3596	0.2226	0.135
REP=50ft & DEP=25ft	0.1958	0.1718	0.1405
	0.1987	0.152	0.1066
REP=50ft & DEP=50ft	0.1237	0.1127	0.0969
	0.1263	0.1039	0.0794

Table 4.3: Results of JMEM and Monte Carlo for $MAE_F=5000ft^2$

$MAE_F=1000ft^2$	Ballistic Dispersion (mils in normal plane)		
	0	3	5
REP=25ft & DEP=50ft	0.0486	0.0381	0.0286
	0.0487	0.0321	0.021
REP=25ft & DEP=25ft	0.0879	0.0647	0.0445
	0.0881	0.0511	0.0295
REP=50ft & DEP=25ft	0.0466	0.0394	0.0309
	0.0487	0.0349	0.0232
REP=50ft & DEP=50ft	0.0258	0.0233	0.02
	0.0269	0.0219	0.0166

Table 4.4: Results of JMEM and Monte Carlo for $MAE_F=1000ft^2$

The trends shown in Tables 4.3 and 4.4 are very similar to those presented for the other Monte Carlo versus JMEM comparisons. The most recognizable trend is the fact that error between the JMEM and Monte Carlo simulation is greatest for cases involving high weapon ballistic dispersion. For this comparison less error is present for cases with zero ballistic dispersion than there was for the comparison using the Monte Carlo simulation representing the weapon with the Carleton damage function. A viable explanation for this is that the JMEM and Monte Carlo using the rectangular cookie cutter model the weapon lethal area more in a similar fashion. This is not the case that was presented earlier in this chapter, because the Monte Carlo simulation representing the weapon with the Carleton damage function does not use an approximation of the Carleton damage function in calculating the weapon lethal area matrix.

Again, a prevalent similarity between all the comparisons presented is that the Monte Carlo simulation consistently gives greater damage results than the JMEM method, leading to the conclusion that the JMEM is more conservative than the Monte Carlo for all delivery accuracy inputs.

THIS PAGE INTENTIONALLY LEFT BLANK

V. STICKS OF WEAPONS VERSUS AREA TARGETS

There are times when an aircraft will drop several weapons against an area target, rather than just a single weapon. By dropping several weapons during one aircraft sortie we increase the damaged area in the ground plane. At times this can increase the probability of damaging the target area depending on the size and delivery accuracy of the stick.

When a stick of weapons is released against an area target, the pattern of weapon impacts on the ground determines the effectiveness of the stick. Figure 5.1 shows an example of the pattern of impact points for a stick of four weapons. Each impact point is modeled with a small explosive picture. The dotted rectangles represent the lethal area of each weapon. By knowing the location of each impact point and the size of the lethal area for each weapon, we define dimensions of the stick, and impact point pattern. We define the stick length and width, L_S and W_S, as the dimensions of the smallest rectangle in the ground plane that encloses each impact point of the weapons in the stick. These impact points are also the centers of each respective weapon's lethal area. The pattern length and width, L_P and W_P, are defined as the dimensions of the smallest rectangle that encloses all of the weapons' lethal areas.

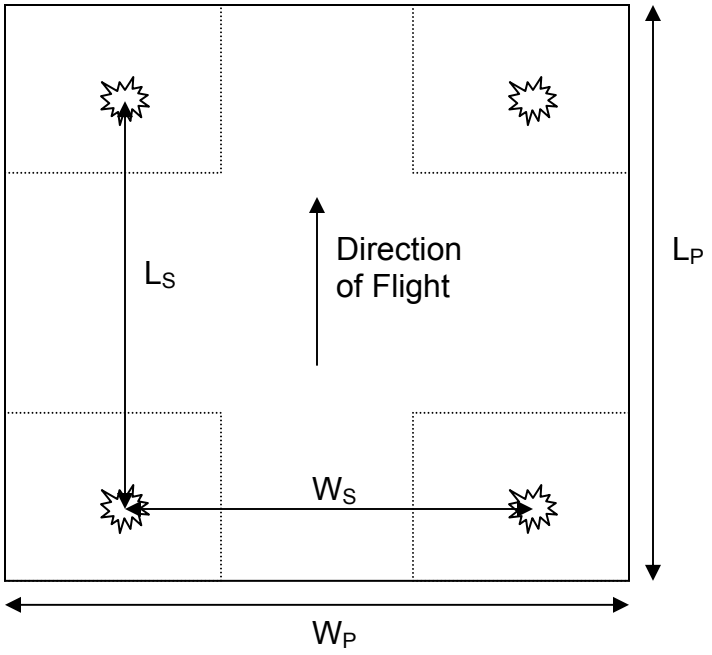

Figure 5.1: Definition of stick and pattern dimensions.

The location of the impact points on the ground plane is highly dependent on each of the weapons' release conditions. In the range direction, the impact pattern is determined by the intervalometer setting of the aircraft. The intervalometer is a timer in the aircraft that sends electrical pulses to the mechanism that releases the weapons from the aircraft. For example, an intervalometer setting of .5 seconds, which means, one or more weapons will be released every .5 seconds. For a given stick delivery we define the number of intervalometer pulses sent, or n_r, and the number of weapons released for each pulse, or n_p. In the deflection direction, the stick pattern is also affected by the release conditions. This dependence stems from the location of the weapon on the aircraft. We would expect a weapon released from under the aircraft fuselage to land in a different position than one released from the tip of the aircraft's wing. In this thesis, we will not study the effects of different weapon

release positions on the aircraft. We instead assume that half of the weapons fall from the same position one wing, and the other half on the same position on the opposite wing.

Another determining factor of the impact points is the delivery accuracy of the stick. We define the delivery accuracy differently for a stick of weapons than we do a single weapon.

In this thesis we will compare the probability of damaging an area target of dimensions 50 ft by 50 ft using different methodologies. As before, we will compare the JMEM method of computing the probability of damaging an area target to two Monte Carlo simulations; the first determining damage using the Carleton damage function, and the other using a rectangular cookie cutter approximation of the Carleton damage function to compute damage.

A. OLD JMEM METHOD

Recently the JMEM changed its methodologies to determine the effectiveness of sticks of weapons against area targets. The old JMEM method for sticks takes heavily into account the possible overlap situations between the weapon lethal areas of the stick weapons. The following explains this method in detail.

The old JMEM method first calculates the effectiveness of the individual weapons in the stick. For the calculations in this thesis, we assume that each of the weapons in the stick is identical. From the user inputted value of MAE_F, impact angle I, and the slant range SR, we find the effective target length and width (L_{ET} and W_{ET}). This is shown in equations (5.1), (5.2), and (5.3).

$$a = \frac{WR_r}{WR_d} = MAX[1 - 0.8\cos(I), 0.3] \tag{5.1}$$

$$L_{ET} = \sqrt{MAE_F \times a} \tag{5.2}$$

$$W_{ET} = L_{ET} / a \tag{5.3}$$

The next step is to determine the effect of ballistic dispersion on the stick effectiveness. We apply the ballistic dispersion to each weapon by enlarging the area of effectiveness. This is done because we cannot combine the ballistic dispersion error with the aiming error when dealing with sticks. Ballistic dispersion in sticks affects each weapon in the stick, rather than the stick as a whole. A detailed explanation of the delivery accuracy of sticks is presented in Section C of this chapter. The new dimensions of the effective area are called L_B and $W_{B.}$ To calculate L_B and $W_{B,}$ we first need to find the ballistic dispersion as a standard deviation in feet in the range and deflection directions in the ground plane, or x_{br} and x_{bd}. These values are calculated using equations (5.4) and (5.5).

$$x_{br} = \frac{SR \times \sigma_b}{1000 \sin(I)} \tag{5.4}$$

$$x_{bd} = \frac{SR \times \sigma_b}{1000} \tag{5.5}$$

One thing to note about these values is that they are similar to the x_{br} and x_{bd} calculated for the single weapon cases, but they are without the factor of 0.6745. For sticks, we do not use the ballistic dispersion as an error probable because we do not root sum square the values with the aiming error probables. Therefore, we keep the ballistic dispersion in terms of standard deviations. Once we have x_{br} and x_{bd} we can find L_B and W_B through equations (5.6) and (5.7).

$$L_B = \sqrt{{L_{ET}}^2 + 8{x_{br}}^2} \tag{5.6}$$

$$W_B = \sqrt{{W_{ET}}^2 + 8{x_{bd}}^2} \tag{5.7}$$

With L_B and W_B, along with the user inputted values of stick length (L_S) and stick width (W_S), we can define the pattern length (L_P) and width (W_P). This is shown in equations (5.8) and (5.9).

$$L_P = L_S + L_B \tag{5.8}$$

$$W_P = W_S + W_B \tag{5.9}$$

Since we have increased the effective area of the weapon with the ballistic dispersion, we need to change the conditional probability of damage inside the rectangle L_B by W_B in order to preserve the lethality of the weapon. The new conditional probability of damage, P_{CD1}, is a fraction of the original probability of damage, $P_{CD.}$ This is found through equation (5.10).

$$P_{CD1} = P_{CD} \frac{L_{ET} W_{ET}}{L_B W_B} \tag{5.10}$$

A graphical representation of how the old JMEM method enlarges the area of effectiveness of each weapon is shown in Figure 5.2.

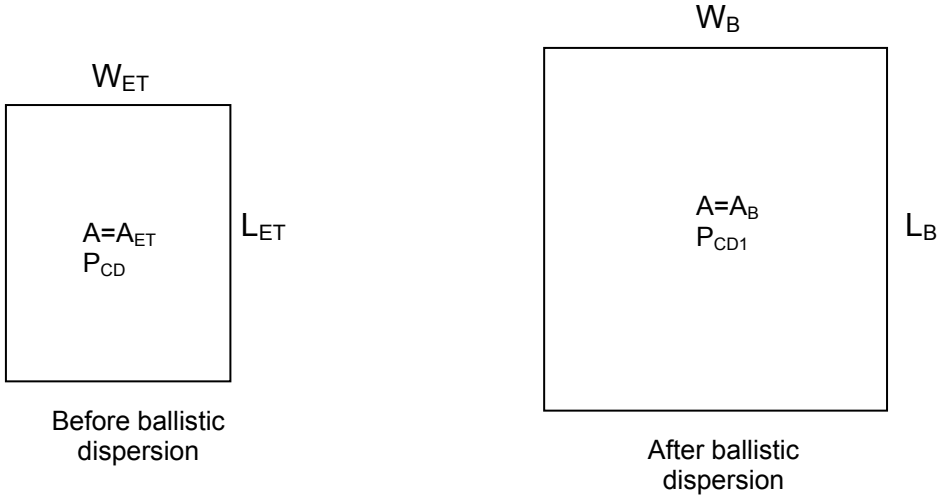

Figure 5.2: Modification of weapon A_{ET} due to ballistic dispersion. (From Ref. [1])

The next step in the old JMEM method for sticks is to account for different overlap scenarios of the stick weapons' effective areas. For the overlap in the deflection direction, we define a degree of overlap in the deflection direction, or n_{od}, which is calculated from the stick width (W_S), pattern width (W_P), and the number of weapons released per intervalometer pulse (n_p). This is shown in equation (5.11).

$$n_{od} = \frac{n_p W_B}{W_P} \tag{5.11}$$

In Figure 5.3 we show how weapons may overlap (left) or may not overlap (right) in the deflection direction.

