NAVAL POSTGRADUATE SCHOOL

Monterey, California

THESIS

AN EVALUATION OF THE POISSON CLASS
OF TWO DIMENSIONAL EVASIVE GAME STRATEGIES
BY COMPUTER SIMULATION

by

Thomas John Clothier

Thesis Advisor: A. R. Washburn

March 1972

78

Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author) Naval Postgraduate School Monterey, California 93940	2a. REPORT SECURITY CLASSIFICATION Unclassified 2b. GROUP

3. REPORT TITLE

An Evaluation of the Poisson Class of Two Dimensional Evasive Game Strategies by Computer Simulation

4. DESCRIPTIVE NOTES (Type of report and, inclusive dates)

Master's Thesis; March 1972

5. AUTHOR(S) (First name, middle initial, last name)

Thomas John Clothier

6. REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
March 1972	78	4

8a. CONTRACT OR GRANT NO. b. PROJECT NO. c. d.	9a. ORIGINATOR'S REPORT NUMBER(S) 9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)

10. DISTRIBUTION STATEMENT

Approved for public release; distribution unlimited.

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY
	Naval Postgraduate School Monterey, California 93940

13. ABSTRACT

This paper evaluates five maneuvering strategies for an evader in a two dimensional continuous evasive game. An evader's movement, in two dimensions at constant speed, is simulated by choosing courses that are independent and distributed according to one of the five course change rules. The times between course changes are independent exponential random variables. An improvement over a previously established least upper bound on evader survivability is presented. The bulk of this paper is the presentation of evader survivability as a result of the maneuvering strategy he employs.

DD FORM 1 NOV 65 1473 (PAGE 1)

S/N 0101-807-6811

Security Classification

A-31408

Security Classification

14. KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
1. Evasive Games						
2. Continuous Games						
3. Time-Lagged Games						

DD FORM 1 NOV 65 1473 (BACK)

S/N 0101-807-6821

Security Classification

A-31409

An Evaluation of the Poisson Class
of Two Dimensional Evasive Game Strategies
by Computer Simulation

by

Thomas John Clothier
Lieutenant, United States Navy
B.A., University of Colorado, 1966

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN OPERATIONS RESEARCH

from the

NAVAL POSTGRADUATE SCHOOL
March 1972

Author Thomas J Clothier

Approved by: Alan R Washburn
Thesis Advisor

Chairman, Department of Operations Research
and Administrative Sciences

Academic Dean

ABSTRACT

This paper evaluates five maneuvering strategies for an evader in a two dimensional continuous evasive game. An evader's movement, in two dimensions at constant speed, is simulated by choosing courses that are independent and distributed according to one of the five course change rules. The times between course changes are independent exponential random variables. An improvement over a previously established least upper bound on evader survivability is presented. The bulk of this paper is the presentation of evader survivability as a result of the maneuvering strategy he employs.

TABLE OF CONTENTS

I. INTRODUCTION

A classic military problem with current application is how to maneuver a mobile target in order to prevent successful prediction of its position. For example how should a aircraft carrier maneuver to maximize its survivability against an ICBM attack?

Of primary importance to this problem is the time lag between the decision to fire at a target, made at time t, and the warhead detonation at time t+TL. This lag is called time-late and denoted as TL. The decision to initiate such an action is based in part upon knowing the target's position at time t while the effectiveness of that action is dependent upon the error in predicting that position at t+TL.

A. ICBM VERSUS AIRCRAFT CARRIER

Consider the ICMB versus aircraft carrier problem. Assume that the carrier operates in mid-ocean, possesses great endurance, can make sharp turns and has as its only kinematic restriction a constant speed, v. The carrier is referred to as the evader, E. The attacker, P, continuously observes E from a nearby unarmed trawler. P's weapon is a land based ICBM which has perfect accuracy, produces a lethal area A (this lethal area can have any shape) but does not have a mid-course guidance nor homing capability. Although P knows

E's position past and present, he also knows that his ICBM will have a one hour time-late.[1]

P's problem is to predict E's position at time t+TL in order to initiate an attack at time t. E's problem, since he will not know of an attack until he observes the detonation, is to maneuver in such a way to confound P's prediction.[2] E's movement should be random because any nonrandom movement would be vulnerable to extrapolation.

B. AN UPPER BOUND ON EVADER SURVIVABILITY

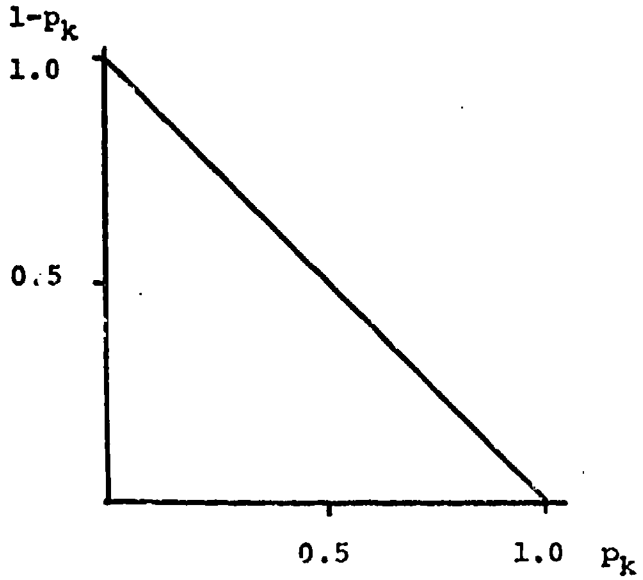
A lower bound on P's success, p_k, and consequently an upper bound on E's survivability, $1-p_k$, is known. P knows E's position at t+TL will be interior to or on a circle of radius v•TL centered on E's position at t (which is known). The enemy can then gaurantee a kill probability of at least $A/\pi(v \cdot TL)^2$ by choosing a lethal area randomly within the circle of uncertainty (a wedge of random orientation will do) [Ref. 4]. Then the maximum survivability E could attain would be $1-A/\pi(v \cdot TL)^2$.

This upper bound on E's survivability is a function only of P's strength, that is the magnitude of A (the value $\pi(v \cdot TL)^2$ is constant). Figure 1 is a graph of that survival function versus P's probability of success.

---

[1]Time-late is a summand of many factors in the command and control problem as well as the missle flight time. For the purpose used here it is sufficient to treat time-late in total and not consider its decomposition.

[2]If E knows the time at which an attack is initiated then the problem is trivial.



Survivability Versus Attacker Strength

FIGURE 1

Unfortunately the specific set of rules for E to maneuver by to achieve the survivability depicted in Fig. 1 is not known.

II. BACKGROUND

A. DISCRETE EVASIVE GAMES

It was felt that a two dimensional continuous game was too difficult to solve. So the approach was to solve a similar but easier game. It was assumed that the ocean was a linear set of discrete points and that E's mobility consisted of being able to jump either right or left to an adjacent point [Refs. 2 and 3]. Time-late became an integer number of jumps that E could make between the attack decision and warhead detonation. As in the exact problem E had no knowledge of the decision to attack.

Discrete problems of this type are called evasive games and are classified by the number of jumps constituting time-late. A one step discrete evasive game means E can move either left or right one jump prior to detonation. Game theory provides an immediate solution to that game; the value is p_k=0.5. Also there is an optimal strategy for E in that game. At each jump E should go left with probability one-half or right with probability one-half. Employment of such a rule for each jump guarantees E a survival probability of at least the game value, regardless of P's firing rule.

Having easily solved the one step game interest was focused on the two step game. In this game E was allowed two jumps during time-late. The solution has been obtained, but it was not as easy to achieve as that of the one step game [Ref.2].

The direction of the analysis was clear. Knowledge gained from solving the more simple games would be a stepping stone to the solution of the more difficult games. Eventually the assumptions of linearity and discreteness could be relaxed. Progress, however, in solving the discrete games has been very slow. Researchers are presently embroiled in solving the three step game.

B. A CONTINUOUS EVASIVE GAME

Washburn in Ref. 1 presented a different approach to analyzing the ICBM versus aircraft carrier problem. He developed the probability density function of a particle moving continuously in a two dimensional medium subject to a specific set of maneuvering rules. That approach was distinct from previous work because it addressed the exact problem. The results presented in this paper are an extension of that approach.

Washburn proposed the following strategy for E and analyzed the evader's subsequent survivability, first as a function of the attacker's strength and secondly as a function of a specific strategy parameter. E was to select courses from a uniform distribution. Course changes were to occur as a Poisson process. This meant the time between course changes would be exponentially distributed.

The exponential time between course changes has an intuitive appeal because of the memoryless property of that distribution. The memoryless property is: the probability that E will not change course in the future, given that he has not changed course for some observation period, is independent of the length of that period. Therefore, use of the exponential time distribution should serve to confound a prediction of future position by extrapolation, regardless of the course distribution used. For example, suppose P's attack decision rule required that he observe E maintain a constant course for at least five hours prior to initiating an attack. Such a procedure would not improve P's probability of a kill because the only information of any benefit at the time an attack is initiated is E's position and last course.

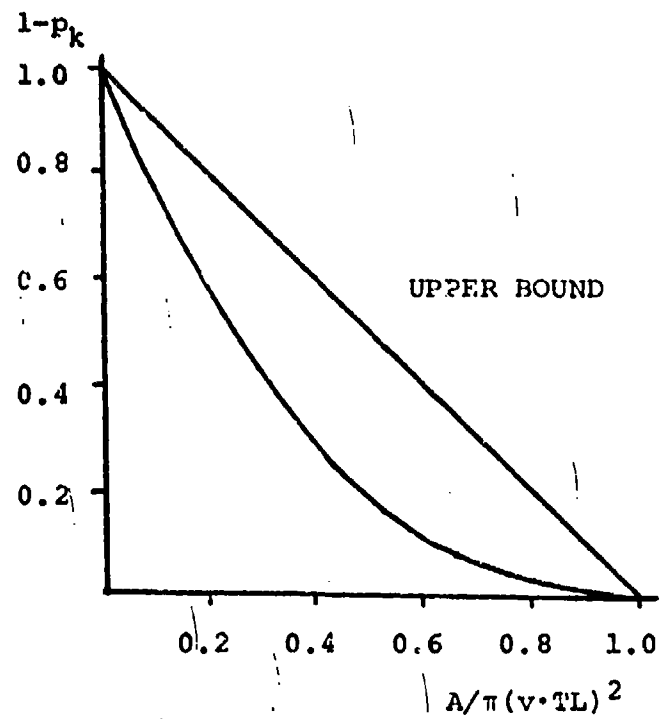
Figure 2 is a graph of the evader's maximum attainable survivability when using the strategy of uniform courses and exponential times versus P's strength for the optimal exponential parameter, λ.[3] For comparison the upper bound on survivability is also presented.

"If we assume the parameters A,v,TL are known to both sides, then the evader can select λ to maximize the survivability. The evader will clearly be in trouble if he makes λ too small, because the kill probability is at least $\exp(-\lambda \cdot TL)$.[4] On the other hand making course changes too frequently will lead to a density function that is highly peaked at the origin, which is equally undersirable..." [Ref. 1].

---

[3]The exponential parameter, λ, is the inverse of the mean time between course changes.

[4]$\mathrm{Exp}(-\lambda \cdot TL)$ is the probability that E will make no course change during TL.



Minimum p_k for Optimal Parameter λ Versus Attacker Strength; Strategy Is Uniform Courses and $\exp(\lambda)$ Times

FIGURE 2

"It is not known whether or not the optimal strategy for E is a Poisson strategy of the type just considered, or even whether the uniform distribution on angles is optimal within the class." [Ref. 2]

III. RESULTS OF SIMULATING THE CONTINUOUS EVASIVE GAME

The goal of this research was to determine if the uniform course rule was optimal within the class of Poisson strategies or to find a better rule if it was not. For another rule to be better than the uniform one it would have to enable E to attain a higher survivability than that shown in Fig. 2.

The methodology of solving discrete games had not advanced sufficiently to achieve those goals. The methods used by Washburn were also not suitable because of the mathematical difficulty of the problem.

The Poisson class of strategies is, however, uniquely suited for analysis by computer simulation. This is due to the memoryless property of the exponential times between course changes. The results of such a simulation are presented in this paper in the form of improved survivability for the evader.

Five course change rules were evaluated in the simulation. The probability density function of each is presented on subsequent pages along with a graph of the resulting attacker's p_k versus attacker strength, denoted as $S=A/\pi(v \cdot TL)^2$.[5] These graphs are for representative values of the parameter α. Alpha, α, is the product of time-late and the exponential parameter λ.

---

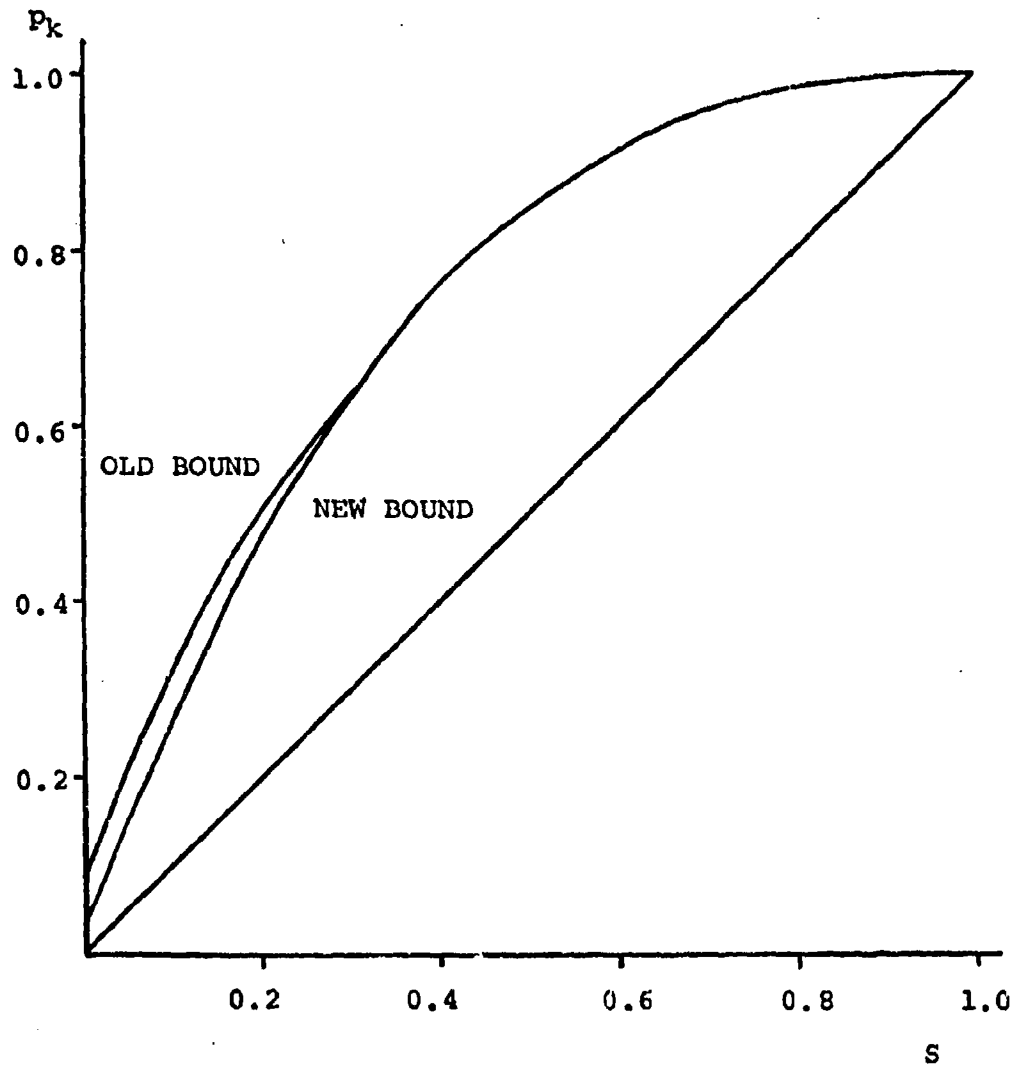
[5]Course changes were assumed to be independent of the underlying Poisson process.

The simulation established a new lower bound on the p_k the evader could yield to the attacker. Figure 3 is a graph illustrating the improvement of the new bound over that found by Washburn. This new bound is the least lower bound of all the p_k versus S curves for the four values of α and all rules simulated. Examples of such curves are shown in Figs. 5-9. The old bound in Fig. 3 is the lower bound of the p_k curves simulated using the uniform rule exclusively.

The minimal p_k as a function of α, over the five rules, is shown in Fig. 4 for three specific attacker strength levels. The curves in Fig. 4 confirm the statement in Ref. 2 that the evader should turn most often, that is α should be highest, against the weakest opponent.

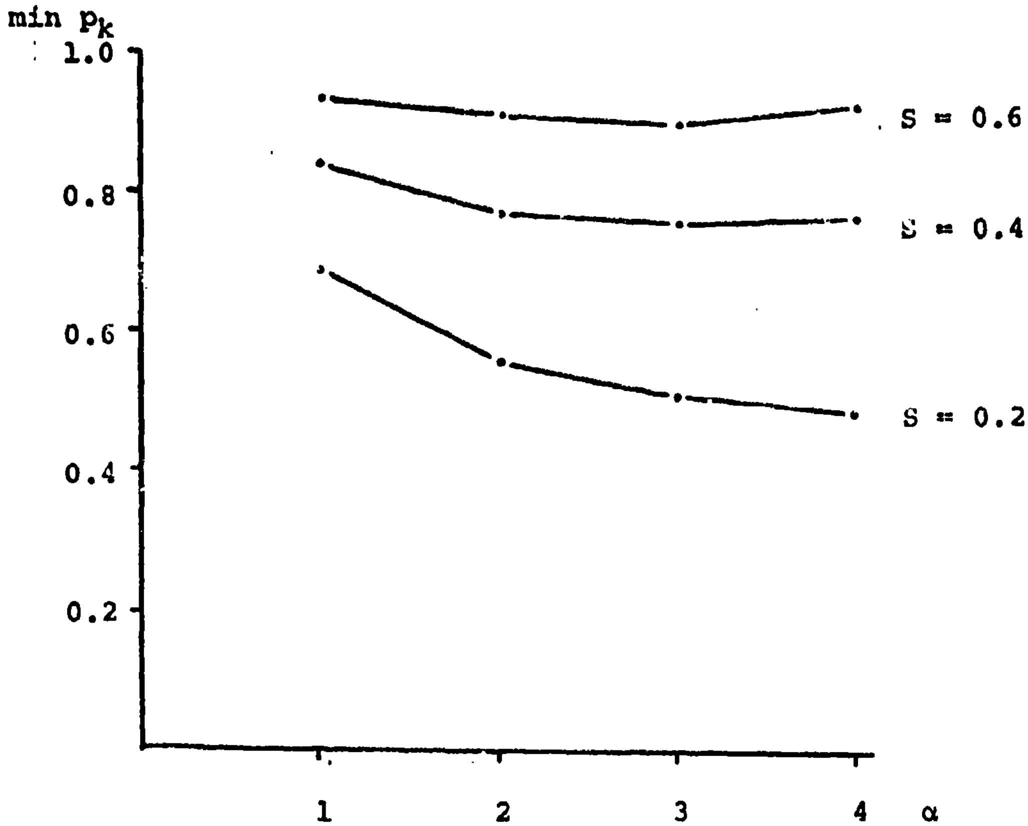
Comparisons of the p_k curves of the different rules have shown that the uniform rule is not optimal for all attacker strengths and α values. For example, the reverse course rule (see Fig. 7) was shown to be better than the uniform rule for $\alpha=1$. The difference between these two rules is illustrated in Fig. 10. Against a weak opponent (strength $\leq .6$) the evader would do better to use the reverse course rule rather than the uniform rule for $\alpha=1$. This improved survivability occurs because the reverse course rule created a "flatter" evader position density.