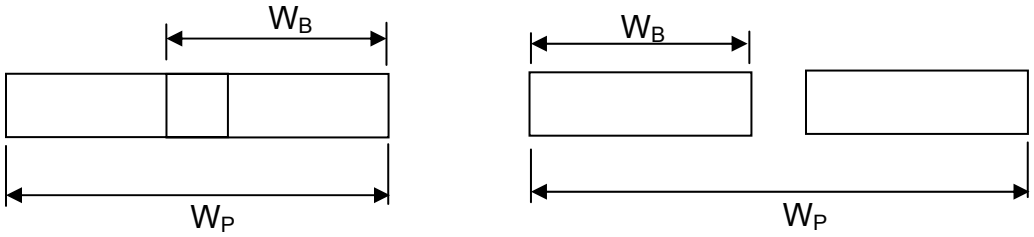

Figure 5.3: Weapons overlapping (left) and not overlapping (right) in the deflection direction. (From Ref. [1])

This n_{od} term can also be defined as what percentage of the n_p released weapons can fit into the stick pattern width. We can tell a great deal from the degree of overlap. If n_{od} >1, we know there is overlap between the stick weapons in the deflection direction. If this is the case, we find the conditional probability of damage in the deflection direction, or $P_{CD/d}$, by using the survival rule and powering up the single weapon conditional damage P_{CD1} by the degree of overlap. This is shown in equation (5.12).

$$P_{CD/d} = 1 - (1 - P_{CD1})^{n_{od}} \tag{5.12}$$

However, if n_{od} <1, we have no overlap in the deflection direction. In this case $P_{CD/d}$ is found by equation (5.13).

$$P_{CD/d} = n_p P_{CD1} \frac{W_B}{W_P} \tag{5.13}$$

This analysis in the deflection direction results in a rectangle that combines weapons in the deflection direction that has a damage function equal to $P_{CD/d.}$ This rectangle is shown in Figure 5.4.

Figure 5.4: Combination of weapons in deflection direction. (From Ref. [1])

A similar analysis is performed for overlap in the range direction. Rather than using n_p to determine the conditional probability, we use the number of intervalometer pulses, or n_r. We define the degree of overlap in the range direction in equation (5.14).

$$n_{or} = \frac{n_r L_B}{L_P} \tag{5.14}$$

In Figure 5.5 we show how weapons may overlap (left) or may not overlap (right) in the deflection direction.

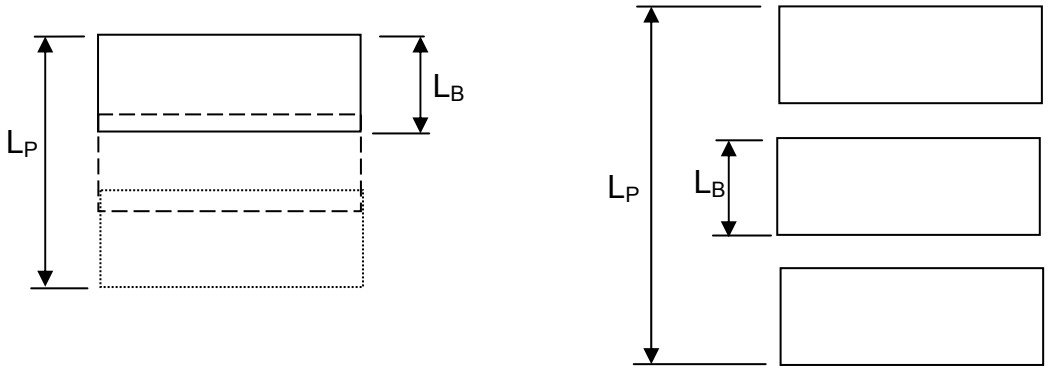

Figure 5.5: Weapons overlapping (left) and not overlapping (right) in the range direction. (From Ref. [1])

If n_{or} >1, we know there is overlap between the stick weapons in the range direction. For this is the case, we find the conditional probability of damage for the whole pattern, or P_{CDS}, by powering up $P_{CD/d}$ by the degree of overlap in the range direction. This is shown in equation (5.15).

$$P_{CDS} = 1 - (1 - P_{CD/d})^{n_{or}} \tag{5.15}$$

However, if n_{or} <1, we have no overlap in the range direction, and P_{CDS} is found by equation (5.16).

$$P_{CDS} = n_r P_{CD/d} \frac{L_B}{L_P} \tag{5.16}$$

Now we have taken the stick of weapons and represented the stick as a single effectiveness area rectangle with dimensions L_P and W_P, with a conditional probability of damage of P_{CDS} as shown in Figure 5.6.

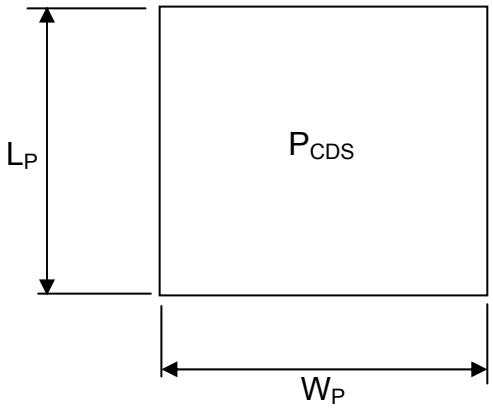

Figure 5.6: Stick pattern dimension and damage function. (From Ref. [1])

The pattern shown in Figure 5.6 is now considered a single effectiveness area. The old JMEM treats this case similarly to the single weapon versus area targets discussed in Chapter IV. The fractional coverage analysis is identical, but the effective pattern length and width (L_{EP} and W_{EP}) is found by taking the maximum of the stick pattern dimensions and the target size. The target size has dimensions of L_a in the range direction, and W_a in the deflection direction. The effective pattern dimensions are calculated with equations (5.17) and (5.18).

$$L_{EP} = MAX(L_P, L_a) \tag{5.17}$$

$$W_{EP} = MAX(W_P, W_a) \tag{5.18}$$

The following is a summary of the equations used to find the expected fractional coverage of a single effectiveness area on a target area. For a detailed explanation of the equations used to find fractional coverage and fractional damage, please refer to Chapter IV Section A.

Range Direction

$$s = \frac{L_{EP} + L_A}{2} \tag{5.19}$$

$$t = \frac{L_{EP} - L_A}{2} \tag{5.20}$$

$$E(F_R) = \{I_1 + I_2 + I_3 + I_4 - I_5\} \tag{5.21}$$

$$I_1 = [normcdf(t) - normcdf(-t)] \tag{5.22}$$

$$I_2 = \frac{L_{EP} + L_A}{2L_A}[normcdf(-t) - normcdf(-s)] \tag{5.23}$$

$$I_3 = \frac{L_{EP} + L_A}{2L_A}[normcdf(s) - normcdf(t)] \tag{5.24}$$

$$I_4 - I_5 = \frac{2\sigma_x}{L_A\sqrt{2\pi}}[e^{-\left(\frac{s}{\sigma_x\sqrt{2}}\right)^2} - e^{-\left(\frac{t}{\sigma_x\sqrt{2}}\right)^2}] \tag{5.25}$$

Deflection Direction

$$s = \frac{W_{EP} + W_A}{2} \tag{5.26}$$

$$t = \frac{W_{EP} - W_A}{2} \tag{5.27}$$

$$E(F_D) = \{I_1 + I_2 + I_3 + I_4 - I_5\} \tag{5.28}$$

$$I_1 = [normcdf(t) - normcdf(-t)] \tag{5.29}$$

$$I_2 = \frac{W_{EP} + W_A}{2W_A}[normcdf(-t) - normcdf(-s)] \tag{5.30}$$

$$I_3 = \frac{W_{EP} + W_A}{2W_A}[normcdf(s) - normcdf(t)] \tag{5.31}$$

$$I_4 - I_5 = \frac{2\sigma_y}{W_A\sqrt{2\pi}}[e^{-\left(\frac{s}{\sigma_y\sqrt{2}}\right)^2} - e^{-\left(\frac{t}{\sigma_y\sqrt{2}}\right)^2}] \tag{5.32}$$

Combined Range and Deflection

$$E(F_C) = E(F_R) \times E(F_D) \tag{5.33}$$

After finding the expected fractional coverage, or $E(F_C)$, we can combined what we've calculated, as well as our knowledge of the weapon reliability (R), to find the expected fractional damage (EFD) of the stick against an area target. This is shown in equation (5.34).

$$EFD = E(F_C) \times \left[\frac{L_P W_P}{L_{EP} W_{EP}} \right] \times R \times P_{CDS} \tag{5.34}$$

B. CURRENT JMEM METHOD

The current JMEM method for computing the effectiveness of sticks of weapons is similar to that of the JMEM method it replaced. The difference between the methods is the handling of the possible overlap conditions of the lethal areas of each stick weapon.

Computing the lethal area of each stick weapon in the JMEM method begins with the handling of ballistic dispersion effects on the individual weapons of the stick. This is done by enlarging the effective lethal area of each weapon to account for ballistic dispersion. The enlarged dimensions are defined as L_B and W_B, and are calculated with same formulas that were used for the old JMEM method. The formulas are restated in equations (5.35) and (5.36).

$$L_B = \sqrt{{L_{ET}}^2 + 8{x_{br}}^2} \tag{5.35}$$

$$W_B = \sqrt{{W_{ET}}^2 + 8{x_{bd}}^2} \tag{5.36}$$

The updated JMEM method then calculates the possible overlap conditions for the lethal enlarged effective areas of the weapons in the stick upon each other. An example of the possible overlap conditions of the lethal weapon areas is shown in Figure 5.7.

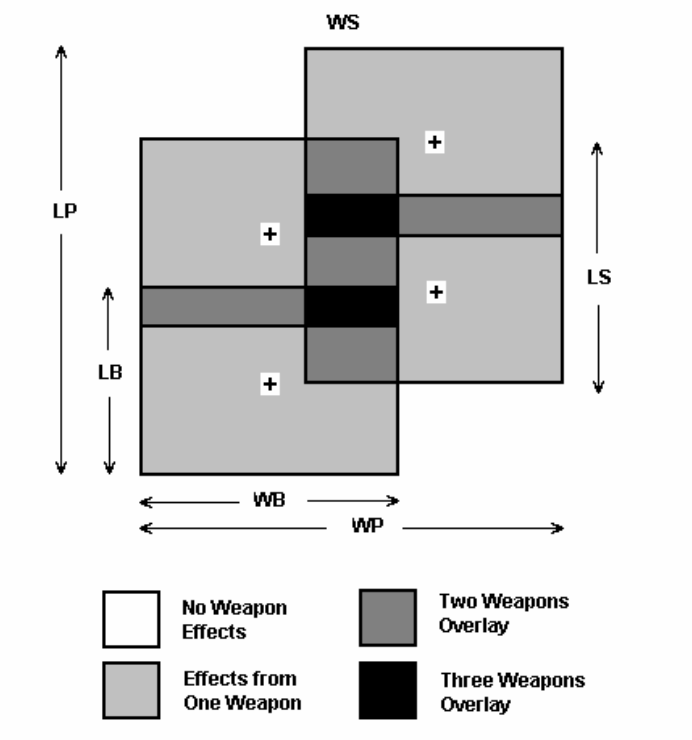

Figure 5.7: Example of weapon lethal areas overlapping in stick. (From Ref. [2])

The JMEM then calculates the conditional probability of damage in each area shown in Figure 5.7. To find the conditional probability of damage for each of the overlapping areas we implement the survival rule, which is given in equation (5.37). This equation gives the P_{CD} of the overlap area, based on the conditional probability of damage for each weapon lethal area ($P_{CD/weapon}$) and the number of weapons that overlap the specific area.

$$P_{CD} = 1 - (1 - P_{CD/weapon})^n \tag{5.37}$$

Once these different damage probabilities for each overlap area have been calculated, the JMEM performs a fractional coverage analysis of the stick against the area target.

The process of calculating the conditional probability of damage for each overlap area, as well as the detailed fractional coverage analysis is explain in great detail in Reference 2.

C. MONTE CARLO SIMULATIONS

For the case of a stick of weapons versus area targets, we run two Monte Carlo simulations to determine the probability of damaging the target. The first Monte Carlo simulation uses the Carleton damage function to determine the damage caused to the area target. The second Monte Carlo simulation uses a rectangular cookie cutter approximation of the Carleton damage function to find the damage to an area target.

As before with single weapons, delivery accuracy plays a large role in determining the probability of damaging a target. In the case of sticks, we cannot directly combine the aiming error and ballistic dispersion error. For sticks, the aiming error affects the accuracy of the stick of weapons as a whole. The ballistic dispersion, on the other hand, affects the impact point of each individual weapon in the stick. Figure 5.8 illustrates two situations where a stick of 4 weapons is aimed at the aim point in the center. The aiming accuracy of the each stick is different, so they do not land in the same location. As can be seen in Figure 5.8, the stick and pattern dimensions no not change, only the location of the stick.

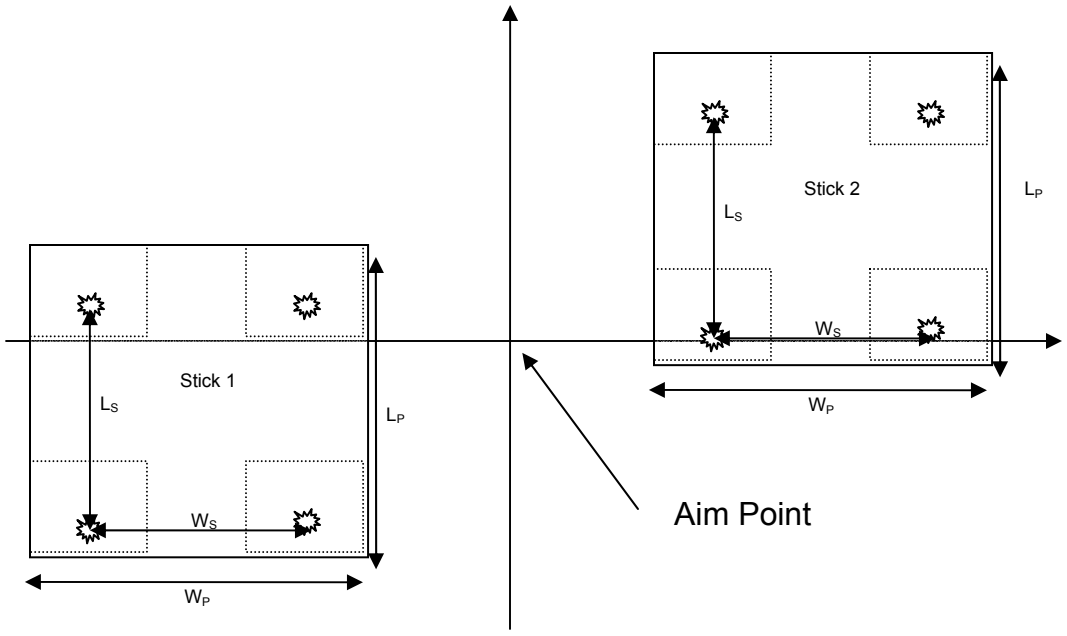

Figure 5.8: Effects of aiming error on a stick of weapons.

Where the aiming error affects the location of the entire stick, ballistic dispersion errors change the pattern length (L_P) and width (W_P) of the stick. As described before, these dimensions create the smallest rectangle that encloses all of the lethal areas of each weapon. The ballistic dispersion changes the pattern dimensions by affecting the individual impact point of each weapon. This is shown in Figure 5.9.

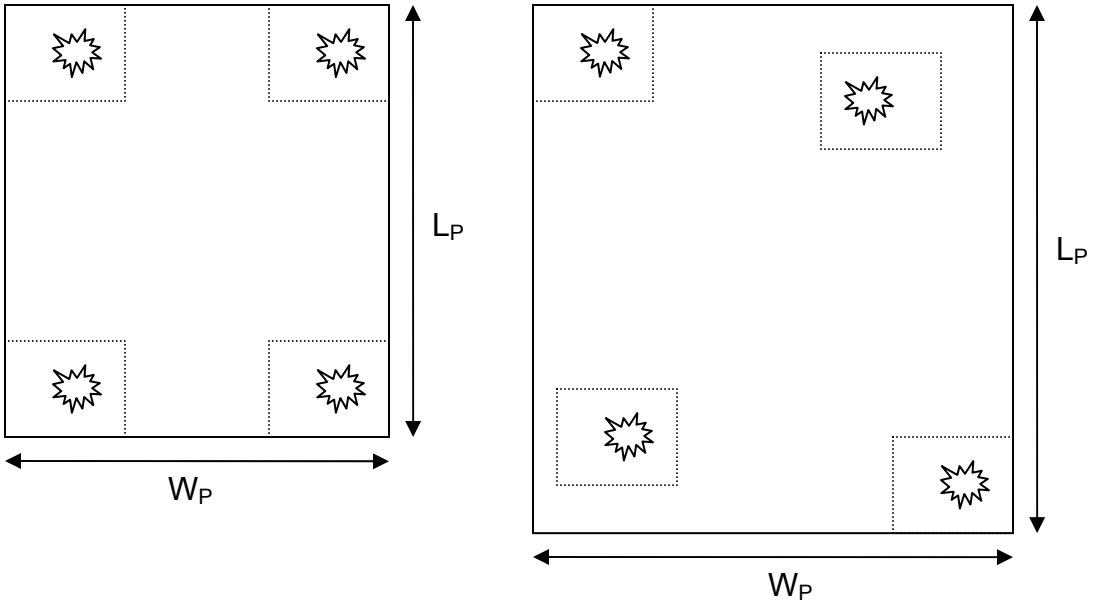

Figure 5.9: Effects of ballistic dispersion error on a stick of weapons.

We can see in Figure 5.9 how the ballistic dispersion affects the impact point of each weapon, thus changing the pattern dimensions of the stick.

The Monte Carlo simulations run in this thesis are for a stick of four weapons that has a stick length of 100 ft and a stick width of 100 ft. Figure 5.10 gives a visual representation of this by placing a Cartesian coordinate system on the ground, and centering the area target about the origin (shown as a dashed rectangle). With no aiming error or ballistic dispersion present in the problem, the stick would land as shown in Figure 5.10. We can see that the weapons are evenly spaced at a distance of 100 ft in the stick.

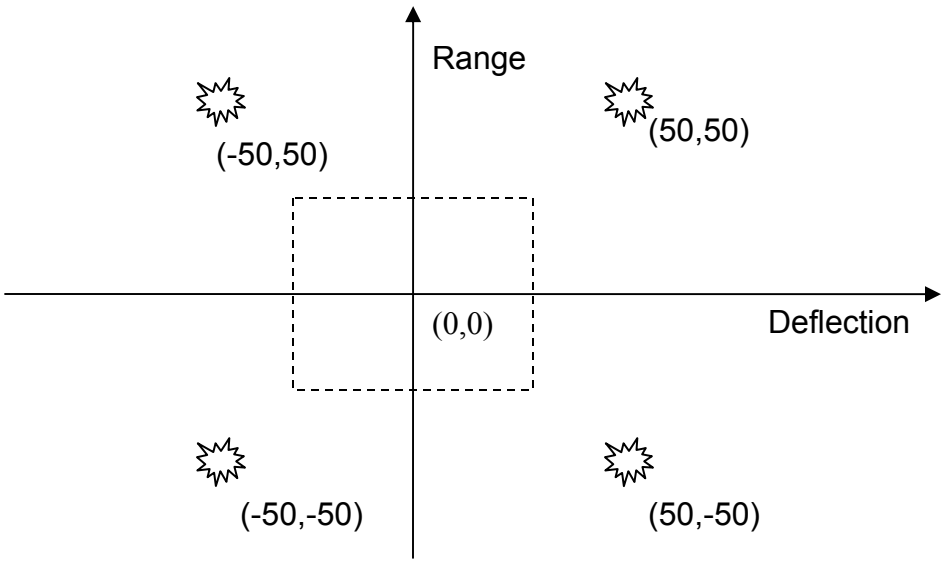

Figure 5.10: Monte Carlo stick simulations scenario.

The first step in the Monte Carlo simulations is to find the standard deviation of the aiming error and ballistic dispersion in the ground plane. This is calculated from the user inputted values of the range error probable (REP), deflection (DEP), slant range (SR), and impact angle (I). This is shown in equations (5.38), (5.39), (5.40), and (5.41).

$$\sigma_{aim\_range} = \frac{REP}{0.6745} \tag{5.38}$$

$$\sigma_{aim\_deflection} = \frac{DEP}{0.6745} \tag{5.39}$$

$$x_{br} = \frac{SR \times \sigma_b}{1000 \sin(I)} \tag{5.40}$$

$$x_{bd} = \frac{SR \times \sigma_b}{1000} \tag{5.41}$$

All four of the delivery accuracy standard deviations (σ_{aim_range}, $\sigma_{aim\_deflection}$, $x_{br,}$ and x_{bd}) are then multiplied by a random number from a normal distribution. This normal distribution has a mean value of zero and a standard deviation of one. Using this normal distribution allows us to assume that the average probability of damage for a large number of stick trials will equal the expected value of the damage function at the desired aim point.

With the ballistic dispersion error affecting each weapon individually, and the aiming error affecting the stick as a whole, we are able to find the location of each individual weapon impact point. At each impact point, we represent the weapon either by a lethal area matrix filled by the Carleton damage function or a rectangular cookie cutter, depending on the simulation.

The final step of the Monte Carlo simulations is to determine the probability of damage to the target from each of the four stick weapons. This is accomplished by dividing the 50 ft by 50 ft target into 1ft by 1ft cells, and using the survival rule to find the probability of damage in each target cell. Figure 5.11 shows an example of how we combine the probabilities of damage from each of the four weapons in a stick on a specific cell in the target area. The probability of damage contributions from the four weapons are defined as Pd_1, Pd_2, Pd_3, and $Pd_{4.}$

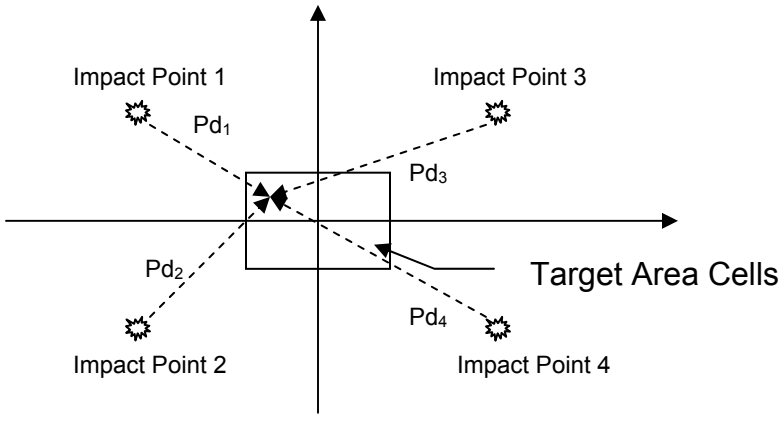

Figure 5.11: Damage contributions of stick weapons.