None of the five course rules evaluated produced a consistently smaller p_k for all values of attacker strengths and α. However, the left-right rule was dominated by the other four in all cases. Each of the five course rules was simulated at six different levels of α by varying ET. The graphs of p_k versus $A/\pi(v \cdot TL)^2$ for each simulation are presented in the Computer Output section along with tabulated p_k's for specific attacker strength levels as a function of the course rule and the α used.



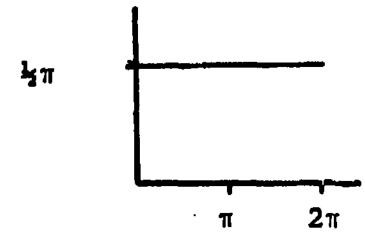
New Bound on p_k That the Evader Can Yield
as a Function of Attacker Strength

FIGURE 3



Minimum p_k, Overall Rules, Versus α

FIGURE 4



Density Function of Uniform Rule

FIGURE 5A

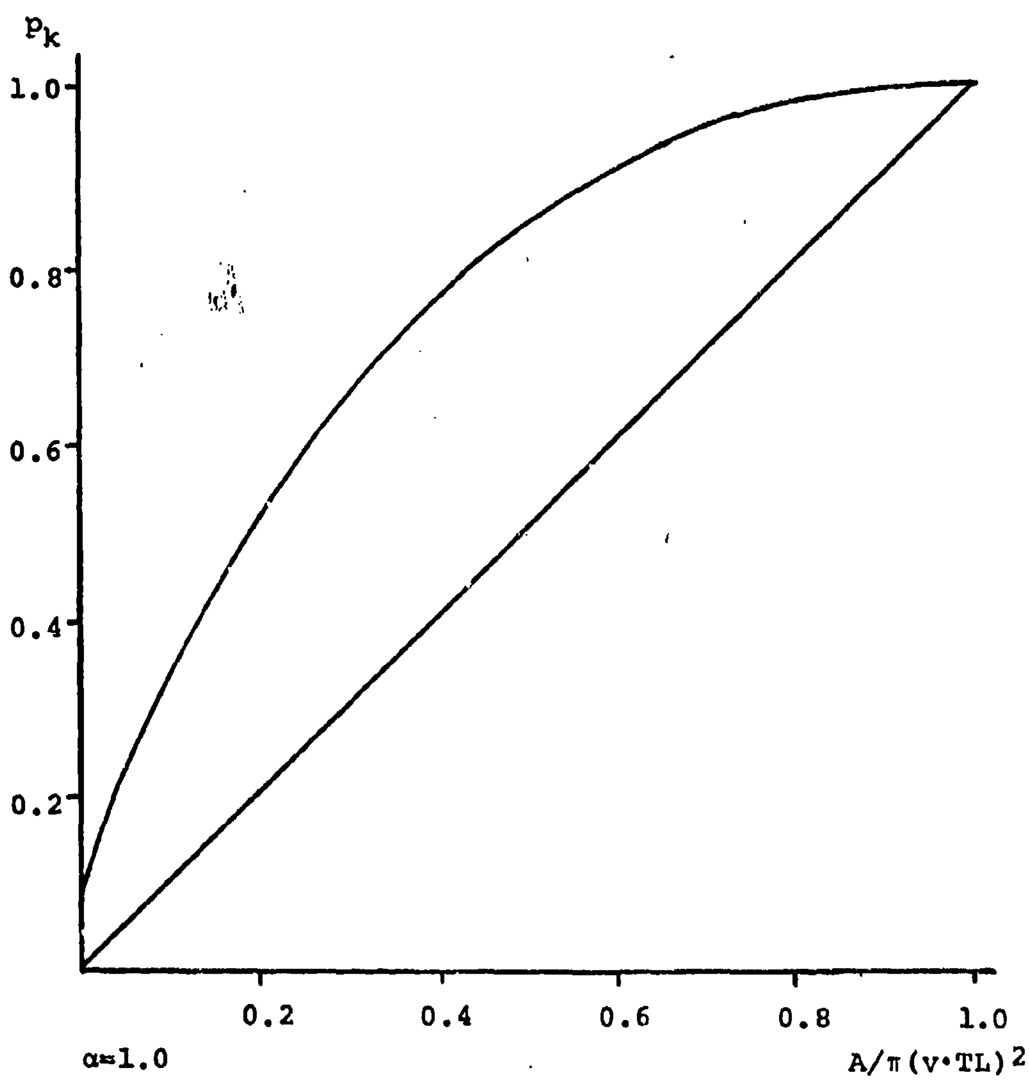
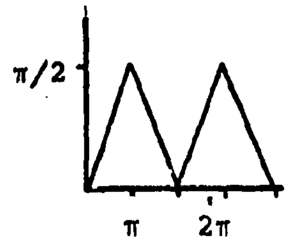


FIGURE 5B



Density Function of Modified Left-Right Rule

FIGURE 6A

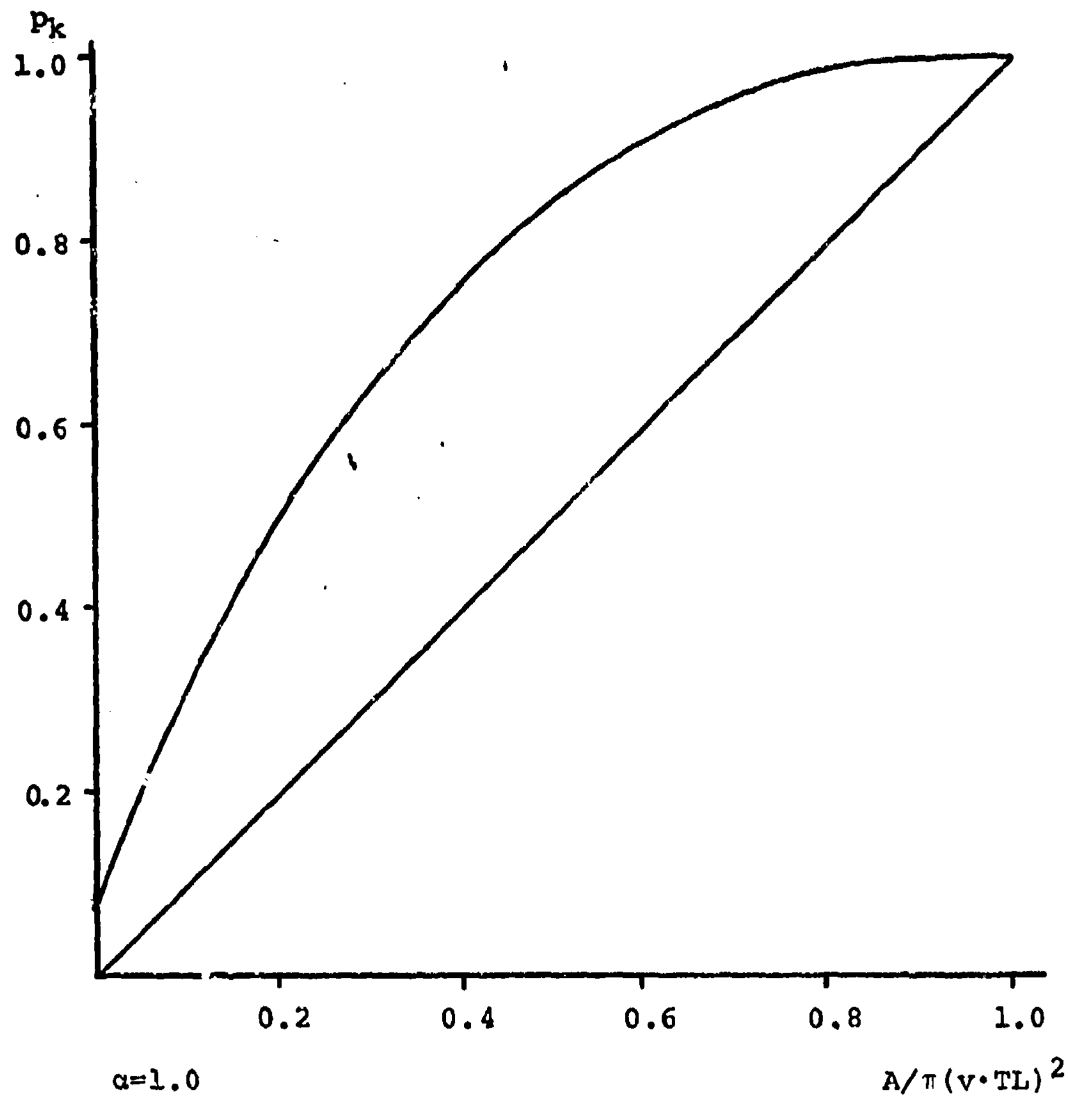
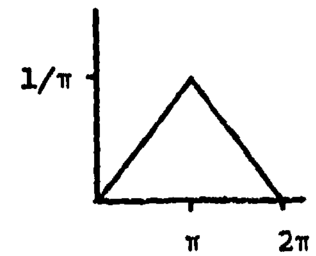


FIGURE 6B



Density Function of Reverse Course Rule

FIGURE 7A

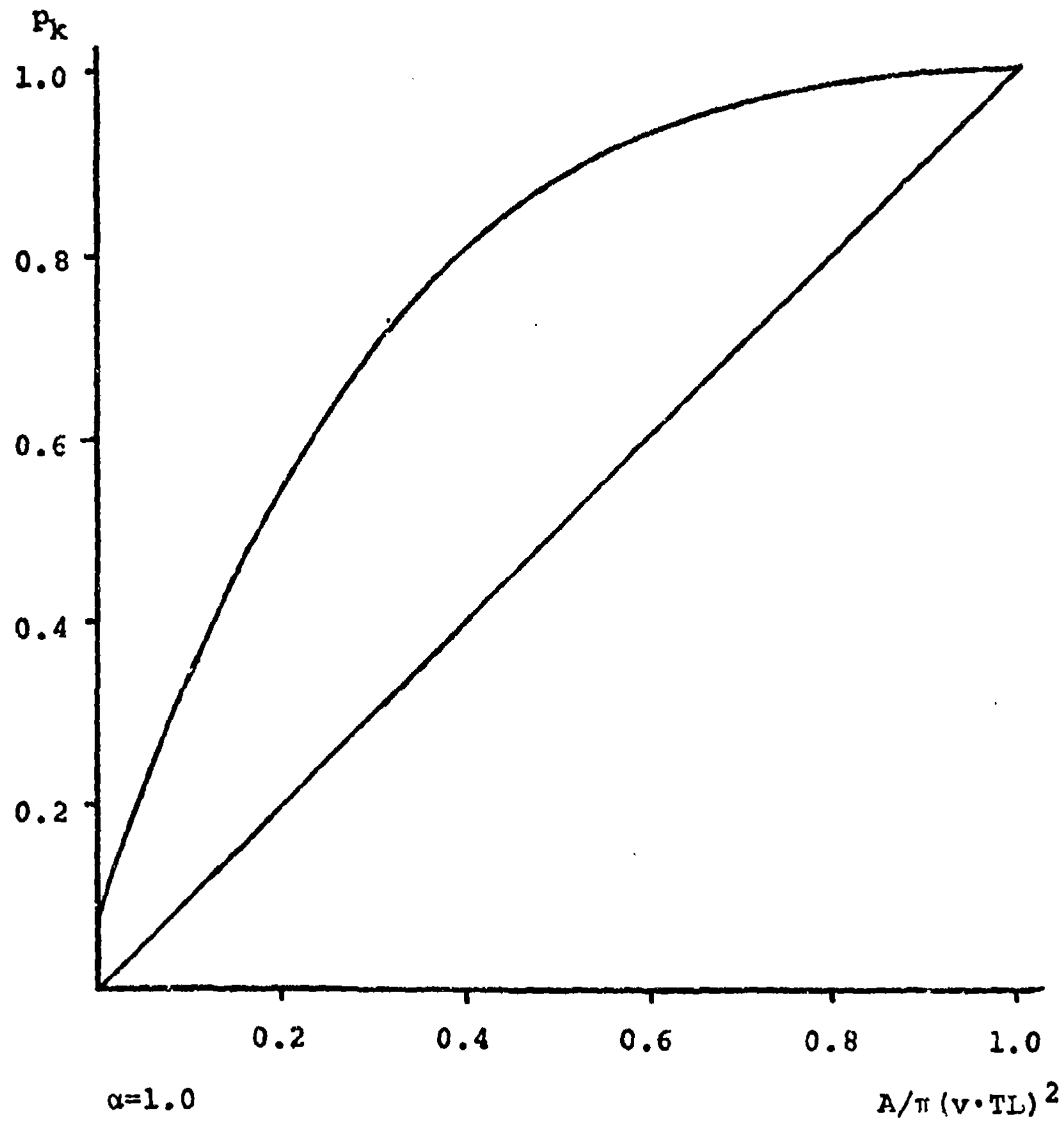


FIGURE 7B

Probability [Course Change = $\pi/2$] = 0.5

Probability [Course Change = $3\pi/2$] = 0.5

Density Function of Left-Right Rule

FIGURE 8A

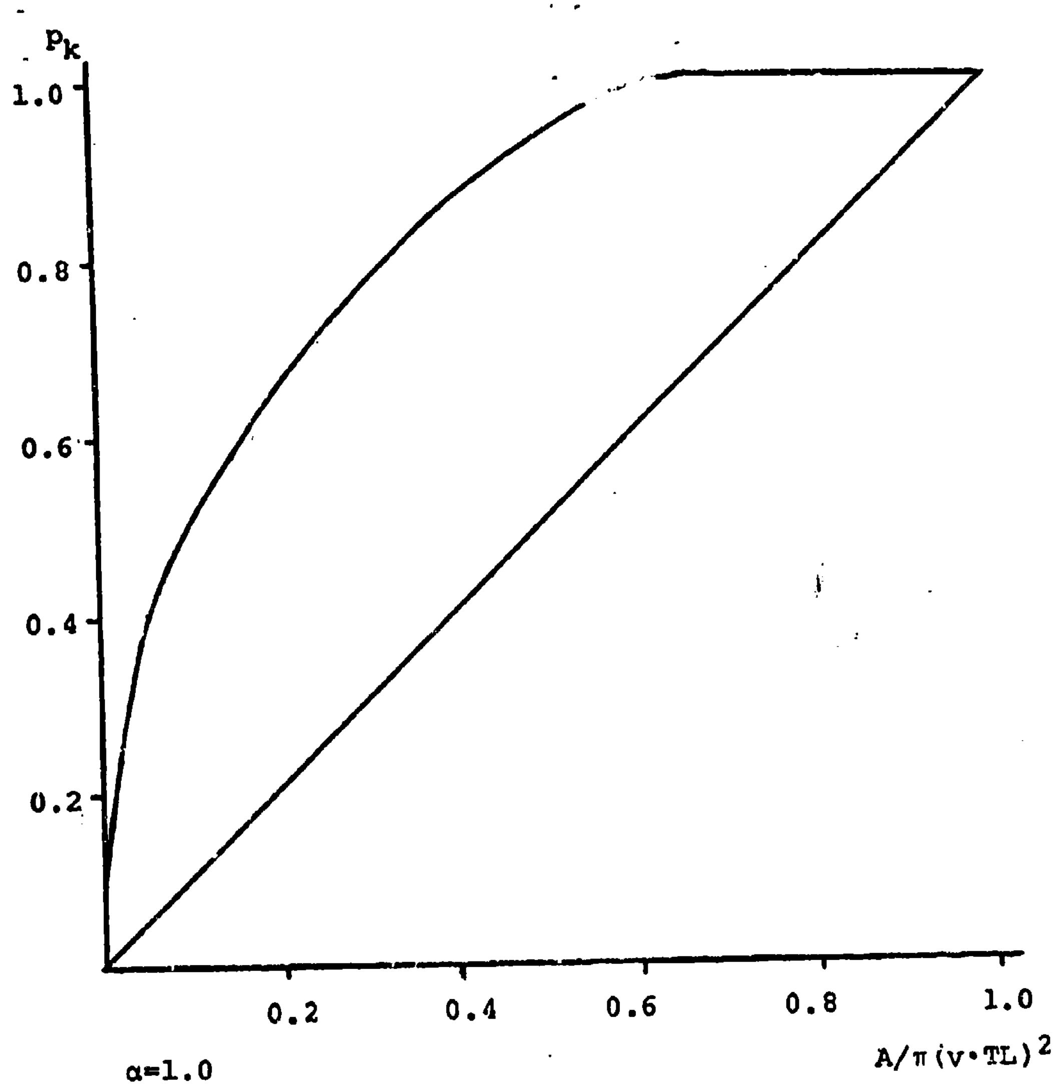
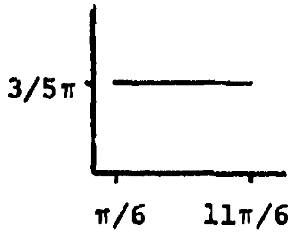