To calculate the probability of damage due to the contributions from each of the four weapons we use the survival rule. The survival rule is a formula that calculates the probability of damaging a target given multiple damage contributions from the four weapons. This is shown in equation (5.42).

$$Pd_{cell} = 1 - (1 - Pd_1) * (1 - Pd_2) * (1 - Pd_3) * (1 - Pd_4) \tag{5.42}$$

We repeat this survival rule on each of the j number of cells in the target matrix. To find the overall probability of damage for a given Monte Carlo iteration i, we sum all of the $P_{d/cell}$, and divide it by the total number of cells j. This is shown in equation (5.43).

$$P_{d/i} = \frac{\sum_{j=1}^{j} P_{d/cell(j)}}{j} \tag{5.43}$$

For the full Monte Carlo simulation, we repeat this for n iterations, each individual iteration (i) resulting in its own probability of damage $P_{d/i}$. The value n is the total number of iterations used in the Monte Carlo simulation. The final probability of damage for the Monte Carlo simulation is the average of all of the $P_{d/I}$ shown in equation (5.44).

$$P_d = \frac{\sum_{i=1}^{n} P_{d/i}}{n} \tag{5.44}$$

1. Modeling Weapons with Carleton Damage Function

The first Monte Carlo simulation utilizes the Carleton damage function to determine the amount of damage done to a target element given the distance from the element to the weapon impact points in a stick of weapons. In this simulation, we will assume that each of the weapons in the stick is identical in their effectiveness. We determine this weapon effectiveness in terms of the weapon radii WR_r and WR_d. The weapon radii are found from the user inputted values of MAE_F, impact angle I, and the slant range SR. This is shown in equations (5.45), (5.46), and (5.47).

$$a = \frac{WR_r}{WR_d} = MAX\left[1 - 0.8\cos(I), 0.3\right] \tag{5.45}$$

$$WR_r = \sqrt{MAE_F \times \frac{a}{\pi}} \tag{5.46}$$

$$WR_d = \frac{WR_r}{a} \tag{5.47}$$

These values of weapon radii stay constant through each iteration of the Monte Carlo simulation. The values that change with each iteration will be the individual weapon impact points, as discussed before.

Unlike the single weapon cases against unitary and area targets, there is not a lethal area matrix used in the Monte Carlo simulation for sticks using the Carleton damage function. Instead, we determine the location of each weapon impact point in the stick and determine the distance from the impact points to the target area. To do this, we divide the target area into 1ft by 1ft cells, and find the distance from each these cells to the impact points. The Carleton damage function is given in equation (5.48).

$$P_d(x, y) = \exp - \left[\frac{x^2}{WR_r^{\,2}} + \frac{y^2}{WR_d^{\,2}} \right] \tag{5.48}$$

We have already calculated the weapon radii, and we define the x and y terms in equation (5.48) as the distance in range and deflection respectively from a weapon impact point to a given cell in the area target. For a given cell in the area target, we will get a Pd from the Carleton damage function, for each of the four weapons in the stick. For example, let us say have a scenario where the weapon radii are both 10 ft (WR_r =10 and WR_d = 10). We will also say we are looking at a target cell 3 ft from the impact point of the first weapon (in a stick of four) in the range direction, and 5 ft in the deflection direction. We can find the probability of damage due to this weapon on that target cell as shown in equation (5.49).

$$Pd_1 = \exp - \left[\frac{3^2}{10^2} + \frac{5^2}{10^2} \right] = 0.7118 \qquad (5.49)$$

This value of 0.7118 will be called Pd_1, since it is the probability of damage caused by the first weapon in the stick. We will assume the other three weapons in the stick cause probabilities of damage equal to 0.5312, 0.3041, and 0.7590 respectively. Now that we have Pd_1, Pd_2, Pd_3, and Pd_4, we use the survival rule explain in equatrion (5.42)to find the overall probability of damage caused by the example stick. The overall Pd of the cell for this example is shown in equation (5.50).

$$Pd_{cell} = 1 - (1 - 0.7118) * (1 - 0.5312) * (1 - 0.3041) * (1 - 0.7590) = 0.9773 \qquad (5.50)$$

The overall Pd of the target cell is 0.9773. This process is then repeated for all of the cells in the target area. To finish the iteration, the Pd's of each of the area target cells are averaged to find a total probability of damaging the area target.

For the full Monte Carlo simulation, we would repeat this process for n iterations, each individual iteration (i) resulting in its own probability of damage $P_{d/i}$. The value n is the total number of iterations used in the Monte Carlo simulation. The final probability of damage for the Monte Carlo simulation is the average of all of the $P_{d/I}$ shown in equation (5.51).

$$P_d = \frac{\sum_{i=1}^{n} P_{d/i}}{n} \tag{5.51}$$

The number of iterations (n) used in the Monte Carlo simulation used to compare the Monte Carlo and JMEM methods for sticks was set to one hundred thousand. In running the Monte Carlo simulation multiple times, we ensure that the resulting P_d from each simulation did not change. This shows that the simulation was converging to a single value, which we would expect. After testing different values of n, it was found that one hundred thousand provides for this convergence.

A Monte Carlo simulation modeling a stick of weapons using with the Carleton damage function is presented in Appendix B, Option 1.

2. Modeling Weapons with Rectangular Cookie Cutter Approximation

The second Monte Carlo simulation utilizes a rectangular cookie cutter approximation of the Carleton damage function to determine the amount of damage done to a target element given the distance from the element to the weapon impact points in a stick of weapons. We again assume that each weapon in the stick has identical lethality. We define this lethality as we have before with equations (5.52), (5.53), and (5.54).

$$a = \frac{WR_r}{WR_d} = MAX\left[1 - 0.8\cos(I), 0.3\right] \tag{5.52}$$

$$L_{ET} = \sqrt{MAE_F \times a} \tag{5.53}$$

$$W_{ET} = L_{ET} / a \tag{5.54}$$

The calculation of the probability of damage against the area target is very similar to the Monte Carlo simulation using the Carleton damage function. We first determine the location of each weapon impact point in the stick and determine the distance from the impact points to the target area. To do this, we divide the target area into 1ft by 1ft cells, and find the distance from each these cells to the impact points. At each of the weapon impact points, a rectangular cookie cutter with dimensions L_{ET} and W_{ET} is placed to represent the weapon.

An example iteration of a Monte Carlo simulation using rectangular cookie cutter representations of the weapons is shown in Figure 5.12. In this example we have four weapons, each represented by rectangular cookie cutters of dimensions L_{ET} and W_{ET}= 3ft. The stick of weapons dropped against a 3ft by 3ft area target which is shaded in light grey. Due to the assumed delivery accuracy errors, the stick has not fallen centered upon the area target.

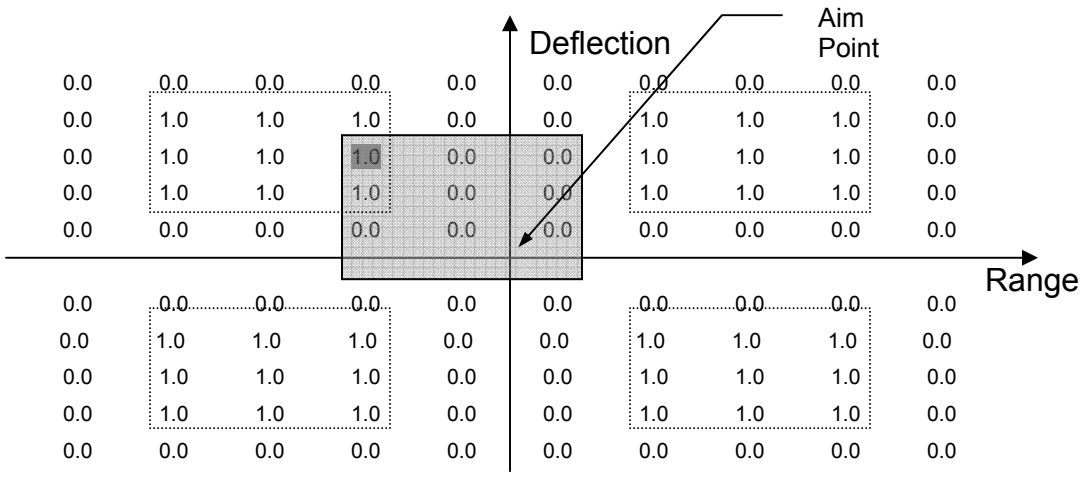

Figure 5.12: Example iteration of Monte Carlo stick simulation using rectangular cookie cutter lethal area matrices against an area target.

Unlike the Monte Carlo simulation using the Carleton damage function, for the rectangular cookie cutter Monte Carlo simulation we do not need to use the survival rule to find the probability of damage caused by the stick. Since each weapon is represented by a rectangular cookie cutter, the P_d inside each weapon effective area is equal to one. Therefore, each 1 ft by 1 ft cell of the area target is either completely damaged by the stick (Pd_{cell}= 1) or not at all (Pd_{cell}=0). For the example shown in Figure 5.12 we see that the top left hand cell, which is shaded in dark grey, is completely damaged by the top left hand weapon of the stick. This Pd_{cell} would equal 1 whether or not another weapon of the stick contributes damage to the cell. The probability of damaging the whole area target for this iteration, or $P_{d/i}$, is equal to the average of all the $P_{d/cell}$ of each cell in the target area. For the example iteration above we find that the $P_{d/i}$ is shown in equation (5.55)

$$P_{d/i} = \frac{1.0+0.0+0.0+1.0+0.0+0.0+0.0+0.0+0.0}{9} = 0.2222 \qquad (5.55)$$

For the full Monte Carlo simulation, we would repeat this process for n iterations, each individual iteration (i) resulting in its own probability of damage $P_{d/i}$. The value n is the total number of iterations used in the Monte Carlo simulation. The final probability of damage for the Monte Carlo simulation is the average of all of the $P_{d/I}$ shown in equation (5.56).

$$P_d = \frac{\sum_{i=1}^{n} P_{d/i}}{n} \tag{5.56}$$

The number of iterations (n) used in the Monte Carlo simulation used to compare the Monte Carlo and JMEM methods for sticks was set to one hundred thousand. In running the Monte Carlo simulation multiple times, we ensure that the resulting P_d from each simulation did not change. This shows that the simulation was converging to a single value, which we would expect. After testing different values of n, it was found that one hundred thousand provides for this convergence.

A Monte Carlo simulation with a stick modeled as rectangular cookie cutter approximations of the Carleton damage function is presented in Appendix B, Option 2.

D. RESULTS OF STICK COMPARISONS

The comparisons done this section are to find the differences between weapon effectiveness methodologies for the case of a stick of weapons against area targets. In order to get a wide array of results each comparison was done for different user inputted values of MAE_F, REP & DEP and ballistic dispersion in mils, in the normal plane. The weapon impact angle and release slant range were kept at a constant 45° and 10,000 ft respectively for each comparison. The

size of the area target used for all comparisons was a 50 ft by 50 ft area. Each test case dealt with a stick of four weapons with a stick length and width equal to 100 ft.

1. Updated JMEM Method versus Old JMEM Method

The first comparison completed was the updated JMEM method results for computing damage probabilities of sticks of weapons versus area targets against the old JMEM method. We would expect these methods to produce different results due to the way each method calculates the possible overlap scenarios for weapons in the stick. Tables 5.1 and 5.2 present the results for the P_K found for both the JMEM methods. The shaded values in the Tables are obtained from the new JMEM method, while the unshaded values are those of the old JMEM method.

$MAE_F=5000ft^2$	Ballistic Dispersion (mils in normal plane)		
	0	3	5
REP=25ft & DEP=50ft	0.4933	0.3099	0.2093
	0.3636	0.3422	0.2877
REP=25ft & DEP=25ft	0.589	0.35	0.2241
	0.4334	0.4159	0.3522
REP=50ft & DEP=25ft	0.421	0.3057	0.2144
	0.3888	0.3428	0.2928
REP=50ft & DEP=50ft	0.3526	0.2707	0.2003
	0.3261	0.2813	0.2385

Table 5.1: Results of new JMEM and old JMEM for weapons of $MAE_F=5000ft^2$

MAE_F=1000ft^2	Ballistic Dispersion (mils in normal plane)		
	0	3	5
REP=25ft & DEP=50ft	0.1311	0.0732	0.0475
	0.0773	0.08	0.0674
REP=25ft & DEP=25ft	0.1826	0.0893	0.0527
	0.0863	0.0966	0.0845
REP=50ft & DEP=25ft	0.1208	0.077	0.0502
	0.0797	0.0781	0.0687
REP=50ft & DEP=50ft	0.0867	0.0631	0.0453
	0.0714	0.0647	0.0547

Table 5.2: Results of new JMEM and old JMEM for weapons of MAE_F=1000ft^2

It can be seen from Tables 5.1 and 5.2 that the damage results do not match, as was expected from the different methods. We see that the new JMEM method is generally more conservative in its damage estimates for cases of zero ballistic dispersion than the JMEM method it replaced. For the cases involving ballistic dispersion, the new JMEM gives larger damage estimates. We would assume that the ballistic dispersion is the cause of the differences in damage estimates, but both JMEM methods treat the ballistic dispersion in the same way; by enlarging the effective target area of the weapon. This leaves the difference in modeling the overlap conditions of the weapons in the stick. The older JMEM method defines degrees of overlap of stick weapons in the deflection and range directions. The new JMEM method calculates conditional probabilities of damage in each of the areas where the weapon effectiveness rectangles overlap, as discussed in detail in Reference 2.

2. JMEM versus Carleton Damage Function

The next comparison done was between the JMEM method results for computing damage probabilities of sticks of weapons versus area targets and a Monte Carlo using the Carleton damage function to calculate damage probabilities. The JMEM and Monte Carlo simulation are expected to yield different damage probabilities. This is due to a difference in incorporating weapon ballistic dispersion and the calculation of damage itself. Tables 5.3 and 5.4 present the results for the P_K found for both the JMEM method and Monte Carlo simulation. The shaded values in the Tables are obtained from the JMEM method, while the unshaded values are those of the Monte Carlo simulation.

$MAE_F=5000ft^2$	Ballistic Dispersion (mils in normal plane)		
	0	3	5
REP=25ft & DEP=50ft	0.3443	0.2992	0.241
	0.3636	0.3422	0.2877
REP=25ft & DEP=25ft	0.4097	0.3569	0.2838
	0.4334	0.4159	0.3522
REP=50ft & DEP=25ft	0.3373	0.2966	0.2448
	0.3888	0.3428	0.2928
REP=50ft & DEP=50ft	0.2827	0.249	0.2085
	0.3261	0.2813	0.2385

Table 5.3: Results of JMEM and Monte Carlo for $MAE_F=5000ft^2$

MAE_F=1000ft^2	Ballistic Dispersion (mils in normal plane)		
	0	3	5
REP=25ft & DEP=50ft	0.0803	0.0711	0.0563
	0.0773	0.08	0.0674
REP=25ft & DEP=25ft	0.0971	0.0861	0.0673
	0.0863	0.0966	0.0845
REP=50ft & DEP=25ft	0.0804	0.0704	0.0572
	0.0797	0.0781	0.0687
REP=50ft & DEP=50ft	0.0666	0.0581	0.0479
	0.0714	0.0647	0.0547

Table 5.4: Results of JMEM and Monte Carlo for MAE_F=1000ft^2

It can be seen in Tables 5.3 and 5.4 that the probability of damage results for the two methodologies does not match. This was expected due to the significant difference between the JMEM and Monte Carlo simulation methods. When looking at the results, a majority of the comparisons show that the JMEM method is more conservative than the Monte Carlo simulation. The few times where the JMEM results are greater than the Monte Carlo simulation are cases with a MAE_F = 1000 ft^2. This leads us to believe that the differences in the dealing with the lethality of the stick weapons in the two methods leads to differences in the damage caused by those weapons. This is intuitive because the JMEM method increases each weapons lethal area due to the ballistic dispersion, while the Monte Carlo simulation obtains the damage directly from the Carleton damage function. The Monte Carlo simulation deals with ballistic dispersion by moving the individual weapon impact points of the stick.

The calculation of the damage to each 1 ft by 1 ft cell of the area target is also handled differently in the JMEM method and Monte Carlo simulation. As discussed earlier, the JMEM method takes each impact point of the stick

weapons, and places an effective lethal area on each impact point to represent the weapon lethality. The method then calculates the conditional probability of each lethal area, and each overlap area where the lethal areas of two or more stick weapons have overlapped. The JMEM then uses fractional coverage analysis to find the damage done to the whole area target. The Monte Carlo simulation represents each weapon with a Carleton damage function. The fact that the Monte Carlo simulation uses the Carleton damage function directly leads to assumption that the Monte Carlo is more accurate than the JMEM method, which uses approximations of the Carleton damage function.

3. JMEM versus Rectangular Cookie Cutter Approximation

The last comparison for sticks against area targets is between the JMEM method and a Monte Carlo simulation which represents the stick weapons rectangular cookie cutters. Again, we do not expect the same results for both of these models. We would expect, however, that the Monte Carlo simulation using rectangular cookie cutters to be closer to the JMEM results than the Monte Carlo simulation which used the Carleton damage function directly. This is because both the JMEM and the Monte Carlo simulation using the rectangular cookie cutter make approximations of the Carleton damage function in order to find the probability of damaging the area target. Tables 5.5 and 5.6 present the results for the P_K found for both the JMEM method and Monte Carlo simulation. The shaded values in the Tables are obtained from the JMEM method, while the unshaded values are those of the Monte Carlo simulation.

$MAE_F=5000ft^2$	Ballistic Dispersion (mils in normal plane)		
	0	3	5
REP=25ft & DEP=50ft	0.3754	0.3129	0.2491
	0.3636	0.3422	0.2877
REP=25ft & DEP=25ft	0.4527	0.3754	0.2944
	0.4334	0.4159	0.3522
REP=50ft & DEP=25ft	0.372	0.3148	0.2547
	0.3888	0.3428	0.2928
REP=50ft & DEP=50ft	0.3077	0.2599	0.2152
	0.3261	0.2813	0.2385

Table 5.5: Results of JMEM and Monte Carlo for $MAE_F=5000ft^2$

$MAE_F=1000ft^2$	Ballistic Dispersion (mils in normal plane)		
	0	3	5
REP=25ft & DEP=50ft	0.0797	0.0706	0.0566
	0.0773	0.08	0.0674
REP=25ft & DEP=25ft	0.0962	0.0863	0.0676
	0.0863	0.0966	0.0845
REP=50ft & DEP=25ft	0.0804	0.0713	0.0576
	0.0797	0.0781	0.0687
REP=50ft & DEP=50ft	0.066	0.0582	0.0477
	0.0714	0.0647	0.0547

Table 5.6: Results of JMEM and Monte Carlo for $MAE_F=1000ft^2$

The trends shown in Tables 5.5 and 5.6 are very similar to those presented for the other Monte Carlo versus JMEM comparison for stick weapons. But, as expected, the differences in this comparison are smaller than they were for the previous comparison between the JMEM method and the Monte Carlo simulation using the Carleton damage function. From the results, it seems that since the compared methods shown in Tables 5.5 and 5.6 both use

approximations of the Carleton damage function to represent the stick weapons' lethality, they would give similar results. Since they results are not the same, we can assume that the differences in the results stem from the treatment of ballistic dispersion and aiming errors. As discussed earlier, the JMEM uses the ballistic dispersion to enlarge the effective lethal area of each weapon in the stick. The JMEM method then does a fractional coverage analysis on the weapon lethal areas, as well as any areas where the weapon lethal areas overlap. The Monte Carlo simulations use the delivery accuracy errors to place the actual impact point of each weapon in the stick. The ballistic dispersion error affects each weapon individually, while the aiming error moves the location of the entire stick. We assume that the difference between the damage results of the JMEM and Monte Carlo methods are due to these large differences in methodologies.

THIS PAGE INTENTIONALLY LEFT BLANK

VI. SUMMARY OF RESULTS

This thesis successfully compared and contrasted different methodologies in determining weapon effectiveness for three different attack scenarios. These scenarios were a single weapon dropped against a unitary target, a single weapon against an area target, and a stick of weapons against an area target. For each weapon/target interaction, we tested an array of ballistic dispersion and aiming errors. A summary of the method comparisons is detailed below.

- The first set of comparisons completed was for the case of a single weapon dropped against a unitary target. The three methodologies compared for this weapon/ target interaction were two Monte Carlo simulations against the results obtained from the method utilized by the Joint Munitions Effectiveness Manuals (JMEM). The first Monte Carlo simulation represented the single weapon as a lethal area matrix with probabilities of damage obtained from the Carleton damage function. When comparing the damage results from this Monte Carlo simulation and the JMEM method results, we found that a majority of the comparisons showed that the JMEM method gave a conservative estimate of damage. For cases of zero ballistic dispersion, however, we saw that the Monte Carlo and JMEM produced near the same result for damage against a unitary target. The second Monte Carlo simulation represented the single weapon as a rectangular cookie cutter approximation of the Carleton damage function. The result of the comparison between the rectangular cookie cutter Monte Carlo simulation and the JMEM is similar to that of the previous Monte Carlo simulation in that the JMEM damage result was consistently more conservative than the Monte Carlo method. However, for this comparison, the damage results were not similar for cases of zero ballistic dispersion.

- The next set of comparisons completed was for the case of a single weapon dropped against an area target. As in the previous case for a single weapon against a unitary target, the JMEM method is compared to two Monte Carlo simulations. The Monte Carlo simulation that represented the weapon as a lethal area matrix populated with values from the Carleton damage function, when compared to the JMEM method, resulted in higher damage values for all non-zero values of ballistic dispersion. The zero ballistic dispersion error cases showed that the Monte Carlo simulation results were generally more conservative than those of the JMEM. Unlike the previous comparison for unitary targets, the JMEM and Monte Carlo simulation did not produce similar results. The same observations were made for the comparison between the rectangular cookie cutter Monte Carlo simulation and the JMEM method. The JMEM produced a more conservative damage estimate for all non-zero ballistic dispersion cases, but a less conservative estimate for the zero ballistic dispersion cases.

- The last set of comparisons was for the case of a stick of four weapons released against an area target. There were four weapon effectiveness methodologies compared in this section of the thesis. These methods were the old JMEM method, the new JMEM method, a Monte Carlo simulation using the Carleton damage function, and a Monte Carlo simulation using the rectangular cookie cutter damage function. Both Monte Carlo simulations, when compared to the new JMEM method, consistently gave smaller damage estimates of the stick on the area target. The only times the Monte Carlo simulations gave higher damage estimates were for some zero ballistic dispersion cases. When the two JMEM methods were compared, the older JMEM gave damage results that were significantly lower than those produced by the new JMEM method for cases on non-zero ballistic dispersion. For cases of zero

ballistic dispersion we saw that the damage results from the old JMEM method were appreciably greater than the new JMEM results.

- The number of iterations needed for the Monte Carlo simulations, as well as the computational time it took to run the simulations, was documented. The computer platform used to run all Monte Carlo simulations was the Matlab program running on an Intel Pentium 4 processor at 2.4 GHz. For the single weapon versus unitary target scenario, both Monte Carlo simulations required 10,000,000 iterations, and averaged a computing time of 10 minutes. The single weapon versus area target simulations also required 10,000,000 iterations, but averaged a computing time of 15 minutes. Lastly, the stick versus area target Monte Carlo simulations needed 100,000 iterations and averaged a computing time of 20 minutes.

- The Matlab program was initially used because it was convenient at the time. It is understood that other program languages, such as C or FORTRAN, are better suited for high speed calculations, and may have reduced computing time.