FIGURE 8B



Density Function of Truncated Uniform Rule

FIGURE 9A

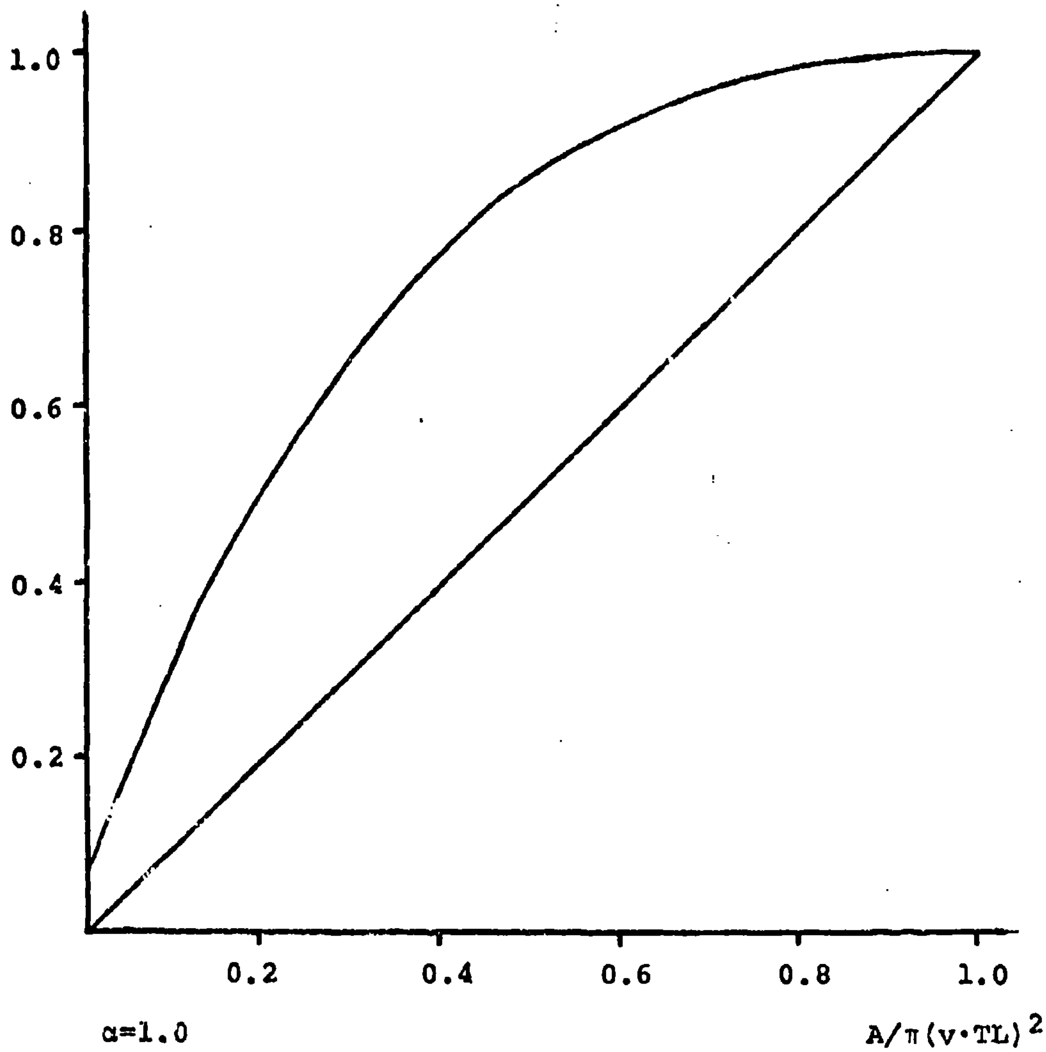
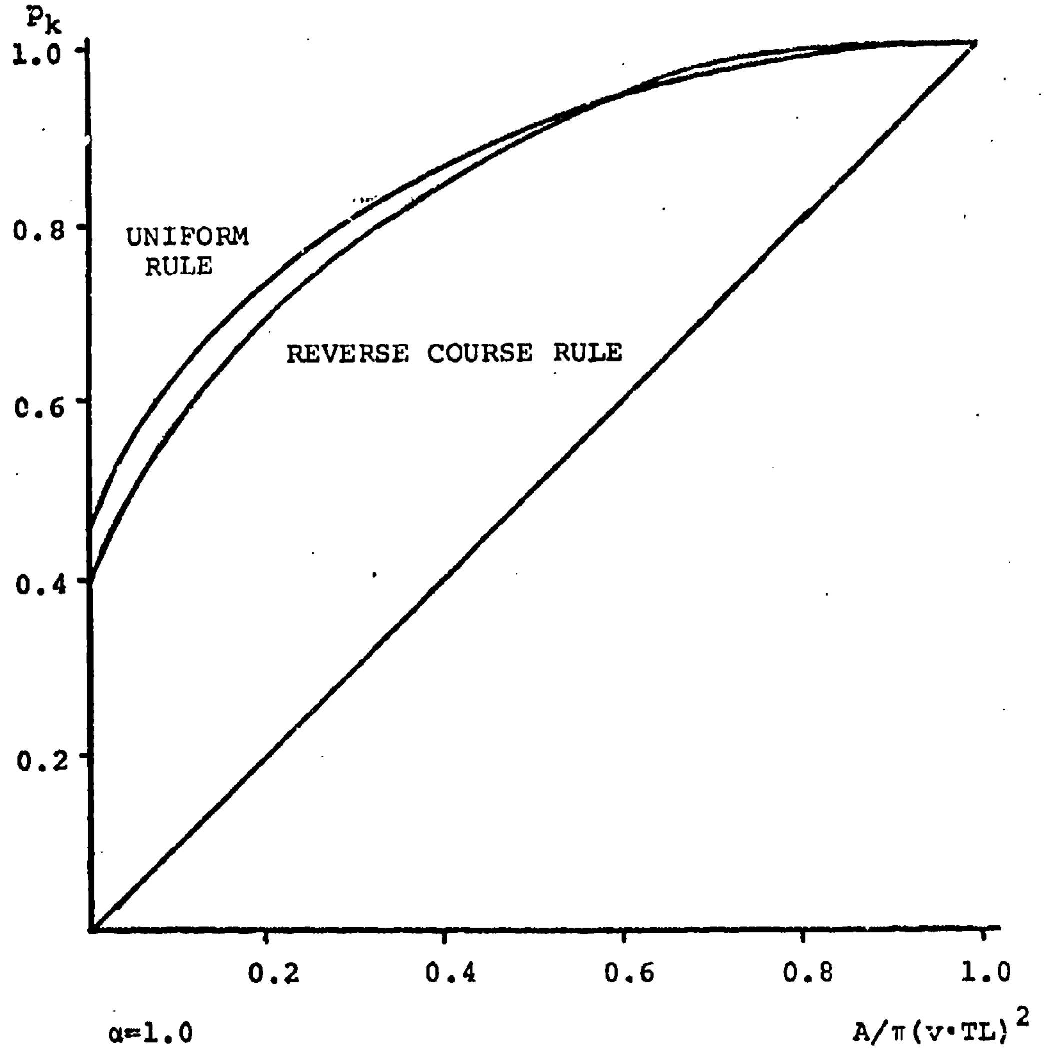


FIGURE 9B



A Comparison of the Tradeoffs Between the Uniform and Reverse Course Rules As a Function of Attacker Strength

FIGURE 10

IV. SIMULATION MODEL

Let E's position at time t be at X=0, Y=0 of a cartesian coordinate system. Then rotate the axis until the positive abscissa is aligned on and pointing in E's last direction of travel. Then regardless of the course rule used by E there is at least exp(-α) probability that he will be at X=v·TL, Y=0 at time t+TL. This is the probability that E does not change course during TL. So the attacker's probability of killing E by centering a small portion of his lethal area at (v·TL,0) is exp(-α). Then add to exp(-α) the integral of E's density function over that part of the uncertainty circle where the rest of the lethal area is targeted to determine the total p_k. The total amount of lethal area allocated in this manner is the numerator, A, of the attacker's strength function S.

A computer program to simulate the above computation was written in FORTRAN and run on the IBM 360. The program consisted of two phases, first the play of a strategy and secondly the scoring of that play.

A. SIMULATION OF A STRATEGY

To create a single play the evader's track was simulated from an initial position at time t to the resultant position at t+TL. The track was the result of a specific maneuvering strategy being simulated. A strategy was made up of two decision rules. The first rule determined the times between course changes which were exponential random variables with mean ET=1/λ. This rule was common throughout all of the simulations, although the parameter ET was a variable. The second rule determined the magnitude of subsequent course changes. The only other kinematic restriction was a constant speed, v, for the evader.

To start a play datum was initialized by setting $t=0$, $X(0)=0$, $Y(0)=0$ and an initial course, c_1, was selected from the uniform distribution. Then a sequence of exponential times $[t_i]_{i=1,N(TL)}$[6] were generated and a sequence of course changes, $[c_i]_{i=2,N(TL)}$, were generated using the course change rule. The evader's position at $t=TL$ was

$$X(TL) = \sum_{i=1}^{N-1} v \cdot t_i \cdot (\mathrm{Cos}(c_i) - \mathrm{Cos}(c_N)) + v \cdot TL \cdot \mathrm{Cos}(c_N) \qquad 1.1$$

$$Y(TL) = \sum_{i=1}^{N-1} v \cdot t_i \cdot (\mathrm{Sin}(c_i) - \mathrm{Sin}(c_N)) + v \cdot TL \cdot \mathrm{Sin}(c_N) \qquad 1.2$$

B. SCORING THE PLAY

Once E's position at $t=TL$ was determined that observation was scored. To score a play the coordinates of E's position, $X(TL)$, $Y(TL)$, had to be transformed. The purpose of the transformation was to make the observed position independent of the particular initial cource c_1. The transformation was a rotation of the coordinate axis about the datum so the positive abscissa would be aligned in the direction of the initial course. The transformed position was:

---

[6] The number of elements of this sequence, N(TL), is a Poisson random variable.

$$X'(TL) = X(TL)\cos(c_1) + Y(TL)\sin(c_1) \qquad 2.1$$

$$Y'(TL) = Y(TL)\cos(c_1) - X(TL)\sin(c_1) \qquad 2.2$$

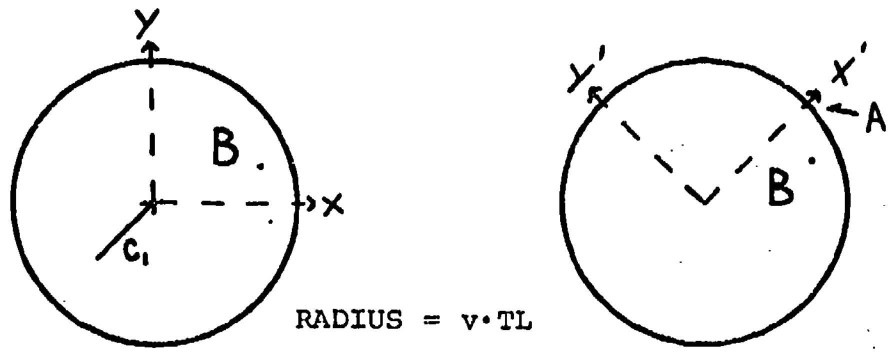
An example of this transformation is shown in Fig. 11.

If E had not made a course change during TL then his transformed position would have been

$$X'(TL) = v \cdot TL \qquad 3.1$$

$$Y'(TL) = 0 \qquad 3.2$$

A grid system of square cells[7] was placed over the playing area of Fig. 11B and a determination was made as to



ORIGIN: Datum

B: X(TL), Y(TL)

C_1: Initial Course

SAMPLE PLAY PRIOR TO TRANSFORMATION

FIGURE 11A

ORIGIN: Datum

B: X'(TL), Y'(TL)

A: (v•TL,0)

SAMPLE PLAY AFTER THE TRANSFORMATION

FIGURE 11B

which cell X'(TL), Y'(TL) was in. Each cell of the grid had an associated value which would represent the number of times a play resulted in an observation in that cell. When

[7] The cumulative cell area was normalized by the factor $\pi(v \cdot TL)^2$.

the appropriate cell was determined, for the play being scored, that value was incremented by one.

A simulation run was composed of 16810 plays and scoring iterations for a strategy utilizing a specific course change rule. The input variables for a run, besides the course rule were v, TL and ET. At the end of a simulation run the grid system was a two dimensional histogram of E's position. The frequencies in the histogram were then ordered, accumulated and normalized to achieve cumulative cell probabilities. The ordering corresponded to the conservative assumption that P could divide his lethal area and target only those cells with the higher probabilities. These probabilities were then plotted against the cumlative cell area they represented.[8] This graph was labeled p_k versus attacker strength and was the primary output of the program.

Thirty simulation runs were made to investigate five different course change rules and six different values of ET. For all runs the following constant values were maintained, v=5 knots and TL=2 hours. The graphs from those runs are included in the Computer Output section. Each course change rule was simulated many different times, the only difference between runs being the pseudo random number generator seeds. This was done to check for the variability of the p_k graph for that rule. In all such runs

---

[8] The amount of area in a cell was denoted as cell size and equal to $(2 \cdot TL/41)^2$. TL and v were held constant therefore cell size was always approximately 1/4 nm^2.

the resultant graphs were so similar that any difference was not distinguishable. For simulation run required two minutes and forty-five seconds using 125 K on the IBM 360.

V. CONCLUSION

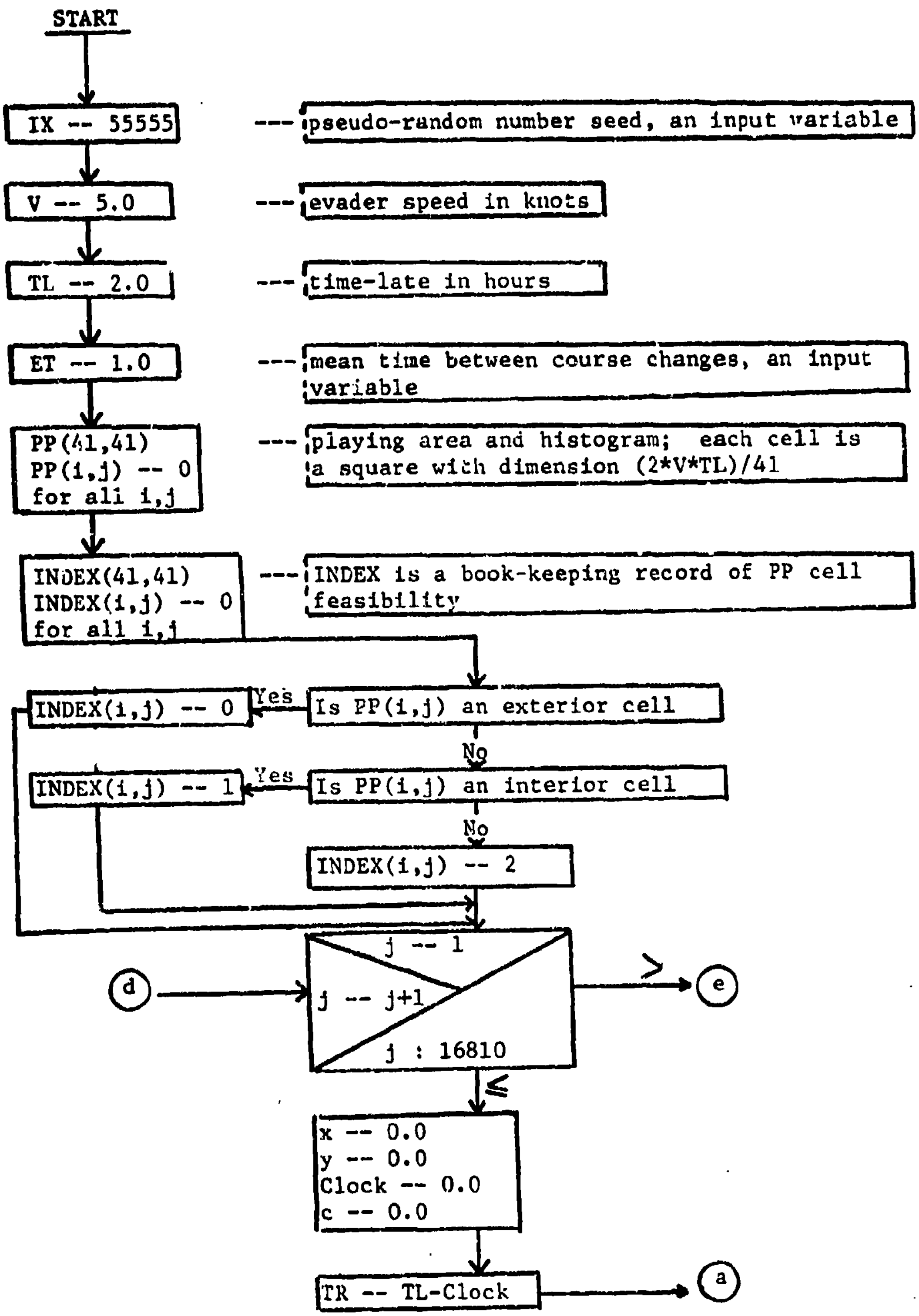
It was determined that the uniform course change rule is not optimal within the Poisson class of strategies. Comparison of various rules, see Fig. 10, showed that under certain conditions a single rule such as the reverse course change rule is better. Also the new lower bound on p_k, taken over all rules and values of α simulated, is an improvement on the lower bound achieved from only the uniform rule.