As summarized above, each comparison showed that the weapon effectiveness models for different weapon/target interactions differed in their damage estimates. These differences were expected for all of the comparisons due to the different approximations and assumptions made with in each of the methods' damage calculations.

THIS PAGE INTENTIONALLY LEFT BLANK

VII. CONCLUSIONS

The results of the comparisons run in this thesis have lead to the formulation of key conclusions. These conclusions help to give an overview of the findings of the comparisons between the JMEM methods and Monte Carlo simulations.

- The Monte Carlo simulations used to find the probability of damage for the single weapon against unitary and area targets needed ten million iterations to converge to a single answer. For sticks, these simulations required one hundred thousand iterations. These are a significant number of iterations, and may not be feasible to utilize in the operational arena due to slow computing speeds.

- The damage probability of the single weapon versus unitary and area target cases, calculated using the Monte Carlo simulations, were consistently greater than the JMEM method results for cases of non-zero ballistic dispersion. As the ballistic dispersion increased, the difference between the Monte Carlo and JMEM result increased, with the Monte Carlo result always being larger.

- For the single weapon versus a unitary target scenario with zero ballistic dispersion, the results from the Monte Carlo simulation representing the weapon as a lethal area matrix filled with values from the Carleton damage function matched the JMEM results within 1%. This was expected since the JMEM uses the Carleton damage function to determine the probability of damage, as does the Monte Carlo simulation.

- For the single weapon versus an area target scenario with zero ballistic dispersion, the results from the Monte Carlo simulation representing the weapon as a rectangular cookie cutter approximation of the Carleton damage function matched the JMEM results within 5%. This closeness was expected because the JMEM method for a single weapon versus an area target utilizes a rectangular cookie cutter approach to solving for weapon damage.

- The damage probability of the stick cases calculated by the Monte Carlo simulations was consistently smaller than the JMEM method results for cases of non-zero ballistic dispersion. It is interesting to note that the behavior of the results is opposite that of the single weapon cases. As the ballistic dispersion increased for the stick weapons, the resulting difference between the Monte Carlo and JMEM methods stayed generally constant, with the Monte Carlo result always being smaller than the JMEM result.

- The comparisons done between the new and old JMEM methods for sticks show that the new JMEM is a great deal more conservative than the old JMEM for zero ballistic dispersion cases. This may be the reason for the old JMEM being replaced.

By analyzing the data obtained in this thesis we have been able to make conclusions about the JMEM methods and Monte Carlo simulations. The goal of the Monte Carlo simulations was to give a more realistic way of modeling the weapon/target interaction. By running these Monte Carlo simulations, we have insight into the degree of the approximations used in the JMEM methods.

VIII. RECOMMENDATIONS AND FUTURE WORK

The goal of this thesis was to compare current weapon effectiveness methods with Monte Carlo simulations to determine whether the current JMEM methods should be replaced. This analysis included comparing damage results produced from the methods, as well as computational time needed to run the methods. By analyzing the data produced by the different methods, it is concluded that the Monte Carlo simulations better represent what actually occurs in the weapon target interaction than the current JMEM techniques. This is probably due to the amount of approximations that the JMEM incorporates into its methods. It is believed that by running the Monte Carlo simulation without these approximations of the Carleton damage function, we obtain damage estimates not tainted by approximating errors in the method. Due to this, the Monte Carlo simulation should be eventually integrated by the military in weapon effectiveness calculations. Though the Monte Carlo simulations are a higher fidelity modeling technique, it is not practical for the military to implement these methods as this time. This is due to the large number of iterations needed to obtain a correct result from the Monte Carlo simulations. These required iterations cannot be completed fast enough with current computing speeds to provide a quick reliable prediction of damage in the battlefield. However, while computing speeds keep increasing in the future, Monte Carlo simulations will become a practical alternative to the current JMEM methods.

The data collected for the JMEM and Monte Carlo simulations in this thesis were limited in time. Listed below are suggestions for follow on work in the topic of weapon effectiveness techniques.

- As of now, the methods all require user inputs of delivery accuracy and weapon lethality. It would be practical to include weapon release conditions from the aircraft as user inputs as well. The user could choose

a weapon release mechanisms which would include weapon trajectory analysis to determine the delivery accuracy of the weapon or stick.

- To test the trends of the data obtained through the comparisons of the JMEM methods and Monte Carlo simulations, a wider array of weapon lethalities should be tested (different user inputted MAE_F values) for a greater number of delivery accuracy combinations.

- The Matlab codes provided in Appendices A and B were not fully debugged and written in the most effective of manners due to time constraints on this thesis. By rewriting the Monte Carlo simulation codes for better performance, the speed by which a damage result is calculated may decrease.

- In order to have all methods coded in Matlab, the new JMEM method for sticks of weapons versus area targets should be coded along the lines of code given in Appendix B. This would enable all of the stick methods detailed in this thesis to be accessed by one program.

- The stick Matlab code provided in Appendix B should be written to enable the user to input the number of stick weapon elements are being dropped, This could also implement weapon release conditions so that the Matlab stick program could cover all stick configurations, rather than being hard coded for a four weapon stick with defined weapon impact points.

- As mentioned before, Matlab was chosen as the programming language for the JMEM and Monte Carlo simulation methods because the program was convenient at the time. It was originally thought that we could exploit other faculties of Matlab to simplify the coding of the Monte Carlo simulations. Coding the methods discussed in this thesis in programming

languages like C or FORTRAN could say valuable computing time, for it is known that C and FORTRAN are generally more efficient coding languages.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX A

```
%INPUTS SAME AS JMEM/AS

%Computes and compares the EFD based on four options.
%1) Lethal area matrix filled using the Carleton Damage function,
%2) Rectangular cookie cutter with Monte Carlo simulation
%3) JMEM single weapon vs. area targets with rectangualr cookie cutter-Ch8
%4) JMEM single weapon vs. unitary target with rectangular cookie-Ch7



function [x]=efdjmem(model)

%Define number of total trials in Monte Carlo Simulation
maxiter=1000000

%-------------------------------------------------------------------------
%USER INPUTS
REP=25;       %range error probable
DEP=25;       %deflection error probable
SR= 10000;    %slant range
I=45;         %impact angle degrees
sigma_b=0;    %Ballistic dispersion in mrads, standard deviation
MAEF=1000;
I_radians=I*pi/180;

a=max(1-.8*cos(I_radians),.3);

% Calculate the ballistic dispersion errors in ft in the ground plane
x_br=0.6745*SR*sigma_b/(1000*sin(I_radians));
x_bd=0.6745*SR*sigma_b/1000;

%Calculate weapon radii and blast radius
WRr=sqrt(MAEF*a/pi);
WRd=WRr/a;
RBI=0;
%-------------------------------------------------------------------------

%Create Ground plane grid
matrixsize=1001;
dmatrix=zeros(matrixsize);

%Define size of weapon matrix
weapon_size_x=round(4*WRr)+1;
weapon_size_y=round(4*WRd)+1;
weaponmatrix=zeros(weapon_size_x,weapon_size_y);

%Initialize target matrix
size_t_range=50;
size_t_deflection=50;
t=zeros(size_t_range, size_t_deflection);

%-------------------------------------------------------------------------

%Offset the center of the weapon matrix
offset=ceil(matrixsize/2);


%Define and place the weapon matrix in the ground plane using the Carleton damage function

h=1;
for i=0:weapon_size_x-1         %------------looping through the columns
    for n=0:weapon_size_y-1     %------------looping through the rowa
```

```
      g=exp(-(((i-floor(weapon_size_x/2))*(i-floor(weapon_size_x/2))/(WRr*WRr))+(((n-floor(weapon_size_y/2))*(n-
floor(weapon_size_y/2)))/(WRd*WRd))));
      r=sqrt(((i-floor(weapon_size_x/2))^2)+((n-floor(weapon_size_y/2))^2));

      if (r<=RBl)
         g=1;
      end
      dmatrix(i+offset-(floor(weapon_size_x/2)),n+offset-(floor(weapon_size_y/2)))=g;

      %Fills in weapon
      weaponmatrix(i+1,n+1)=g;

      h=h+1;
      n=n+1;
   end  %end of n loop
   i=i+1;
end  %end of i loop



%Define mean values and standard deviations of the range and deflection
%for...

% Aiming Error
meanrange_aim=offset;
meandeflection_aim=offset;
sigmarange_aim=REP/0.6745;
sigmadeflection_aim=DEP/0.6745;

%Ballistic Dispersion
meanrange_bd=0;
meandeflection_bd=0;
sigmarange_bd=x_br;
sigmadeflection_bd=x_bd;

%Initialize final sum of target cells
tfinalaverage=0;

%*************************************************************************
%*************************************************************************

%Lethal Area Matrix based on Carleton damage function

if model==1

%Trial loop
for z=1:maxiter

rrange_aim=(randn*sigmarange_aim);
rdefl_aim=(randn*sigmadeflection_aim);

rrange_bd=(randn*sigmarange_bd);
rdefl_bd=(randn*sigmadeflection_bd);

%Places the top left cell of the target matrix using a normal distribution
%of random numbers (rounded in order to ensure whole numbers for the matrix cell

s_range=round(rrange_aim+rrange_bd+meanrange_aim+meanrange_bd);
s_deflect=round(rdefl_aim+rdefl_bd+meandeflection_aim+meandeflection_bd);


k=1; %Counter for the target row currently being filled

%-------------------------------------------------------------------------------------------

for j=s_range:s_range+size_t_range-1   %loop to work through rows in weapon matrix
```

```
d=dmatrix(j-(floor(size_t_range/2)),:);       % d is defined as the current row in the weapon we are looking at


for i=1:size_t_deflection                %takes current target row and fills it with appropriate values of weapon row d
   t(k,i)=d(s_deflect+i-1-(floor(size_t_range/2)));
   i=i+1;

   if i>size_t_range
      break
   end
end  %end of i loop

k=k+1;
j=j+1;
end  %end of j loop
t;

%Loop to sum target elements
t_sum=0;
for p=1:(size_t_range*size_t_deflection)
   t_sum=t(p)+t_sum;
   p=p+1;
end  %end of t loop

%Calculates the average cell value for a single iteration
t_average=t_sum/(size_t_range*size_t_deflection);

%Counter to keep track of all of the calculated cell values from above
tfinalaverage=t_average+tfinalaverage;

z=z+1;
end   %end of z loop

%Takes the average of all of the averages obatined from the trials
EFD=tfinalaverage/maxiter

%**************************************************************************
%**************************************************************************

%Rectangular Cookie Cutter Model

elseif model==2

%Find MAEF by summing elements of weapon matrix
weapon_sum=0;

for j=1:(weapon_size_x*weapon_size_y)
   weapon_sum=weaponmatrix(j)+weapon_sum;
   j=j+1;
end  %end of t loop

MAEF=weapon_sum

%Define ratio of WRr/WRd
a=WRr/WRd;

%Define LET and WET
LET=sqrt(MAEF*a);
WET=LET/a;

%Pd=1 in the rectangle centered in the ground plane with dimensions given
%by WRr WRd
dmatrix=zeros(matrixsize);

for i=0:LET-1
   for n=0:WET-1
      dmatrix(i+offset-(floor(LET/2)),n+offset-(floor(WET/2)))=1;
      n=n+1;
```

```
   end  %end of n loop
   i=i+1;
end  %end of i loop

%Initialize final sum of target cells
tfinalaverage=0;

%Trial loop
for z=1:maxiter

rrange_aim=(randn*sigmarange_aim);
rdefl_aim=(randn*sigmadeflection_aim);

rrange_bd=(randn*sigmarange_bd);
rdefl_bd=(randn*sigmadeflection_bd);

%Places the top left cell of the target matrix using a normal distribution
%of random numbers (rounded in order to ensure whole numbers for the matrix cell

s_range=round(rrange_aim+rrange_bd+meanrange_aim+meanrange_bd);
s_deflect=round(rdefl_aim+rdefl_bd+meandeflection_aim+meandeflection_bd);


k=1; %Counter for the target row currently being filled

%-------------------------------------------------------------------------------------

for j=s_range:s_range+size_t_range-1   %loop to work through rows in weapon matrix


d=dmatrix(j-(floor(size_t_range/2)),:);       % d is defined as the current row in the weapon we are looking at



for i=1:size_t_deflection                 %takes current target row and fills it with appropriate values of weapon row d
   t(k,i)=d(s_deflect+i-1-(floor(size_t_range/2)));
   i=i+1;

   if i>size_t_range
      break
   end
end  %end of i loop

k=k+1;
j=j+1;
end  %end of j loop
t;

%Loop to sum target elements
t_sum=0;
for p=1:(size_t_range*size_t_deflection)
   t_sum=t(p)+t_sum;
   p=p+1;
end  %end of t loop

%Calculates the average cell value for a single iteration
t_average=t_sum/(size_t_range*size_t_deflection);

%Counter to keep track of all of the calculated cell values from above
tfinalaverage=t_average+tfinalaverage;

z=z+1;
end   %end of z loop

%Takes the average of all of the averages obatined from the trials
EFD=tfinalaverage/maxiter


%************************************************************************
```

```
%**********************************************************************


%JMEMs technique Chapter 8 - one weapon aginst area of targets - option #3

elseif model==3

%Define reliabilty of weapon
R=1;

% Calculate and combine BD and aiming error

x_br=0.6745*SR*sigma_b/(1000*sin(I_radians));
x_bd=0.6745*SR*sigma_b/1000;

%   RSS error probables delivery accuracies

REP_dash=sqrt((REP*REP)+(x_br*x_br));
DEP_dash=sqrt((DEP*DEP)+(x_bd*x_bd));


sigma_range=REP_dash/.6745;
sigma_deflection=DEP_dash/.6745;

%Define LET and WET
LET=sqrt(MAEF*a);
WET=LET/a;

%Define LEP and WEP
LEP=max(LET,size_t_range);
WEP=max(WET,size_t_deflection);

%---------------------------------------
%Fractional coverage in range

s_range=(LEP+size_t_range)/2
t_range=(LEP-size_t_range)/2

%Solving piecewise integral for range

I1=((normcdf(t_range,0,sigma_range))-(normcdf(-t_range,0,sigma_range)))
I2=((LEP+size_t_range)/(2*size_t_range))*(normcdf(-t_range,0,sigma_range)-normcdf(-s_range,0,sigma_range))
I3=((LEP+size_t_range)/(2*size_t_range))*(normcdf(s_range,0,sigma_range)-normcdf(t_range,0,sigma_range))

a_range=s_range/(sigma_range*sqrt(2))
b_range=t_range/(sigma_range*sqrt(2))

I4minusI5=((2*sigma_range)/(size_t_range*sqrt(2*pi)))*(exp(-(a_range*a_range))-exp(-(b_range*b_range)))

%Fractional coverage in range
EFRange=I1+I2+I3+I4minusI5


%Fractional coverage in deflection

s_deflection=(WEP+size_t_deflection)/2
t_deflection=(WEP-size_t_deflection)/2

%Solving piecewise integral for deflection

I1=(normcdf(t_deflection,0,sigma_deflection)-normcdf(-t_deflection,0,sigma_deflection))
I2=((WEP+size_t_deflection)/(2*size_t_deflection))*(normcdf(-t_deflection,0,sigma_deflection)-normcdf(-
s_deflection,0,sigma_deflection))
I3=((WEP+size_t_deflection)/(2*size_t_deflection))*(normcdf(s_deflection,0,sigma_deflection)-
normcdf(t_deflection,0,sigma_deflection))

a_deflection=s_deflection/(sigma_deflection*sqrt(2))
b_deflection=t_deflection/(sigma_deflection*sqrt(2))
```

```
I4minusI5=((2*sigma_deflection)/(size_t_deflection*sqrt(2*pi)))*(exp(-(a_deflection*a_deflection))-exp(-
(b_deflection*b_deflection)))

%Fractional coverage in deflection
EFDeflection=I1+I2+I3+I4minusI5

%-------------------------------------------------------
%Compute composite fractional coverage
EFC=EFRange*EFDeflection;

%Compute fractional damage
EFD=EFC*R*LET*WET/(LEP*WEP)

%*************************************************************************
%*************************************************************************
%Option 4
%JMEMs technique Chapter 7 -one weapon against a unitary target
else

%RSS error probables delivery accuracies

REP_dash=sqrt((REP*REP)+(x_br*x_br));
DEP_dash=sqrt((DEP*DEP)+(x_bd*x_bd));

LETprime=1.128*(sqrt(MAEF*a));
WETprime=LETprime/a;

SSPD=(LETprime*WETprime)/(sqrt(((17.6*REP_dash*REP_dash)+(LETprime*LETprime))*((17.6*DEP_dash*DEP_dash)
+(WETprime*WETprime))))


end
```