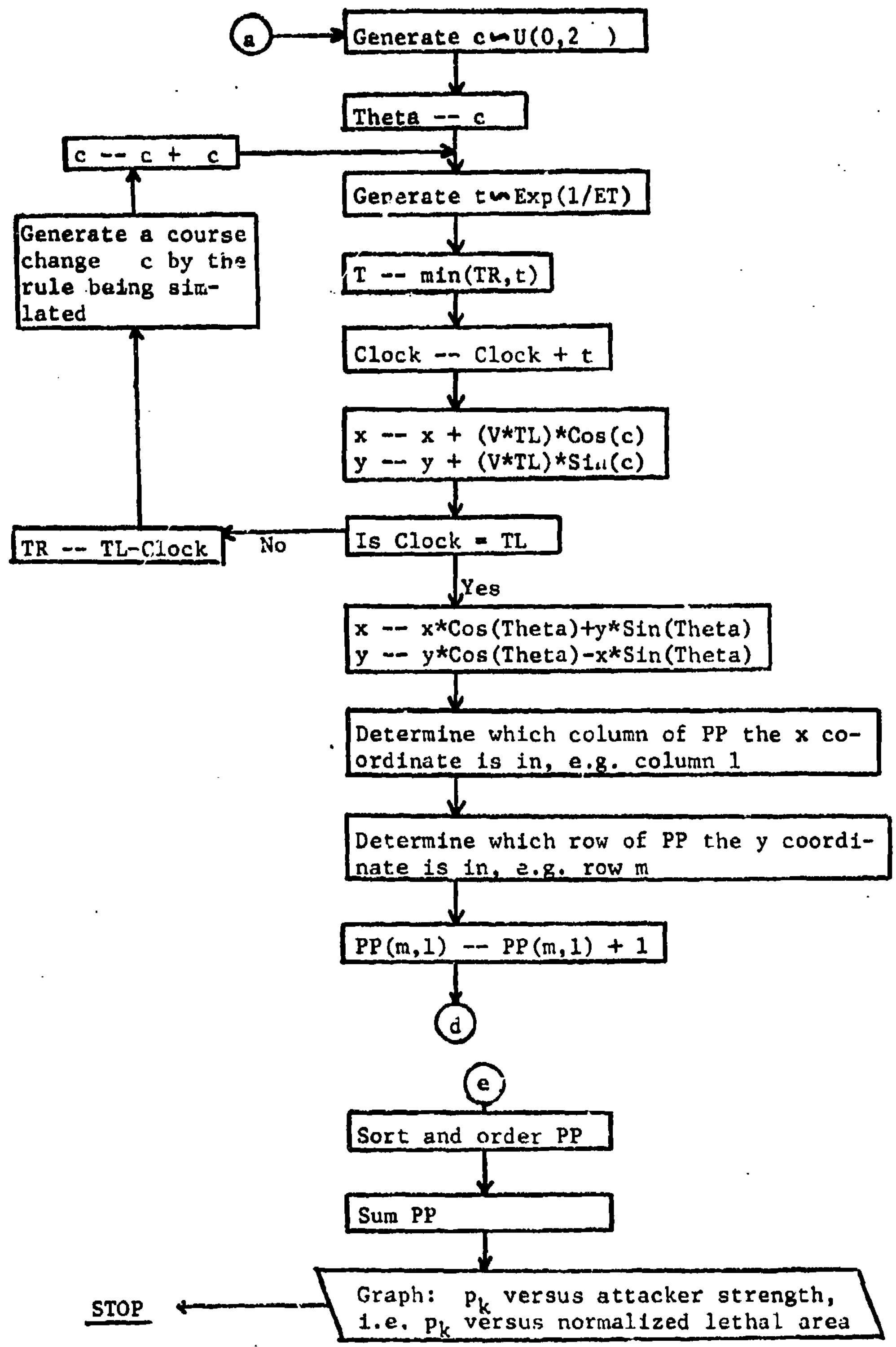
Certainly not all the possible course change rules were simulated. The rules evaluated, however, were representative of the broad class of possible rules. When compared to the uniform rule, all but one of the other rules showed that E could improve his situation if he knew P's strength by selecting the better rule for that encounter. The one rule that was consistently dominated was the left-right rule.

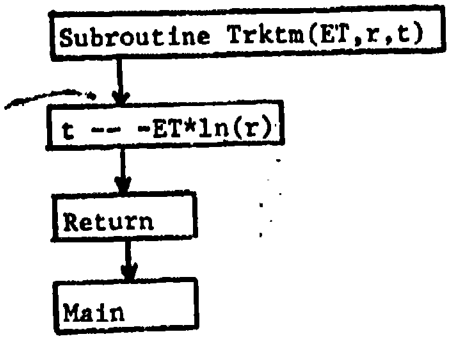
It remains unknown whether or not the optimal strategy is a Poisson strategy.

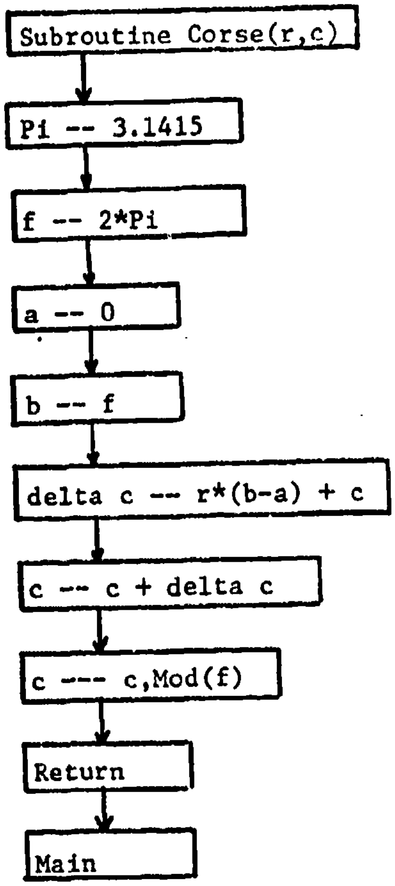
APPENDIX A

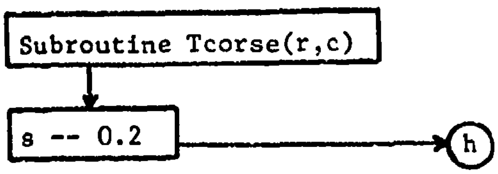
FLOW DIAGRAM

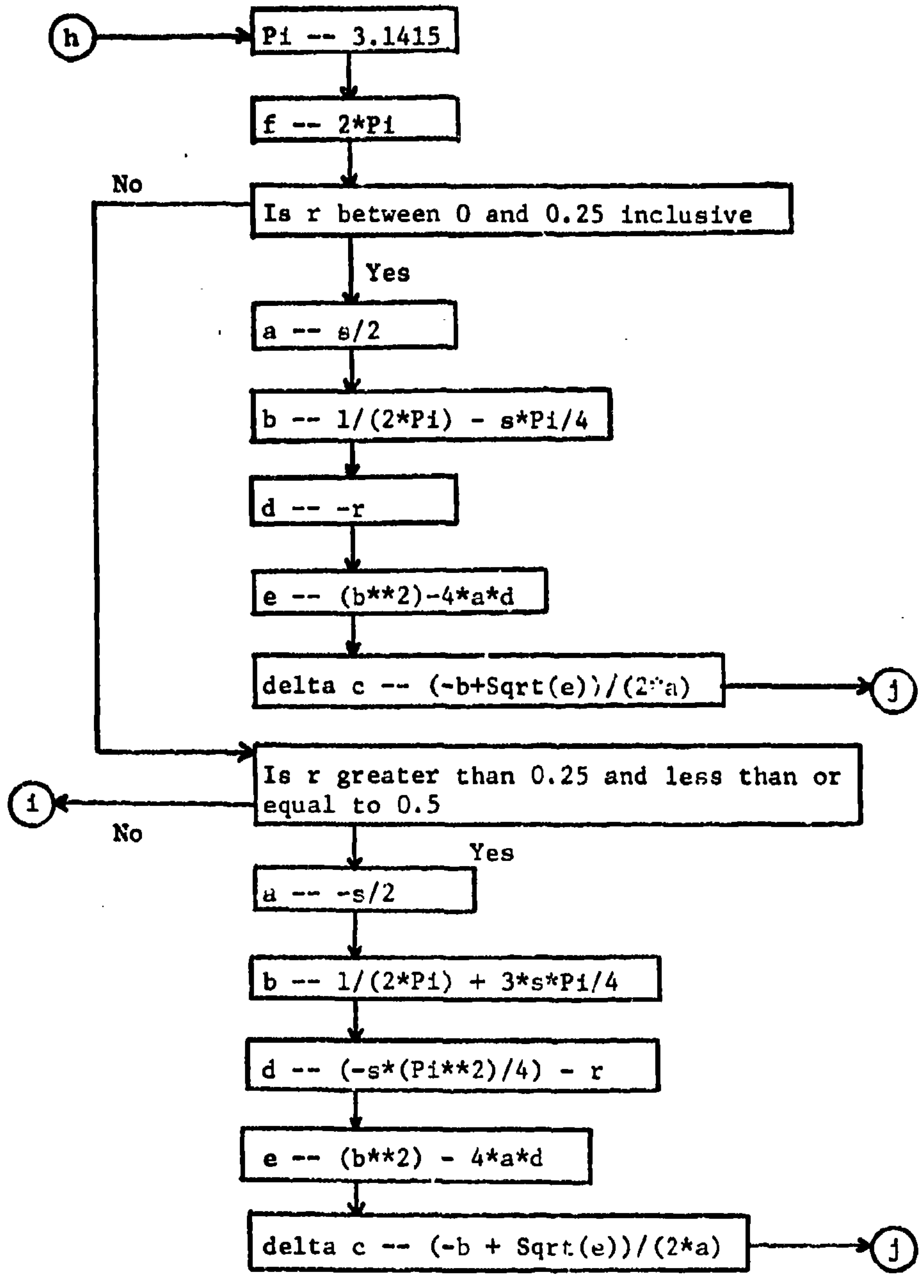


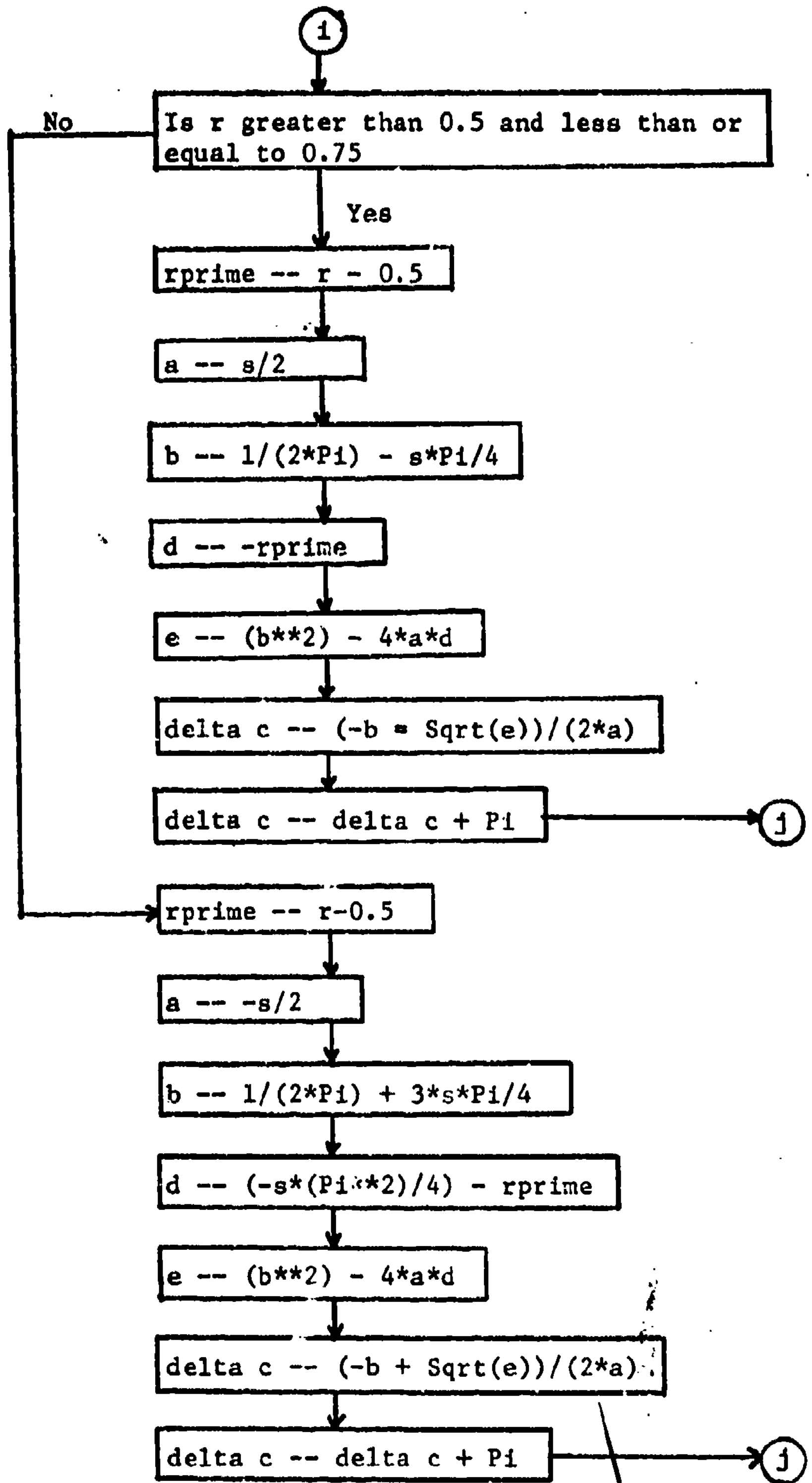

a
Generate c⤳U(0,2)
Theta -- c
c -- c + c
Generate t⤳Exp(1/ET)
Generate a course change c by the rule being simlated
T -- min(TR,t)
Clock -- Clock + t
x -- x + (V*TL)*Cos(c)
y -- y + (V*TL)*Sin(c)
TR -- TL-Clock
No
Is Clock = TL
Yes
x -- x*Cos(Theta)+y*Sin(Theta)
y -- y*Cos(Theta)-x*Sin(Theta)
Determine which column of PP the x coordinate is in, e.g. column 1
Determine which row of PP the y coordinate is in, e.g. row m
PP(m,1) -- PP(m,1) + 1
d
e
Sort and order PP
Sum PP
Graph: pk versus attacker strength, i.e. pk versus normalized lethal area
STOP


Subroutine Trktm(ET,r,t)
t -- -ET*ln(r)
Return
Main

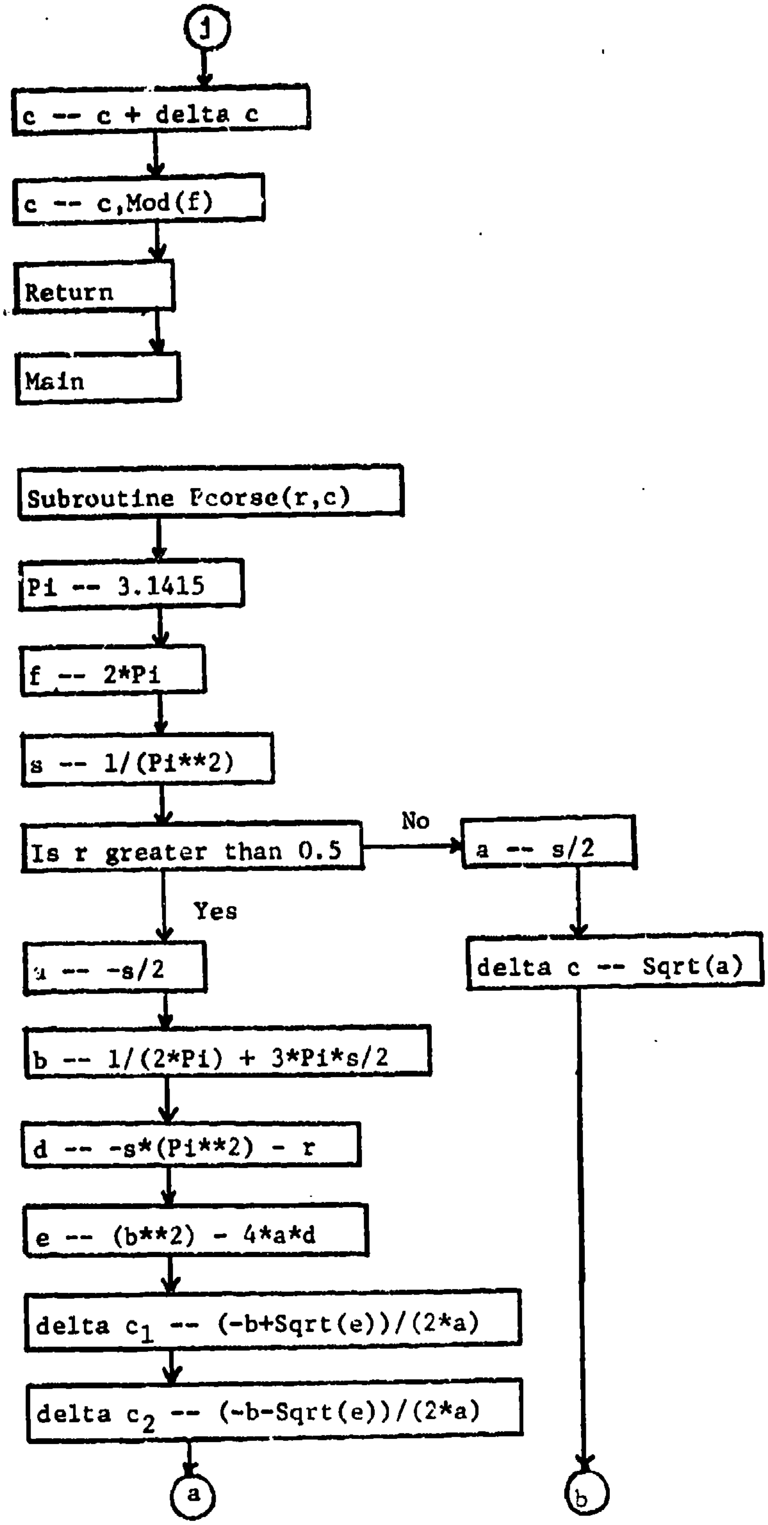

Subroutine Corse(r,c)
Pi -- 3.1415
f -- 2*Pi
a -- 0
b -- f
delta c -- r*(b-a) + c
c -- c + delta c
c --- c,Mod(f)
Return
Main