APPENDIX B

```
%Computes and compares the EFD of Stick Deliveries based on 2 options
% 1)Lethal area matrices filled using the Carleton Damage function for each
%   weapon in stick.
% 2)LAM method with rectangular cookie cutter for each weapon
% 3)WIN-JMEM fractional coverage of each weapon independently method
% 4)Ch 9 stick calculating methods

function [x]=stick(model)

%Define number of total trials
maxiter=100000

%------------------------------------------------------------------------
%USER INPUTS
REP=50;       %range error probable
DEP=50;       %deflection error probable
SR= 10000;    %slant range
I=45;         %impact angle degrees
sigma_b=5;    %Ballistic dispersion in mrads, standard deviation
MAEF=1000;
I_radians=I*pi/180;
a=max(1-.8*cos(I_radians),.3)  %Aspect ratio of weapon radii

R=1;          %Reliabilty of weapon

%Calculate weapon radii and blast radius
WRr=sqrt(MAEF*a/pi)
WRd=WRr/a
RBl=0;
%------------------------------------------------------------------------
%Create Ground plane grid
matrixsize=1001;
dmatrix=zeros(matrixsize);

%Define size of weapon matrix
weapon_size_x=round(4*WRr)+1;
weapon_size_y=round(4*WRd)+1;
weaponmatrix=zeros(weapon_size_x,weapon_size_y);

%Initialize target matrix
size_t_range=50;
size_t_deflection=50;
t=zeros(size_t_range, size_t_deflection);
%------------------------------------------------------------------------



% Calculate the ballistic dispersion errors in ft in the ground plane
x_br=SR*sigma_b/(1000*sin(I_radians));
x_bd=SR*sigma_b/1000;

%Stick User Inputs
totalweapons=4;             %total number of weapons released
np=2;                       %Number of weapons released per pulse
dt=.2;                      %intervalometer setting (seconds)
Ws=100;                     %stick width
Ls=100;                     %stick length
PCD=1;                      %conditional probability of damage
%------------------------------------------------------------------------

if model==1
%Option 1 -- Carleton Damage Function methods

%Define mean values and standard deviations of the range and deflection
```

```
%for...

% Aiming Error
meanrange_aim=0;
meandeflection_aim=0;
sigmarange_aim=REP/0.6745;
sigmadeflection_aim=DEP/0.6745;

%Ballistic Dispersion
meanrange_bd=0;
meandeflection_bd=0;
sigmarange_bd=x_br;
sigmadeflection_bd=x_bd;

%Initialize final sum of target cells
tfinalaverage=0;

%Trial loop
for z=1:maxiter

%Places the top left cell of the target matrix using a normal distribution
%of random numbers (rounded in order to ensure whole numbers for the matrix cell
rrange_aim=(randn*sigmarange_aim);
rdefl_aim=(randn*sigmadeflection_aim);


%Top left cell of target matrix influenced by aiming error of stick
s_range=round(rrange_aim+meanrange_aim);
s_deflect=round(rdefl_aim+meandeflection_aim);


%Place individual impact points including ballistic dispersion errors on
%each weapon
weapon1_range=-50+(randn*sigmarange_bd);
weapon1_deflection=-50+(randn*sigmadeflection_bd);
weapon2_range=-50+(randn*sigmarange_bd);
weapon2_deflection=50+(randn*sigmadeflection_bd);
weapon3_range=50+(randn*sigmarange_bd);
weapon3_deflection=-50+(randn*sigmadeflection_bd);
weapon4_range=50+(randn*sigmarange_bd);
weapon4_deflection=50+(randn*sigmadeflection_bd);


k=0;
i=0;

for k=0:size_t_range-1        % Row counter
for i=0:size_t_deflection-1   % Column counter

      Pd1=exp(-((((weapon1_range-(s_range-k))^2)/(WRr*WRr))+(((weapon1_deflection-(s_deflect+i))^2)/(WRd*WRd))));
      Pd2=exp(-((((weapon2_range-(s_range-k))^2)/(WRr*WRr))+(((weapon2_deflection-(s_deflect+i))^2)/(WRd*WRd))));
      Pd3=exp(-((((weapon3_range-(s_range-k))^2)/(WRr*WRr))+(((weapon3_deflection-(s_deflect+i))^2)/(WRd*WRd))));
      Pd4=exp(-((((weapon4_range-(s_range-k))^2)/(WRr*WRr))+(((weapon4_deflection-(s_deflect+i))^2)/(WRd*WRd))));

   t(k+1,i+1)=1-((1-Pd1)*(1-Pd2)*(1-Pd3)*(1-Pd4));

i=i+1;

end
k=k+1;
i=0;
end

%Loop to sum target elements
t_sum=0;
for p=1:(size_t_range*size_t_deflection)
   t_sum=t(p)+t_sum;
   p=p+1;
end  %end of t loop
```

```
%Calculates the average cell value for a single iteration
t_average=t_sum/(size_t_range*size_t_deflection);

%Counter to keep track of all of the calculated cell values from above
tfinalaverage=t_average+tfinalaverage;

z=z+1;
end   %end of z loop

%Takes the average of all of the averages obatined from the trials
EFD=tfinalaverage/maxiter
t;

%*************************************************************************
%*************************************************************************
elseif model==2

%Option 2
%Rectangular cookie cutter method

offset=500;

%Creates matrix filled by Carleton damage function
h=1;
for i=0:weapon_size_x-1         %------------looping through the columns
    for n=0:weapon_size_y-1    %------------looping through the rowa

       g=exp(-(((i-floor(weapon_size_x/2))*(i-floor(weapon_size_x/2))/(WRr*WRr))+(((n-floor(weapon_size_y/2))*(n-
floor(weapon_size_y/2)))/(WRd*WRd))));
       r=sqrt(((i-floor(weapon_size_x/2))^2)+((n-floor(weapon_size_y/2))^2));

       if (r<=RBl)
          g=1;
       end
       dmatrix(i+offset-(floor(weapon_size_x/2)),n+offset-(floor(weapon_size_y/2)))=g;

       %Fills in wepaon
       weaponmatrix(i+1,n+1)=g;

       h=h+1;
       n=n+1;
   end  %end of n loop
   i=i+1;
end  %end of i loop



%Find MAEF by summing elements of weapon matrix
weapon_sum=0;

for j=1:(weapon_size_x*weapon_size_y)
   weapon_sum=weaponmatrix(j)+weapon_sum;
   j=j+1;
end  %end of t loop



%Define LET and WET
LET=sqrt(MAEF*a)
WET=LET/a



%Define mean values and standard deviations of the range and deflection
%for...

% Aiming Error
meanrange_aim=0;
```

```
meandeflection_aim=0;
sigmarange_aim=REP/0.6745;
sigmadeflection_aim=DEP/0.6745;

%Ballistic Dispersion
meanrange_bd=0;
meandeflection_bd=0;
sigmarange_bd=x_br;
sigmadeflection_bd=x_bd;

%Initialize final sum of target cells
tfinalaverage=0;

%Trial loop
for z=1:maxiter

%Places the top left cell of the target matrix using a normal distribution
%of random numbers (rounded in order to ensure whole numbers for the matrix cell
rrange_aim=(randn*sigmarange_aim);
rdefl_aim=(randn*sigmadeflection_aim);


%Top left cell of target matrix influenced by aiming error of stick
s_range=round(rrange_aim+meanrange_aim);
s_deflect=round(rdefl_aim+meandeflection_aim);

%Place individual impact points including ballistic dispersion errors on
%each weapon
weapon1_range=-50+(randn*sigmarange_bd);
weapon1_deflection=-50+(randn*sigmadeflection_bd);
weapon2_range=-50+(randn*sigmarange_bd);
weapon2_deflection=50+(randn*sigmadeflection_bd);
weapon3_range=50+(randn*sigmarange_bd);
weapon3_deflection=-50+(randn*sigmadeflection_bd);
weapon4_range=50+(randn*sigmarange_bd);
weapon4_deflection=50+(randn*sigmadeflection_bd);

%Pd=1 in the rectangle centered in the ground plane with dimensions given
%by WRr WRd

for r=1:size_t_range
      for d=1:size_t_deflection

         if (abs(r+weapon1_range+s_range)<=LET/2 &
abs(d+weapon1_deflection+s_deflect)<=WET/2)|(abs(r+weapon2_range+s_range)<=LET/2 &
abs(d+weapon2_deflection+s_deflect)<=WET/2)|(abs(r+weapon3_range+s_range)<=LET/2 &
abs(d+weapon3_deflection+s_deflect)<=WET/2)|(abs(r+weapon4_range+s_range)<=LET/2 &
abs(d+weapon4_deflection+s_deflect)<=WET/2)
            t(r,d)=1;
         else
            t(r,d)=0;
         end

      d=d+1;
   end
   r=r+1;
   d=0;
end



%Loop to sum target elements
t_sum=0;
for p=1:(size_t_range*size_t_deflection)
   t_sum=t(p)+t_sum;
   p=p+1;
end  %end of t loop

%Calculates the average cell value for a single iteration
```

```
t_average=t_sum/(size_t_range*size_t_deflection);

%Counter to keep track of all of the calculated cell values from above
tfinalaverage=t_average+tfinalaverage;

z=z+1;
end   %end of z loop

%Takes the average of all of the averages obatined from the trials
EFD=tfinalaverage/maxiter



%**************************************************************************
%**************************************************************************

elseif model==3

%Option 3
%WIN-JEM calculation method

%Ballistic Dispersion Effects
LET=sqrt(MAEF*a)
WET=LET/a

%Define ballistic dispersion as standard deviations in the ground plane
sigma_br=SR*(sigma_b)/(1000*sin(I_radians))
sigma_bd=SR*(sigma_b)/1000

%Define LB and WB
LB=sqrt(LET^2+(8*((sigma_br)^2)))
WB=sqrt(WET^2+(8*((sigma_bd)^2)))

%Define conditional probability of damage from enlarged effected area
PCD1=PCD*LET*WET/(LB*WB)

%Convert REP and DEP into sigmas
sigma_range=(REP/.6745)
sigma_deflection=(DEP/.6745)

%Define LEP and WEP
LEP=max(LB,size_t_range)
WEP=max(WB,size_t_deflection)

%Fractional coverage in range

s_range=(LEP+size_t_range)/2
t_range=(LEP-size_t_range)/2

%Solving piecewise integral for range

I1=((normcdf(t_range,50,sigma_range))-(normcdf(-t_range,50,sigma_range)))
I2=((LEP+size_t_range)/(2*size_t_range))*(normcdf(-t_range,50,sigma_range)-normcdf(-s_range,50,sigma_range))
I3=((LEP+size_t_range)/(2*size_t_range))*(normcdf(s_range,50,sigma_range)-normcdf(t_range,50,sigma_range))

I4=((-sigma_range)/(size_t_range*sqrt(2*pi)))*((exp(-((-t_range-50)^2)/(2*(sigma_range^2))))-(exp(-((-s_range-
50)^2)/(2*(sigma_range^2)))))+((50/size_t_range)*(normcdf(-t_range,50,sigma_range)-normcdf(-
s_range,50,sigma_range)))

I5=((-sigma_range)/(size_t_range*sqrt(2*pi)))*((exp(-((s_range-50)^2)/(2*(sigma_range^2))))-(exp(-((t_range-
50)^2)/(2*(sigma_range^2)))))+((50/size_t_range)*(normcdf(s_range,50,sigma_range)-normcdf(t_range,50,sigma_range)))

%Fractional coverage in range
EFRange=I1+I2+I3+I4-I5


%Fractional coverage in deflection

s_deflection=(WEP+size_t_deflection)/2
```

```
t_deflection=(WEP-size_t_deflection)/2

%Solving piecewise integral for deflection

I1=(normcdf(t_deflection,50,sigma_deflection)-normcdf(-t_deflection,50,sigma_deflection))
I2=((WEP+size_t_deflection)/(2*size_t_deflection))*(normcdf(-t_deflection,50,sigma_deflection)-normcdf(-
s_deflection,50,sigma_deflection))
I3=((WEP+size_t_deflection)/(2*size_t_deflection))*(normcdf(s_deflection,50,sigma_deflection)-
normcdf(t_deflection,50,sigma_deflection))

I4=((-sigma_deflection)/(size_t_deflection*sqrt(2*pi)))*((exp(-((-t_deflection-50)^2)/(2*(sigma_deflection^2))))-(exp(-((-
s_deflection-50)^2)/(2*(sigma_deflection^2)))))+((50/size_t_deflection)*(normcdf(-t_deflection,50,sigma_deflection)-
normcdf(-s_deflection,50,sigma_deflection)))

I5=((-sigma_deflection)/(size_t_deflection*sqrt(2*pi)))*((exp(-((s_deflection-50)^2)/(2*(sigma_deflection^2))))-(exp(-
((t_deflection-50)^2)/(2*(sigma_deflection^2)))))+((50/size_t_deflection)*(normcdf(s_deflection,50,sigma_deflection)-
normcdf(t_deflection,50,sigma_deflection)))

%Fractional coverage in deflection
EFDeflection=I1+I2+I3+I4-I5

%-------------------------------------------------------
%Compute composite fractional coverage
EFC=EFRange*EFDeflection

%Compute fractional damage from one weapon
singleEFD=PCD1*EFC*R

%Total EFD from 4 weapons
EFD=1-((1-singleEFD)*(1-singleEFD)*(1-singleEFD)*(1-singleEFD))

%**************************************************************************
%**************************************************************************

else

%Option 4
%Old JMEM Method

%Define LET and WET
LET=sqrt(MAEF*a);
WET=LET/a;

%Define ballistic dispersion as standard deviations in the ground plane
sigma_br=SR*(sigma_b)/(1000*sin(I_radians))
sigma_bd=SR*(sigma_b)/1000

%Define LB and WB
LB=sqrt(LET^2+(8*(sigma_br)^2))
WB=sqrt(WET^2+(8*(sigma_bd)^2))


%Define conditional probability of damage from enlarged effected area
PCD1=PCD*LET*WET/(LB*WB);

% Calculate delivery accuracy errors

sigma_range=REP/.6745;
sigma_deflection=DEP/.6745;

%Define Pattern Dimensions
Lp=Ls+LB;
Wp=Ws+WB;

%Overlap
nr=totalweapons/np          %Number of intervalometer pulses

%Define Degree of overlap in the deflection direction
nod=np*WB/Wp;
```

```
%Define the conditional probability of damage in the deflection direction
if nod>=1
   PCDd=1-(1-PCD1)^nod;
else
   PCDd=np*PCD1*WB/Wp;
end

%Define Degree of overlap in the range direction
nor=nr*LB/Lp;

%Calculate the conditional probability for the whole pattern
if nor>=1
   PCDS=1-(1-PCDd)^nor;
else
   PCDS=nr*PCDd*LB/Lp;
end

%---------------------------------------
%Define LEP and WEP
LEP=max(Lp,size_t_range);
WEP=max(Wp,size_t_deflection);

%Fractional coverage in range
s_range=(LEP+size_t_range)/2;
t_range=(LEP-size_t_range)/2;

%Solving piecewise integral for range
I1=((normcdf(t_range,0,sigma_range))-(normcdf(-t_range,0,sigma_range)));
I2=((LEP+size_t_range)/(2*size_t_range))*(normcdf(-t_range,0,sigma_range)-normcdf(-s_range,0,sigma_range));
I3=((LEP+size_t_range)/(2*size_t_range))*(normcdf(s_range,0,sigma_range)-normcdf(t_range,0,sigma_range));

a_range=s_range/(sigma_range*sqrt(2));
b_range=t_range/(sigma_range*sqrt(2));

I4minusI5=((2*sigma_range)/(size_t_range*sqrt(2*pi)))*(exp(-(a_range*a_range))-exp(-(b_range*b_range)));

%Fractional coverage in range
EFRange=I1+I2+I3+I4minusI5


%Fractional coverage in deflection
s_deflection=(WEP+size_t_deflection)/2;
t_deflection=(WEP-size_t_deflection)/2;

%Solving piecewise integral for deflection
I1=(normcdf(t_deflection,0,sigma_deflection)-normcdf(-t_deflection,0,sigma_deflection));
I2=((WEP+size_t_deflection)/(2*size_t_deflection))*(normcdf(-t_deflection,0,sigma_deflection)-normcdf(-
s_deflection,0,sigma_deflection));
I3=((WEP+size_t_deflection)/(2*size_t_deflection))*(normcdf(s_deflection,0,sigma_deflection)-
normcdf(t_deflection,0,sigma_deflection));

a_deflection=s_deflection/(sigma_deflection*sqrt(2));
b_deflection=t_deflection/(sigma_deflection*sqrt(2));

I4minusI5=((2*sigma_deflection)/(size_t_deflection*sqrt(2*pi)))*(exp(-(a_deflection*a_deflection))-exp(-
(b_deflection*b_deflection)));

%Fractional coverage in deflection
EFDeflection=I1+I2+I3+I4minusI5

%Compute composite fractional coverage
EFC=EFRange*EFDeflection;

%Compute fractional damage
EFD=EFC*Lp*Wp*R*PCDS/(LEP*WEP)

end %end of model if/else statement
```

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF REFERENCES

1. Driels, M., *Weaponeering*, AAIA, 2004.

2. Joint Technical Coordination Group for Munitions Effectiveness, *Derivation of JMEM/AS Open-End Methods,* Distribution authorized to U.S. Government agencies and their contractors only, 2004.

THIS PAGE INTENTIONALLY LEFT BLANK

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, VA

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, CA

3. Mechanical Engineering Department Chairman, Code ME
   Naval Postgraduate School
   Monterey, CA

4. Naval/Mechanical Engineering Curriculum Code 34
   Naval Postgraduate School
   Monterey, CA

5. Professor Morris R. Driels, Code ME/Dr
   Department of Mechanical Engineering
   Naval Postgraduate School
   Monterey, CA

6. Commander NAWC-WD
   Attn: W. Tonkin 4J6000D/W
   China Lake, CA

7. Director US ANSAA
   Attn: AMXSY-SA (R. Chandler)
   Aberdeen Proving Ground, MD

8. ASC/XREW
   Attn: Carolyn Holland
   Eglin AFB, FL