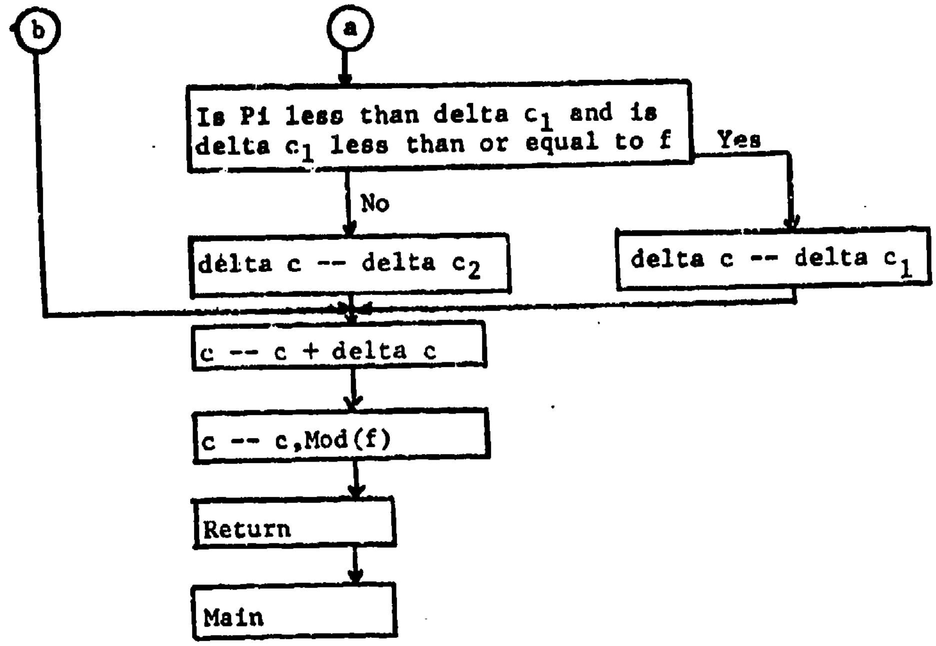

Subroutine Tcorse(r,c)
s -- 0.2
h

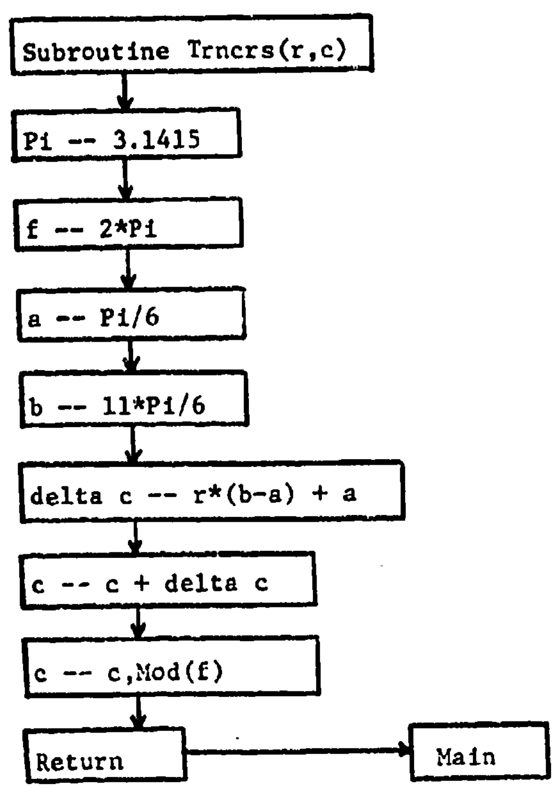

h
Pi -- 3.1415
f -- 2*Pi
Is r between 0 and 0.25 inclusive
No
Yes
a -- s/2
b -- 1/(2*Pi) - s*Pi/4
d -- -r
e -- (b**2)-4*a*d
delta c -- (-b+Sqrt(e))/(2*a)
j
Is r greater than 0.25 and less than or equal to 0.5
i
No
Yes
a -- -s/2
b -- 1/(2*Pi) + 3*s*Pi/4
d -- (-s*(Pi**2)/4) - r
e -- (b**2) - 4*a*d
delta c -- (-b + Sqrt(e))/(2*a)
j

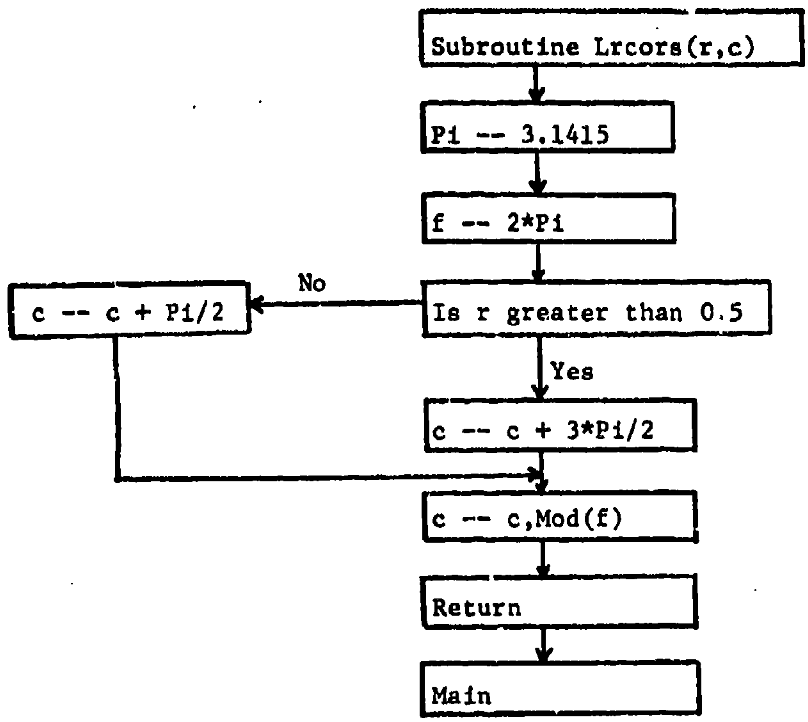


i
No
Is r greater than 0.5 and less than or equal to 0.75
Yes
rprime -- r - 0.5
a -- s/2
b -- 1/(2*Pi) - s*Pi/4
d -- -rprime
e -- (b**2) - 4*a*d
delta c -- (-b = Sqrt(e))/(2*a)
delta c -- delta c + Pi
j
rprime -- r-0.5
a -- -s/2
b -- 1/(2*Pi) + 3*s*Pi/4
d -- (-s*(Pi**2)/4) - rprime
e -- (b**2) - 4*a*d
delta c -- (-b + Sqrt(e))/(2*a)
delta c -- delta c + Pi
j


1
c -- c + delta c
c -- c,Mod(f)
Return
Main
Subroutine Pcorse(r,c)
Pi -- 3.1415
f -- 2*Pi
s -- 1/(Pi**2)
Is r greater than 0.5
No
a -- s/2
delta c -- Sqrt(a)
Yes
a -- -s/2
b -- 1/(2*Pi) + 3*Pi*s/2
d -- -s*(Pi**2) - r
e -- (b**2) - 4*a*d
delta c1 -- (-b+Sqrt(e))/(2*a)
delta c2 -- (-b-Sqrt(e))/(2*a)
a
b


b
a
Is Pi less than delta c_1 and is delta c_1 less than or equal to f
Yes
No
délta c -- delta c_2
delta c -- delta c_1
c -- c + delta c
c -- c,Mod(f)
Return
Main


Subroutine Trncrs(r,c)
Pi -- 3.1415
f -- 2*Pi
a -- Pi/6
b -- 11*Pi/6
delta c -- r*(b-a) + a
c -- c + delta c
c -- c,Mod(f)
Return
Main


Subroutine Lrcors(r,c)
Pi -- 3.1415
f -- 2*Pi
Is r greater than 0.5
No
c -- c + Pi/2
Yes
c -- c + 3*Pi/2
c -- c,Mod(f)
Return
Main

COMPUTER OUTPUT

TL/ET

Rule	1.0	2.0	3.0	4.0
Uniform	.729	.580	.501	.486
Modified Left-Right	.737	.580	.501	.486
Reverse Course	.690	.572	.548	.564
Left-Right	.917	.768	.650	.588
Truncated Uniform	.697	.556	.509	.517
column minimum	.690	.556	.501	.486

p_k for attacker strength equal 0.2

Table I

TL/ET

Rule	1.0	2.0	3.0	4.0
Uniform	.854	.784	.752	.760
Modified Left-Right	.878	.784	.752	.768
Reverse Course	.838	.791	.799	.831
Left-Right	.964	.909	.870	.854
Truncated Uniform	.846	.768	.768	.815
column minimum	.838	.768	.752	.760

p_k for attacker strength equal 0.4

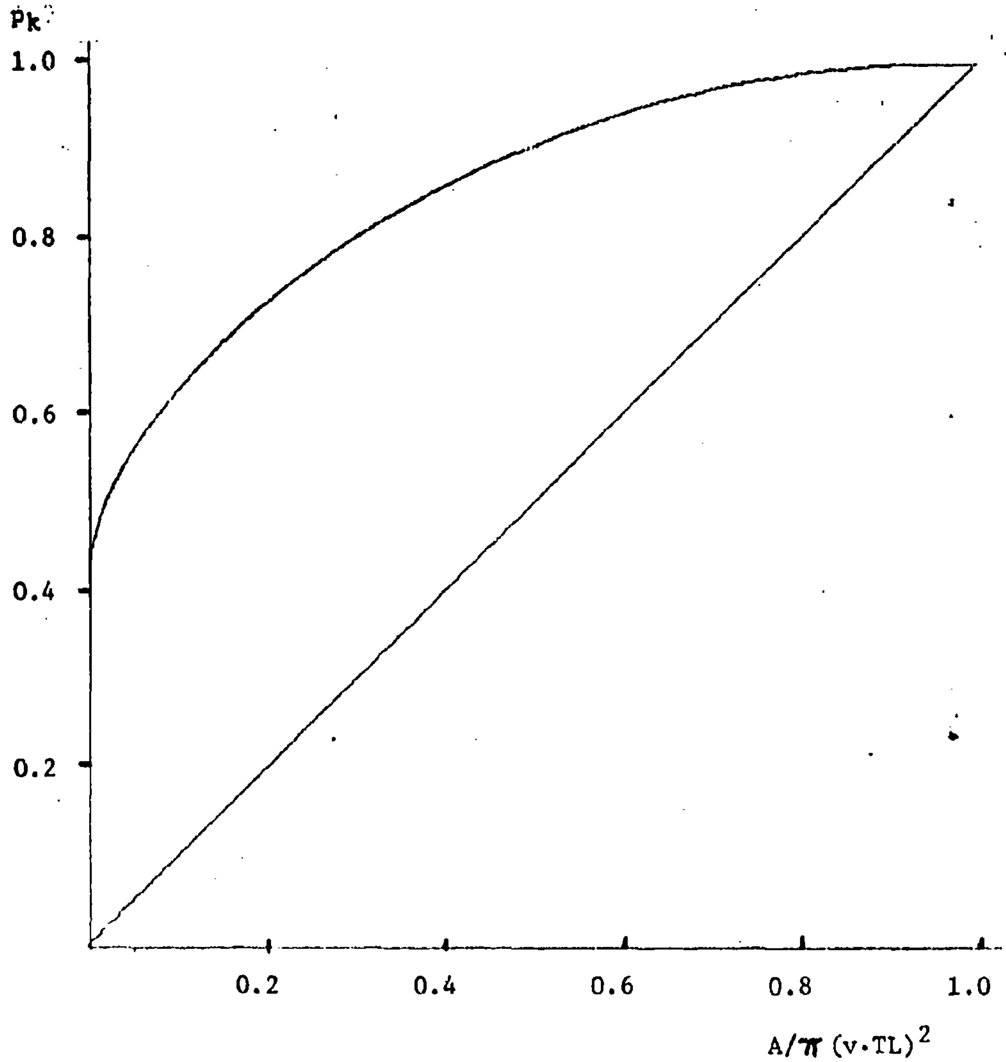
Table II

TL/ET

Rule	1.0	2.0	3.0	4.0
Uniform	.933	.909	.901	.917
Modified Left-Right	.948	.909	.901	.917
Reverse Course	.933	.917	.925	.948
Left-Right	.987	.987	.980	.987
Truncated Uniform	.933	.909	.909	.933
column minimum	.933	.909	.901	.917

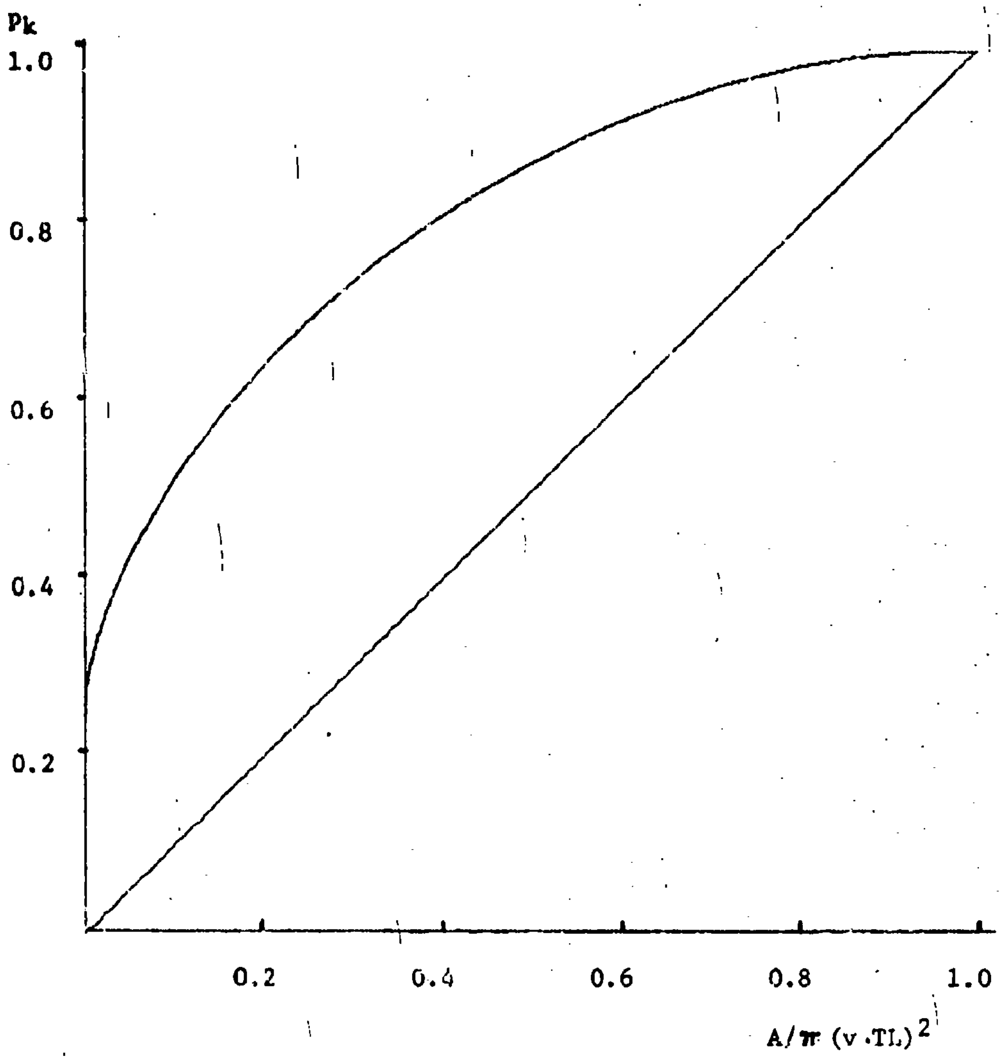
p_k for attacker strength equal 0.6

Table III



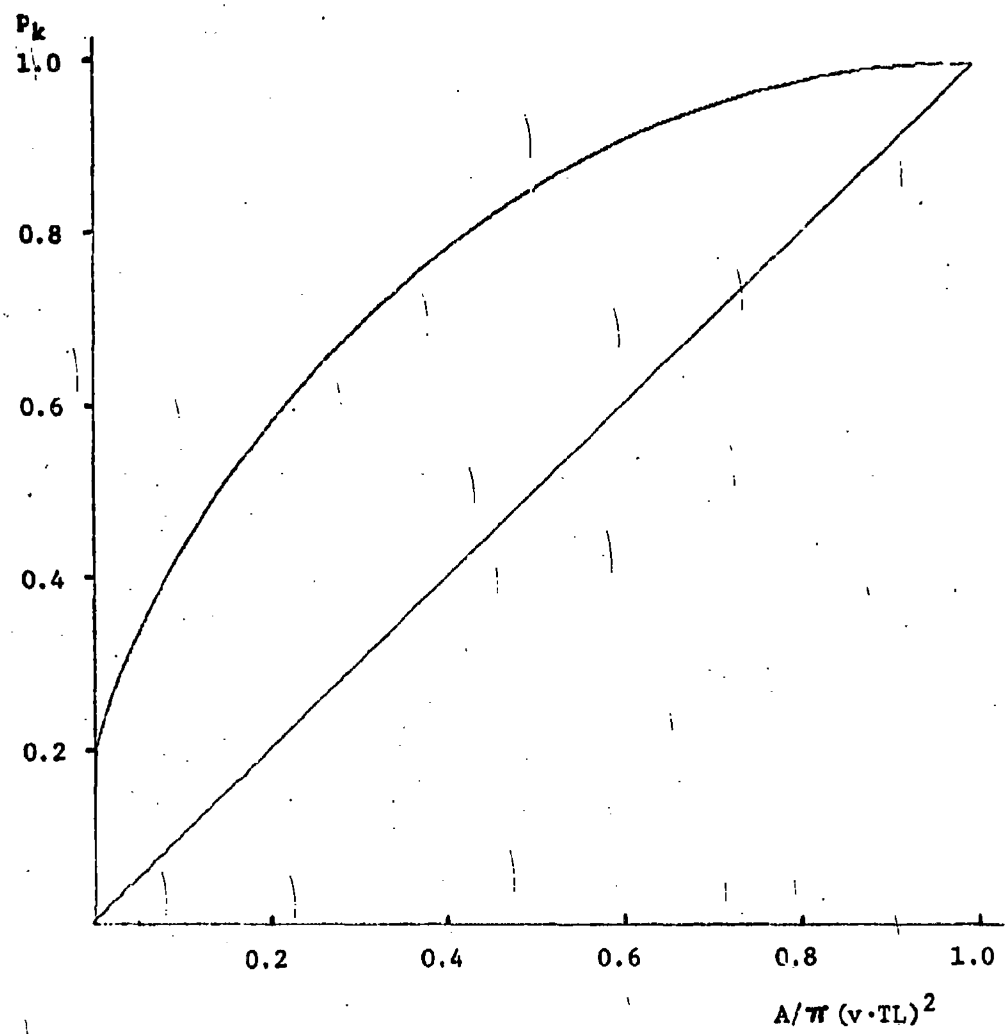
p_k versus Attacker Strength

Course Change Rule: Uniform ET = 2.0



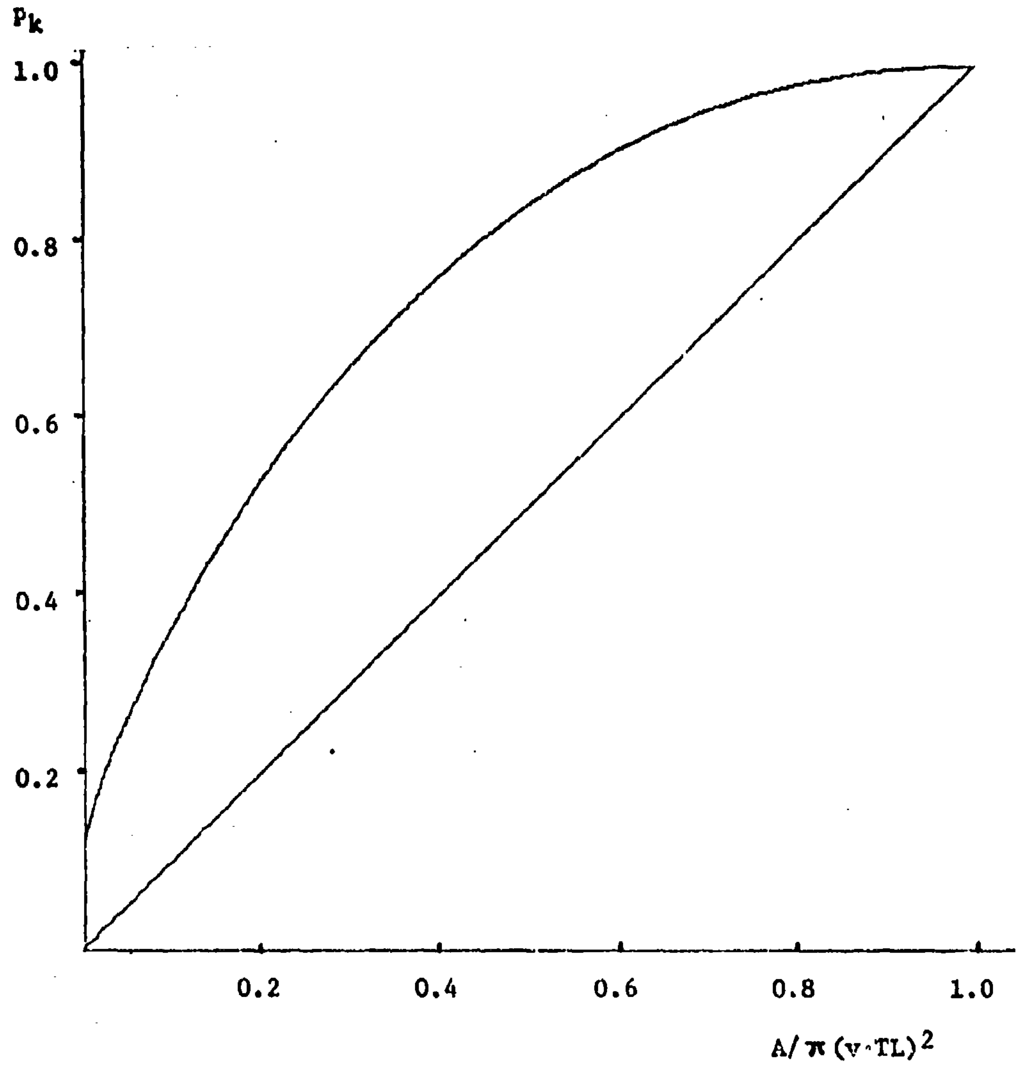
P_k versus Attacker Strength

Course Change Rule: Uniform ET = 1.33



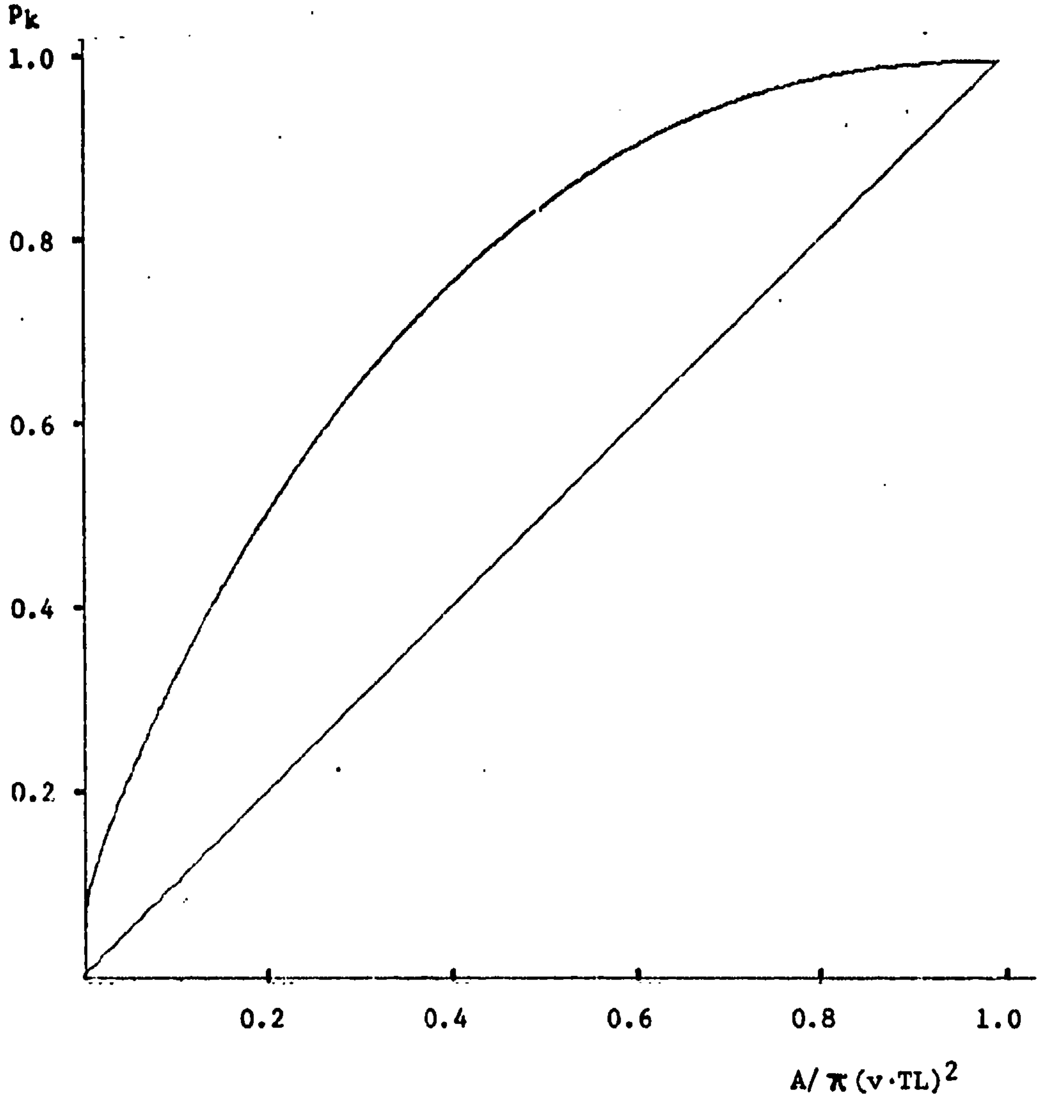
P_k versus Attacker Strength

Course Change Rule: Uniform ET = 1.0



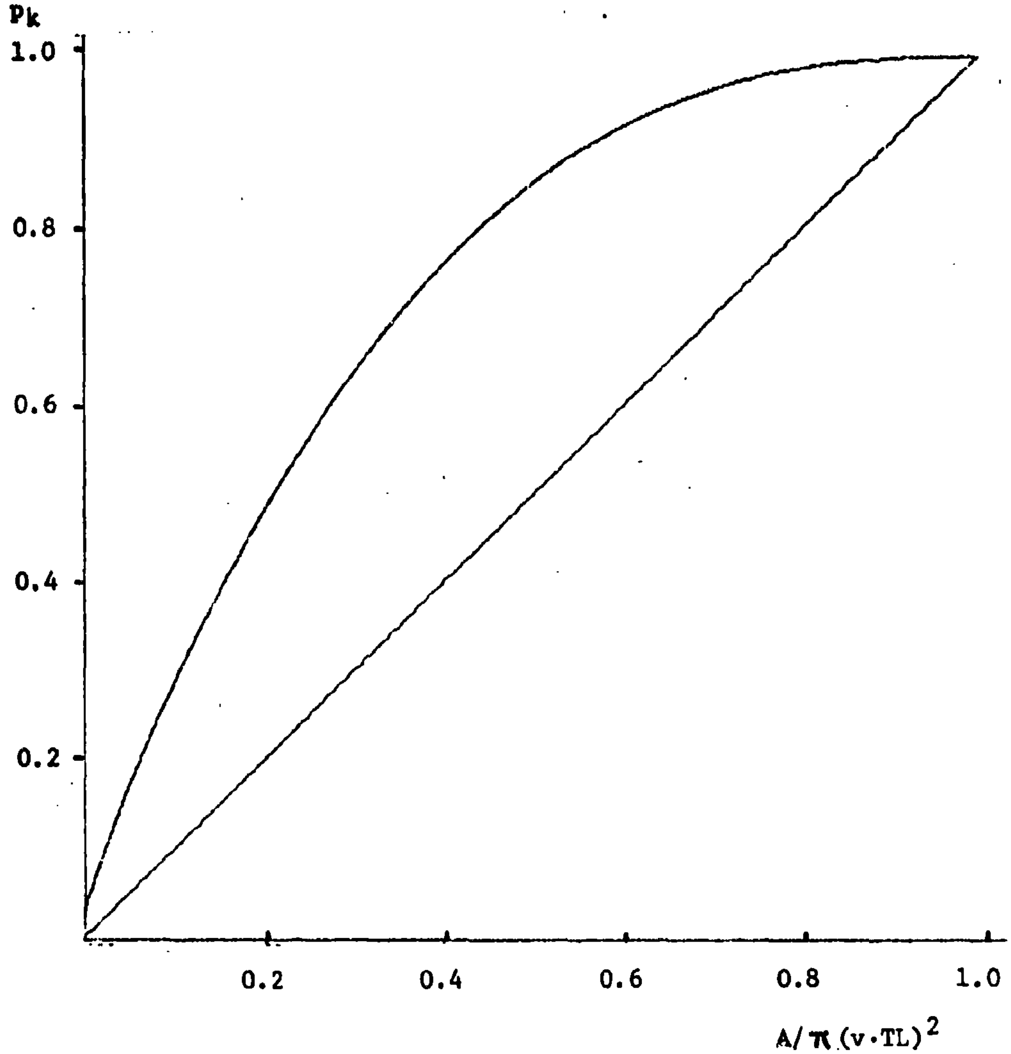
P_k versus Attacker Strength

Course Change Rule: Uniform ET = 0.8



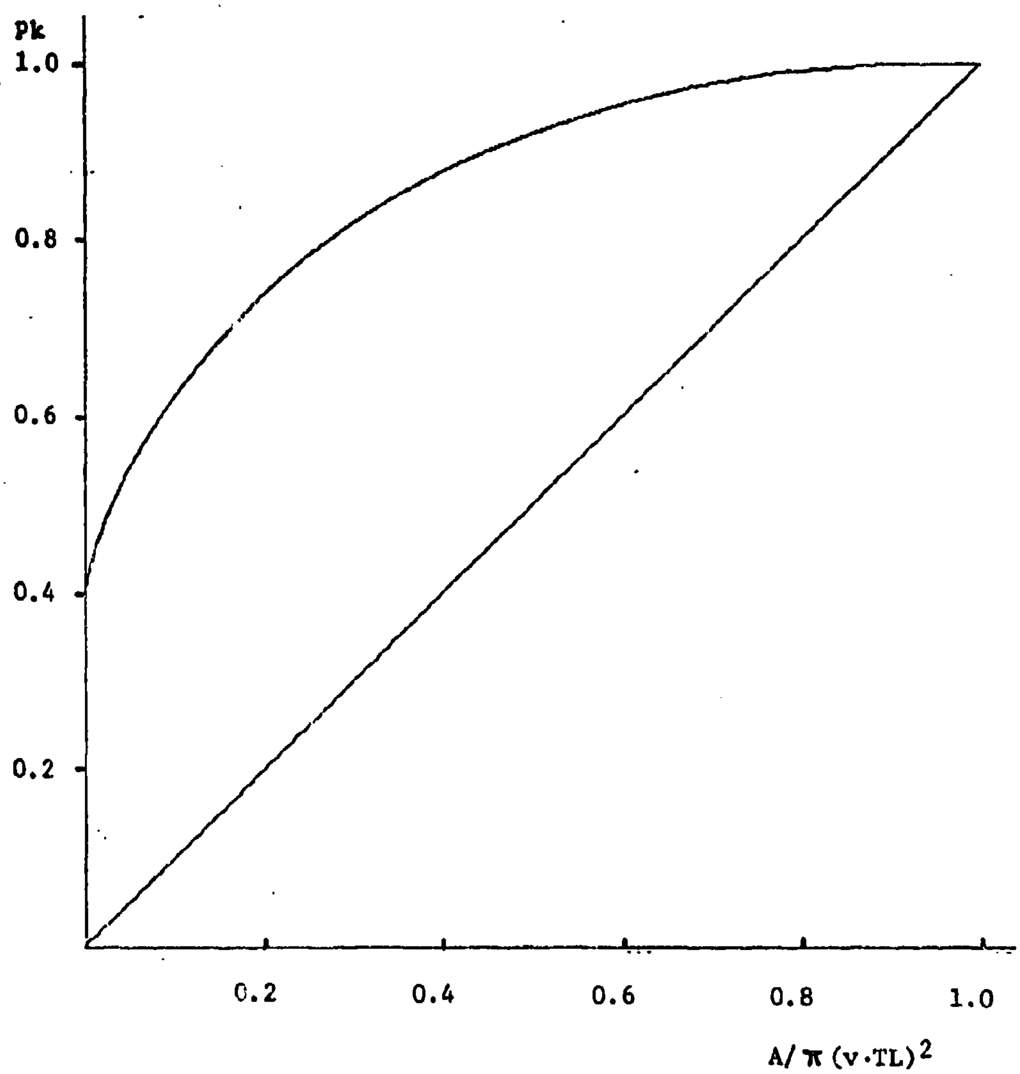
p_k versus Attacker Strength

Course Change Rule: Uniform ET = 0.66



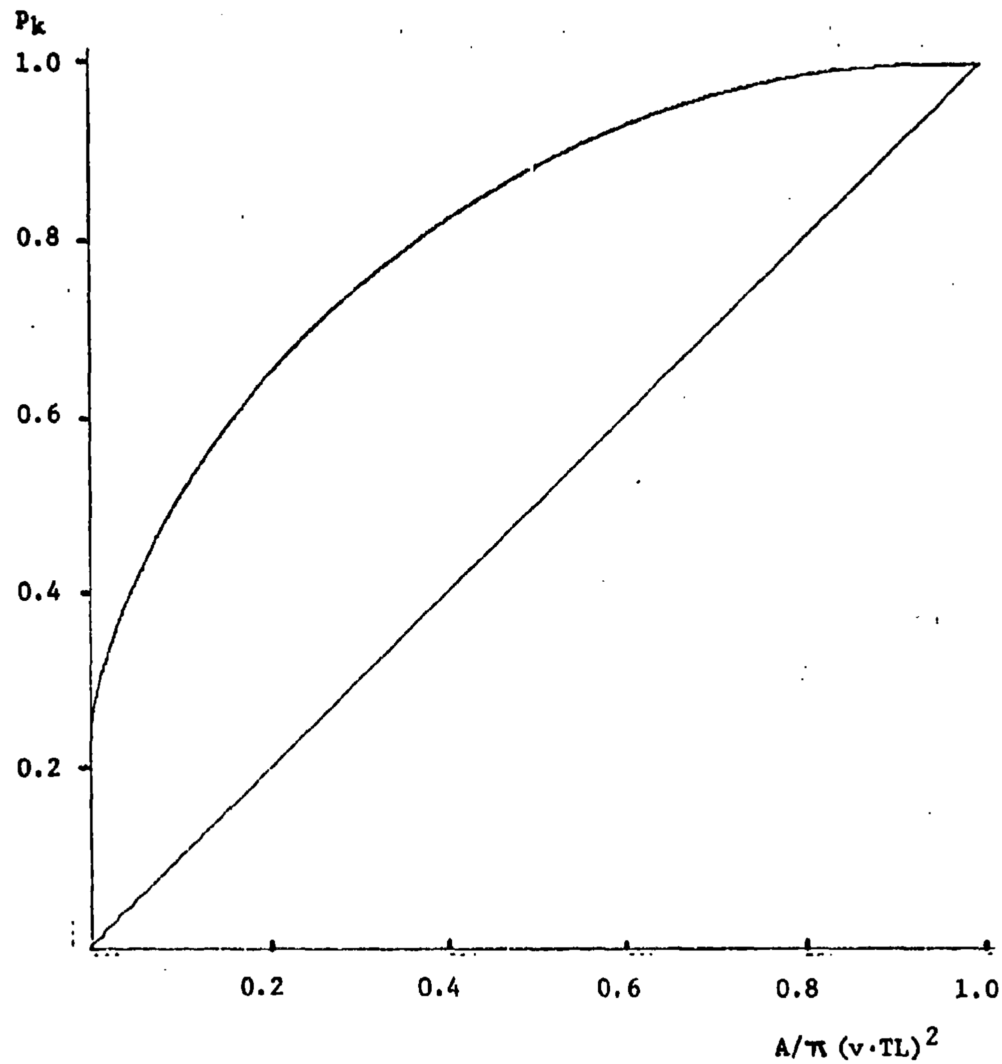
p_k versus Attacker Strength

Course Change Rule: Uniform ET = 0.5



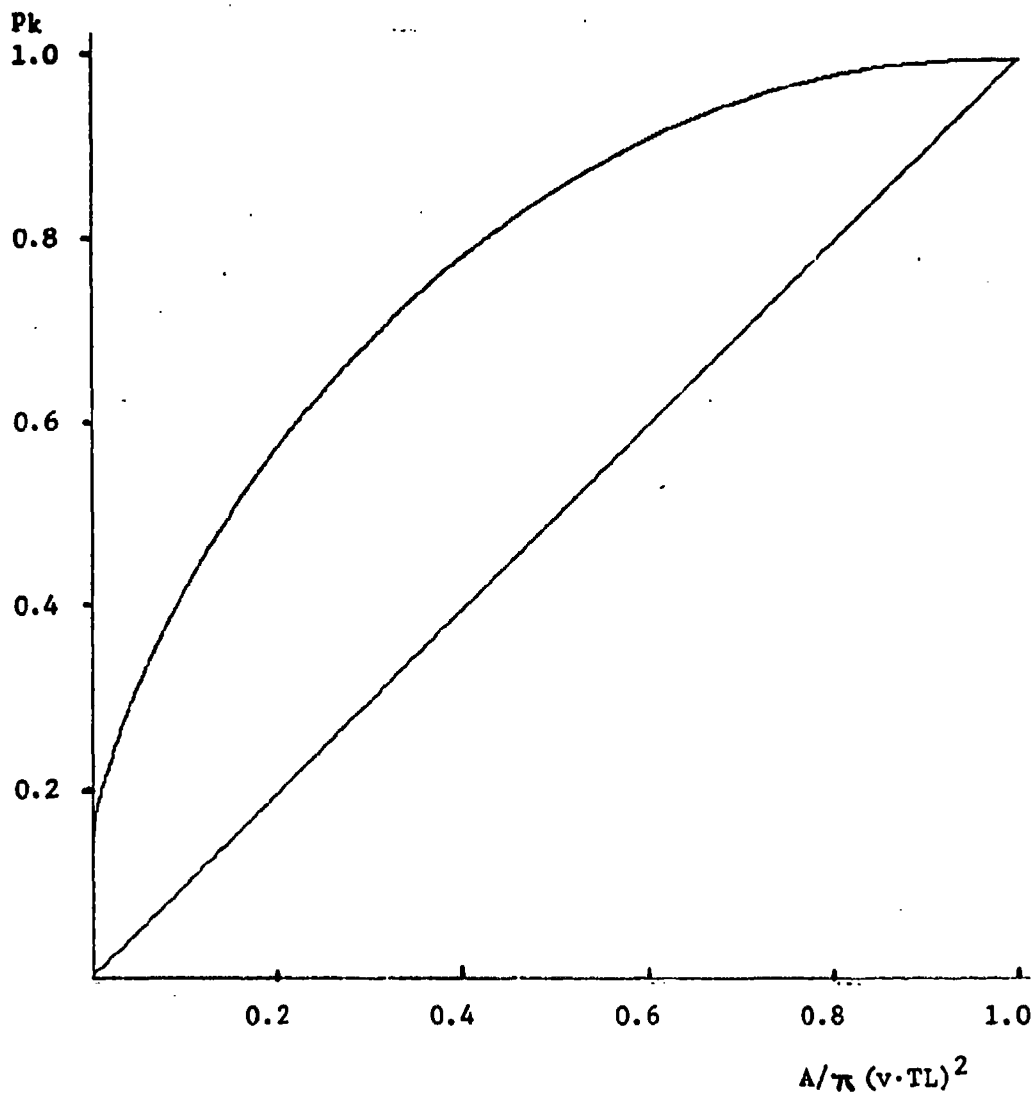
p_k versus Attacker Strength

Course Change Rule: Modified Left-Right ET = 2.0



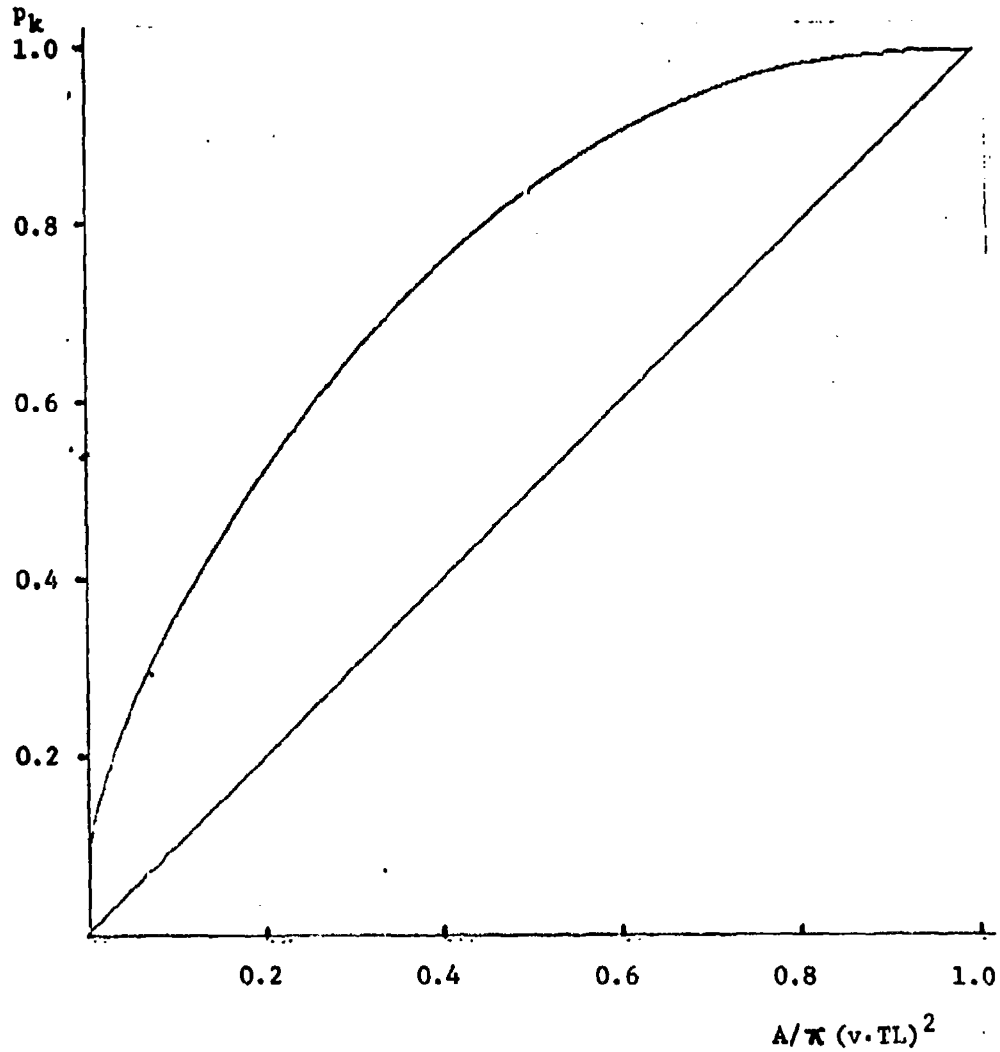
P_k versus Attacker Strength

Course Change Rule: Modified Left-Right ET = 1.33



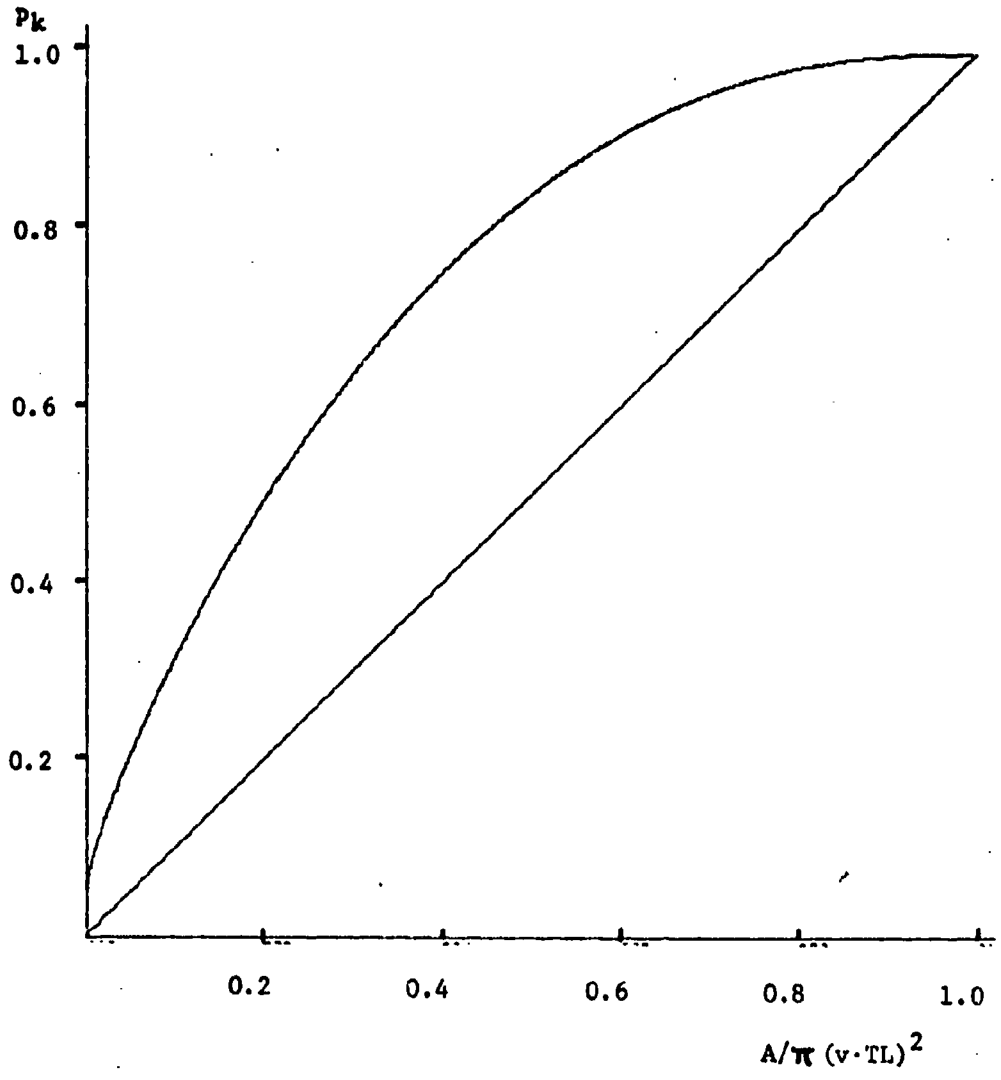
p_k versus Attacker Strength

Course Change Rule: Modified Left-Right ET = 1.0



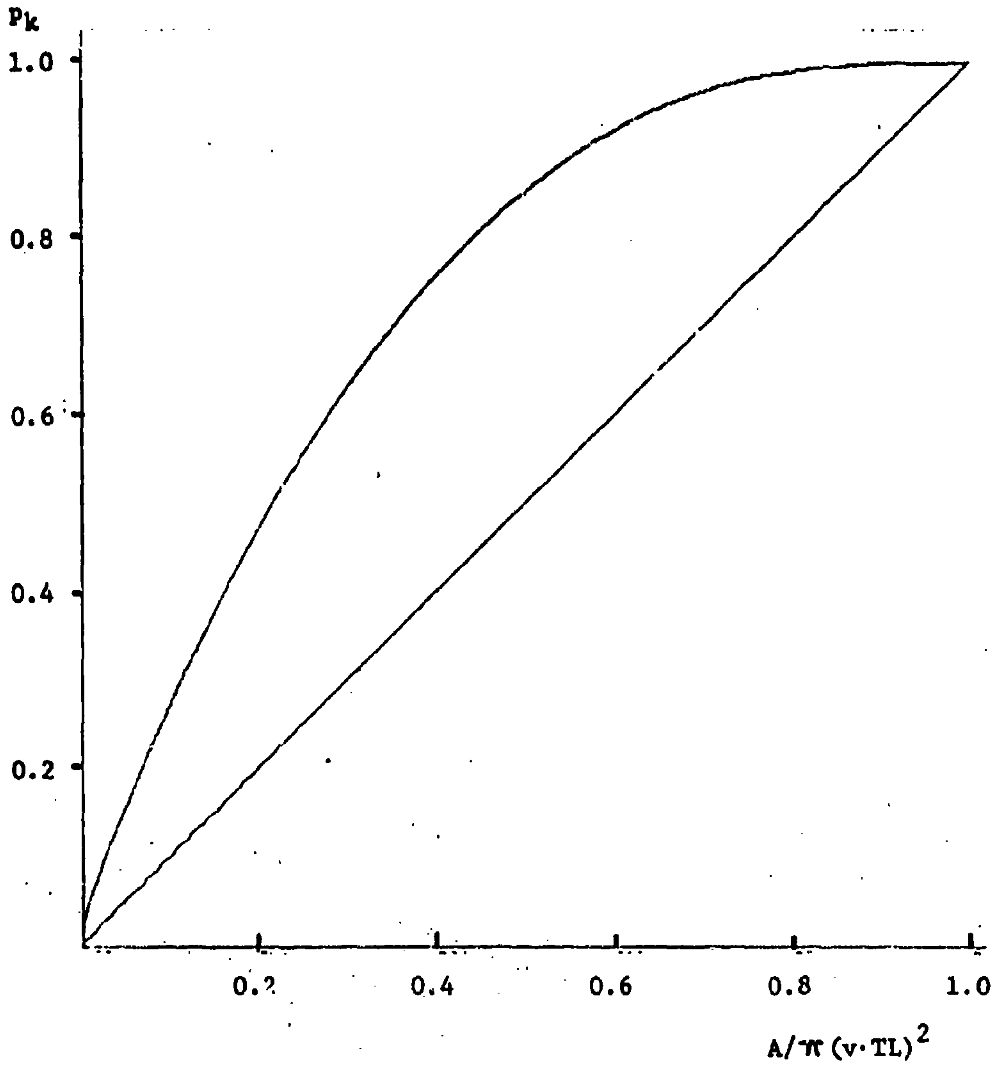
p_k versus Attacker Strength

Course Cnange Rule: Modified Left-Right ET = 0.8



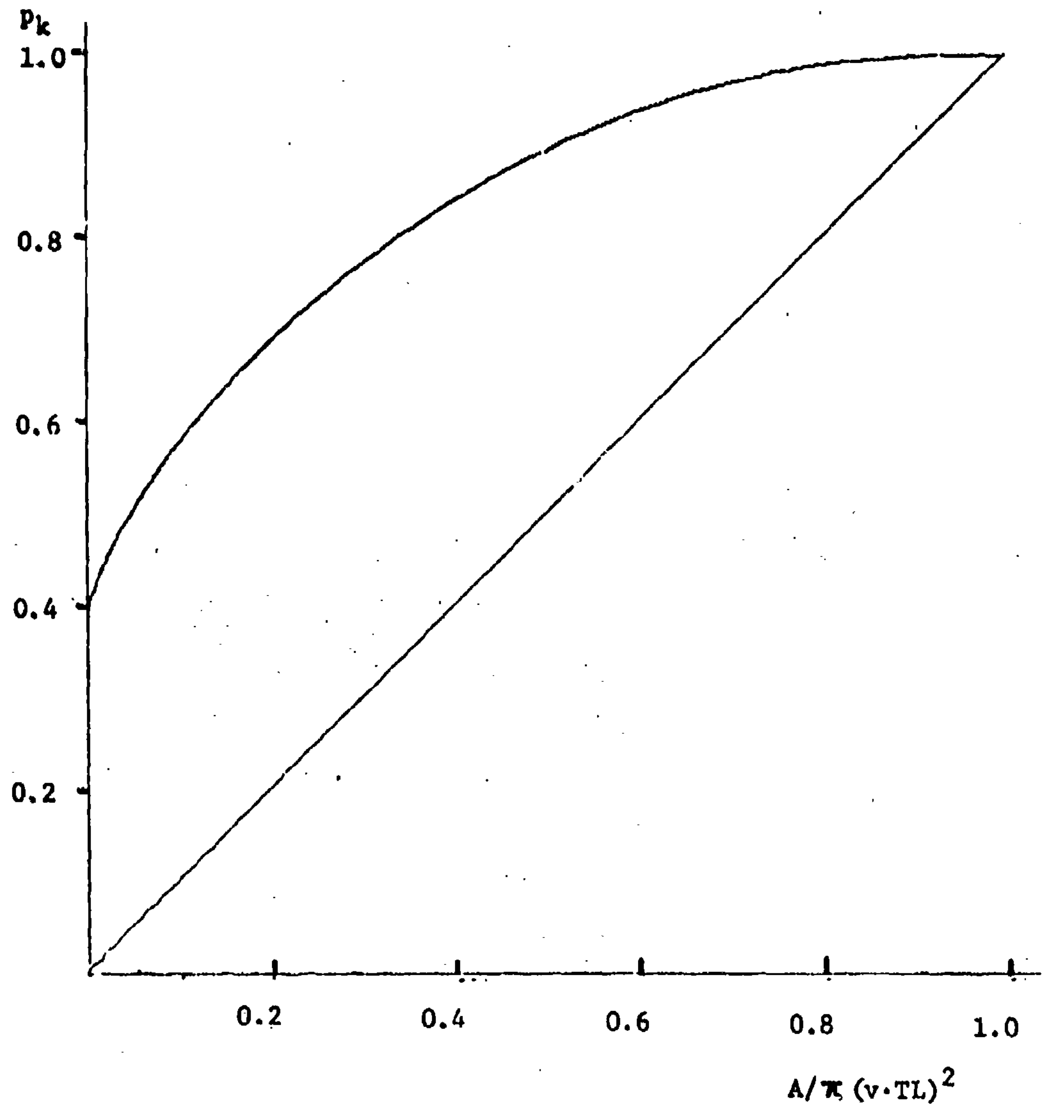
P_k versus Attacker Strength

Course Change Rule: Modified Left-Right ET = 0.66



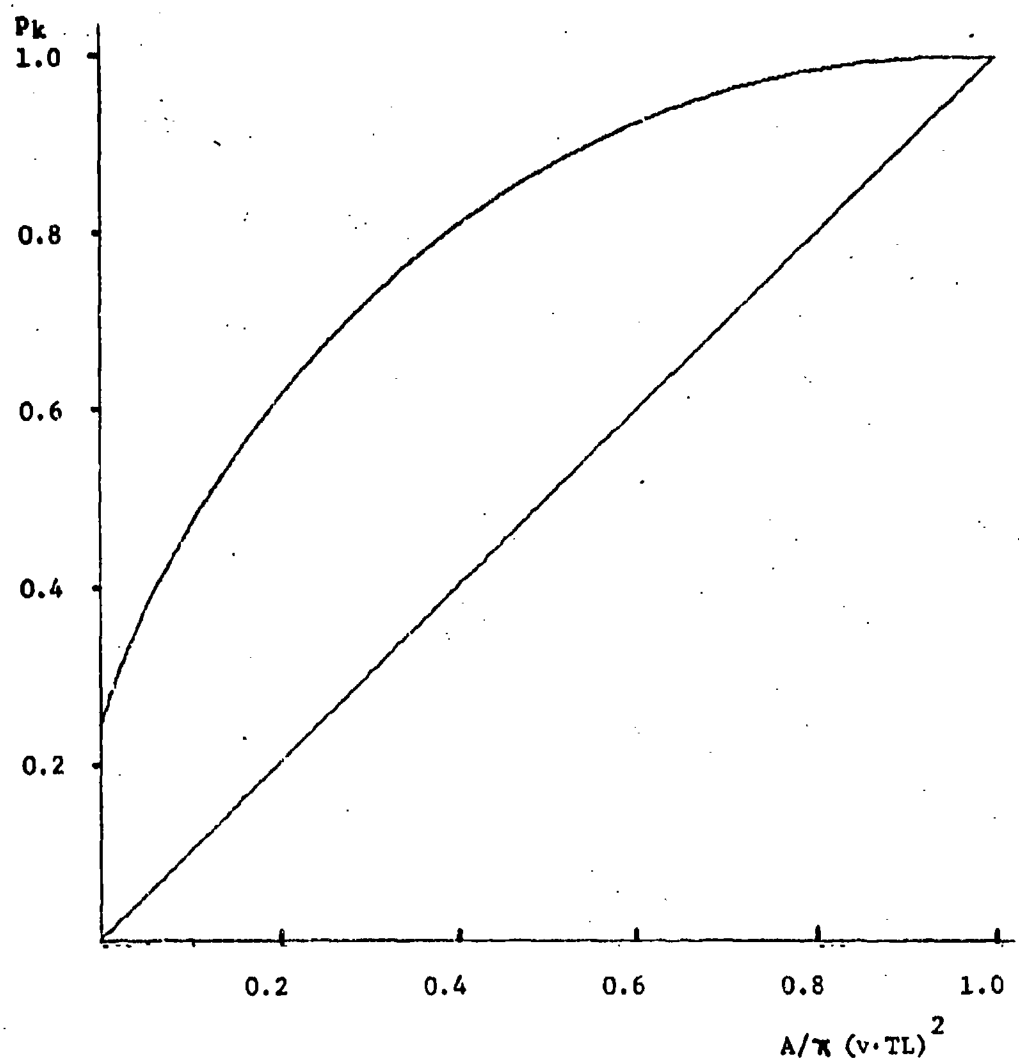
p_k versus Attacker Strength

Course Change Rule: Modified Left-Right ET = 0.5



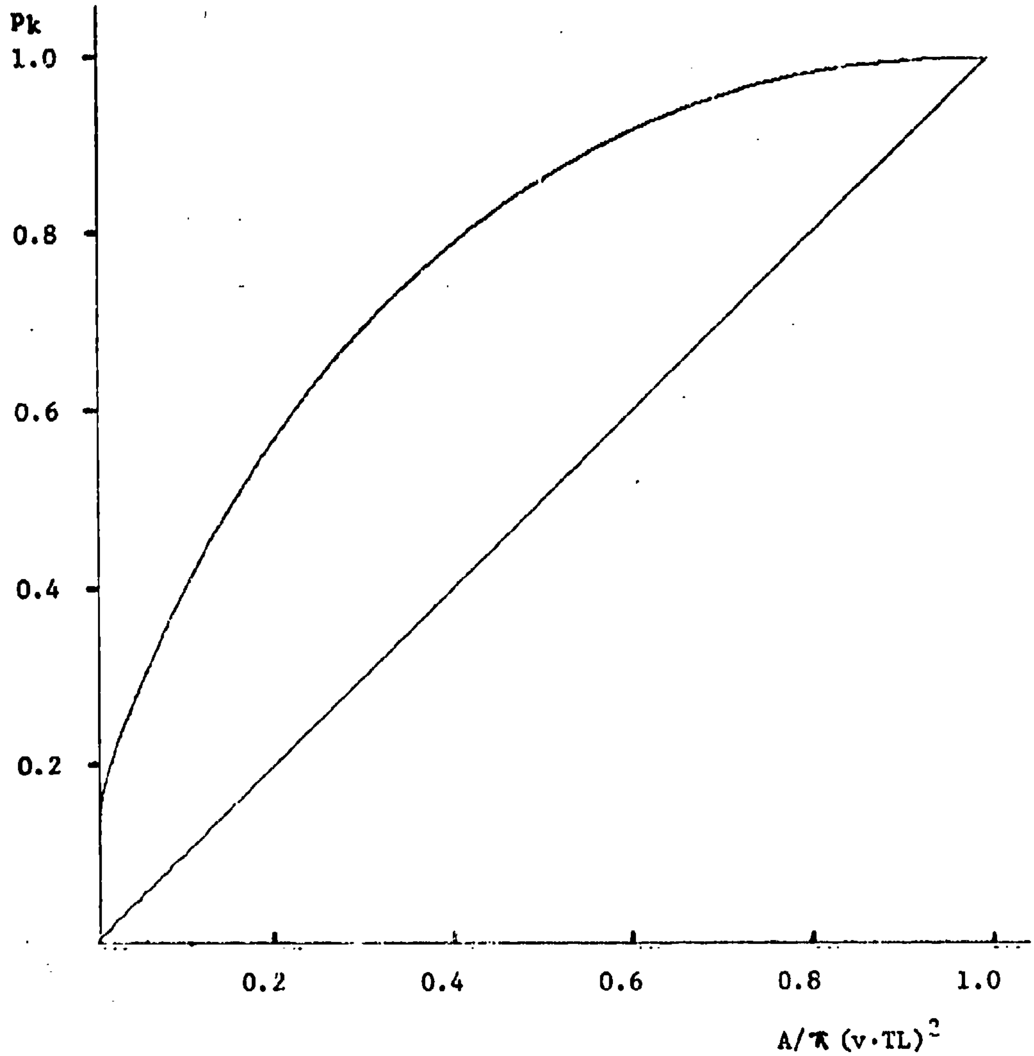
p_k versus Attacker Strength

Course Change Rule: Reverse Course ET = 2.0



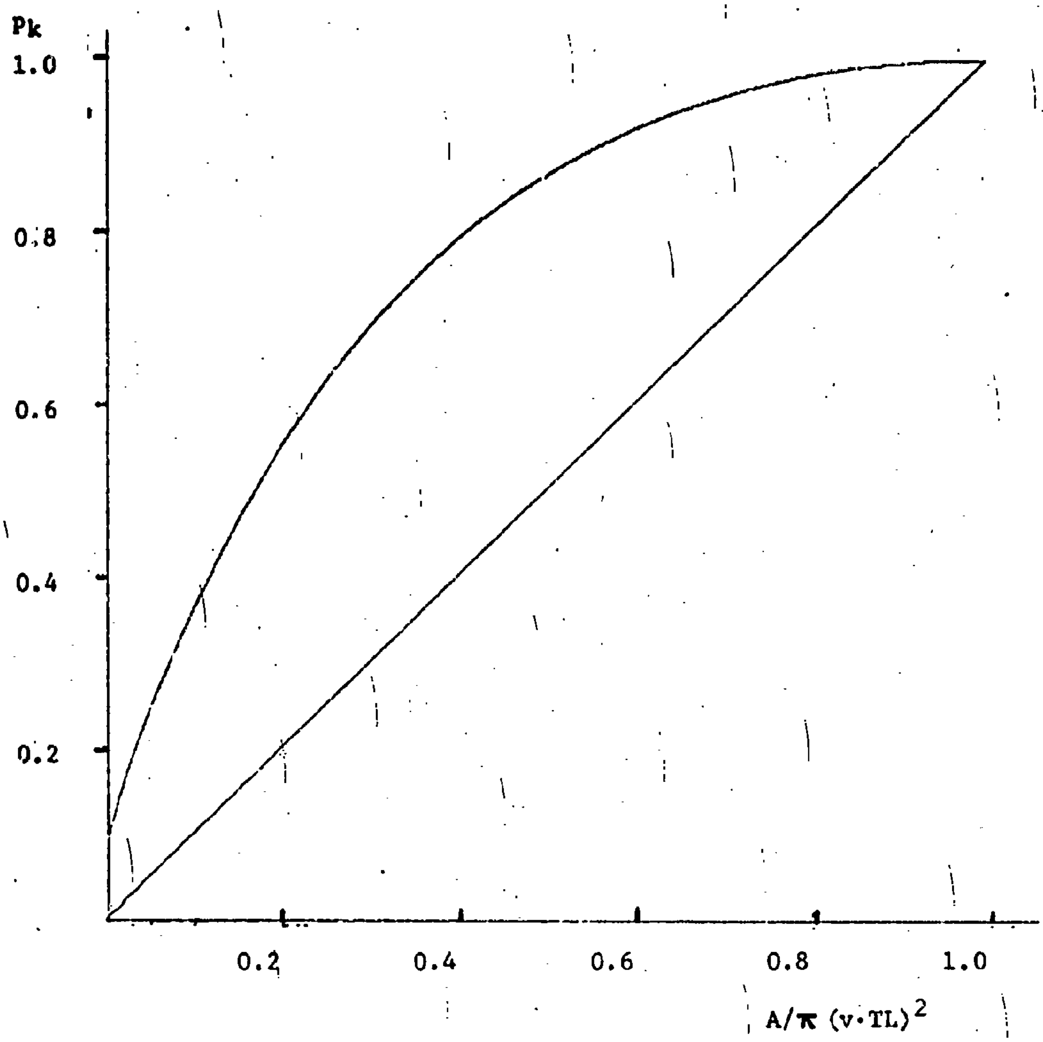
P_k versus Attacker Strength

Course Change Rule: Reverse Course ET = 1.33



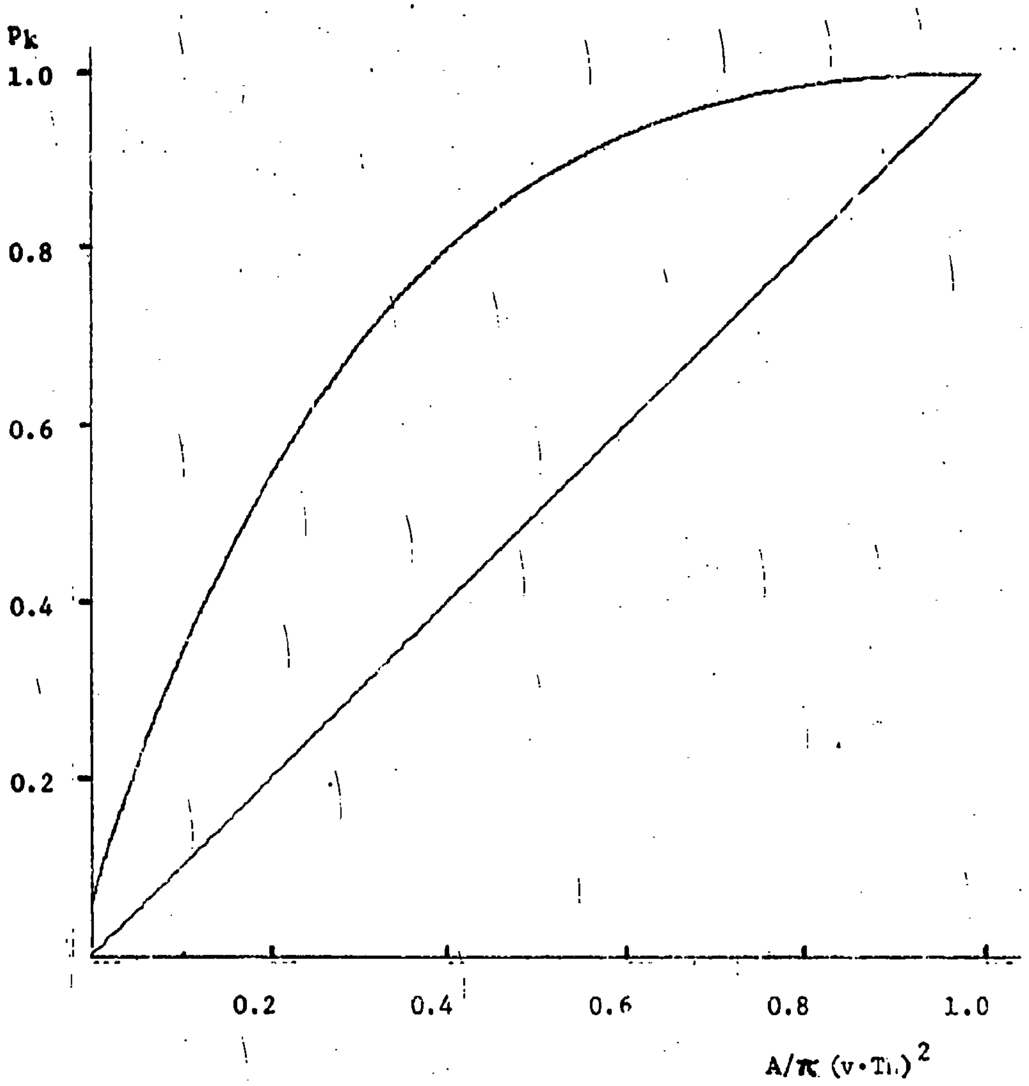
P_k versus Attacker Strength

Course Change Rule: Reverse Course ET = 1.0



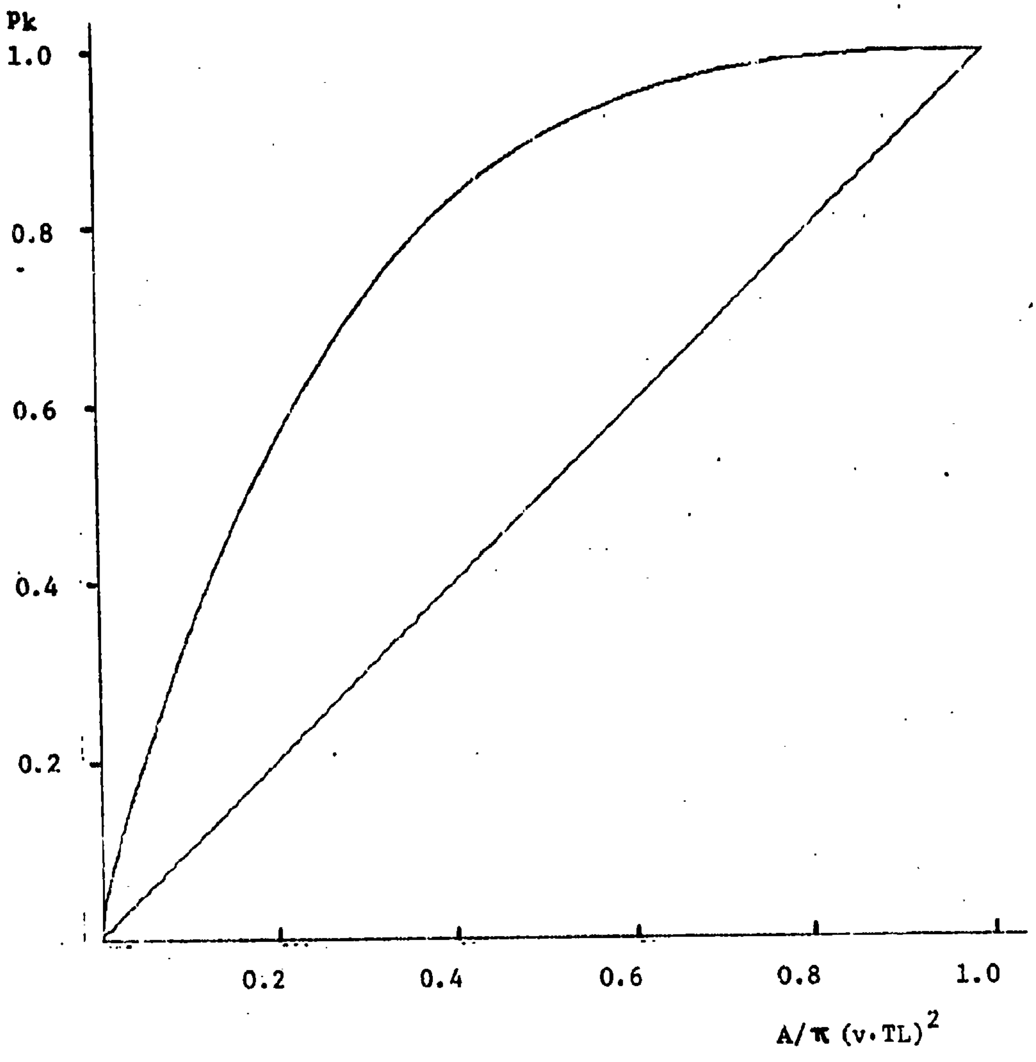
P_k versus Attacker Strength

Course Change Rule: Reverse Course ET = 0.8



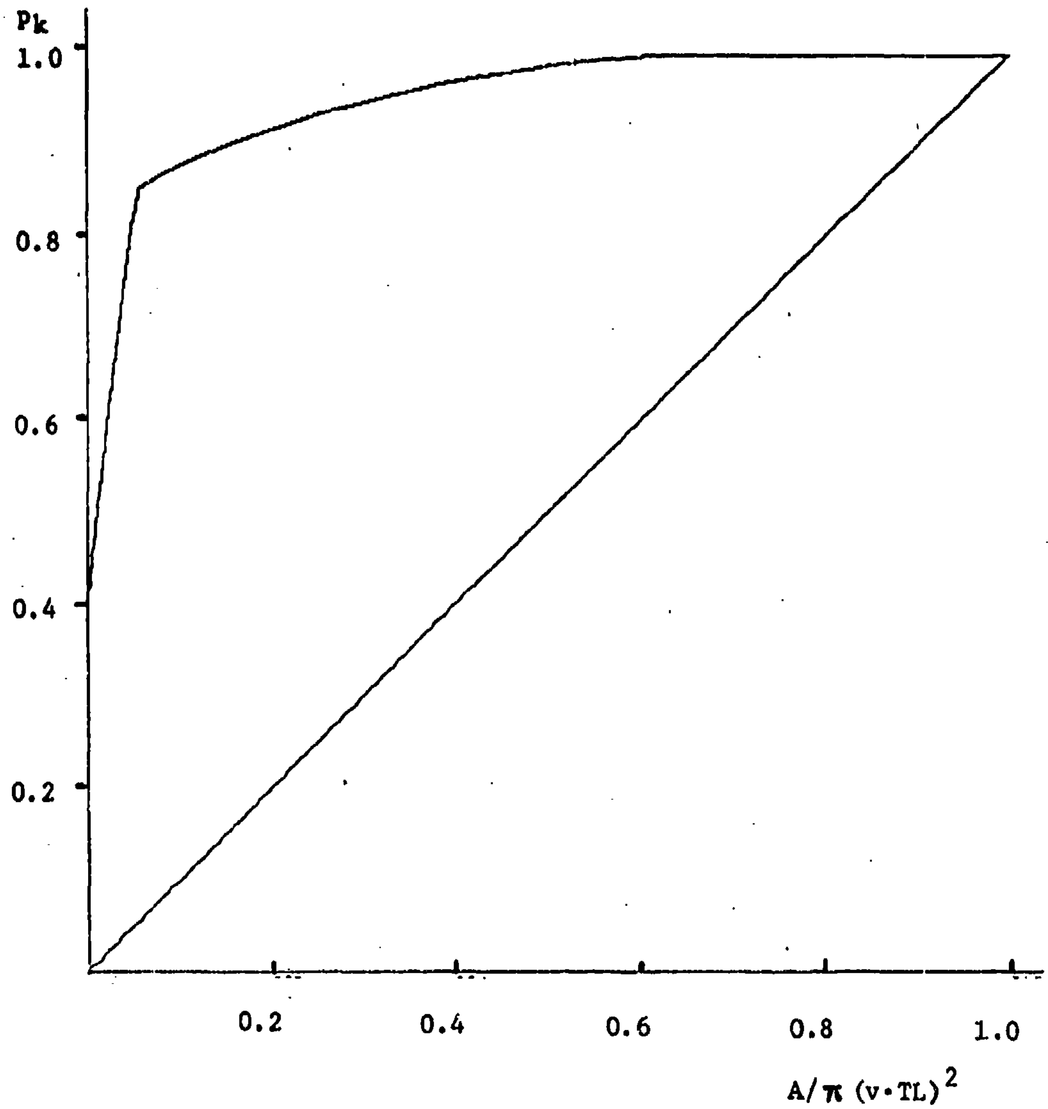
P_k versus Attacker Strength

Course Change Rule: Reverse Course ET = 0.66



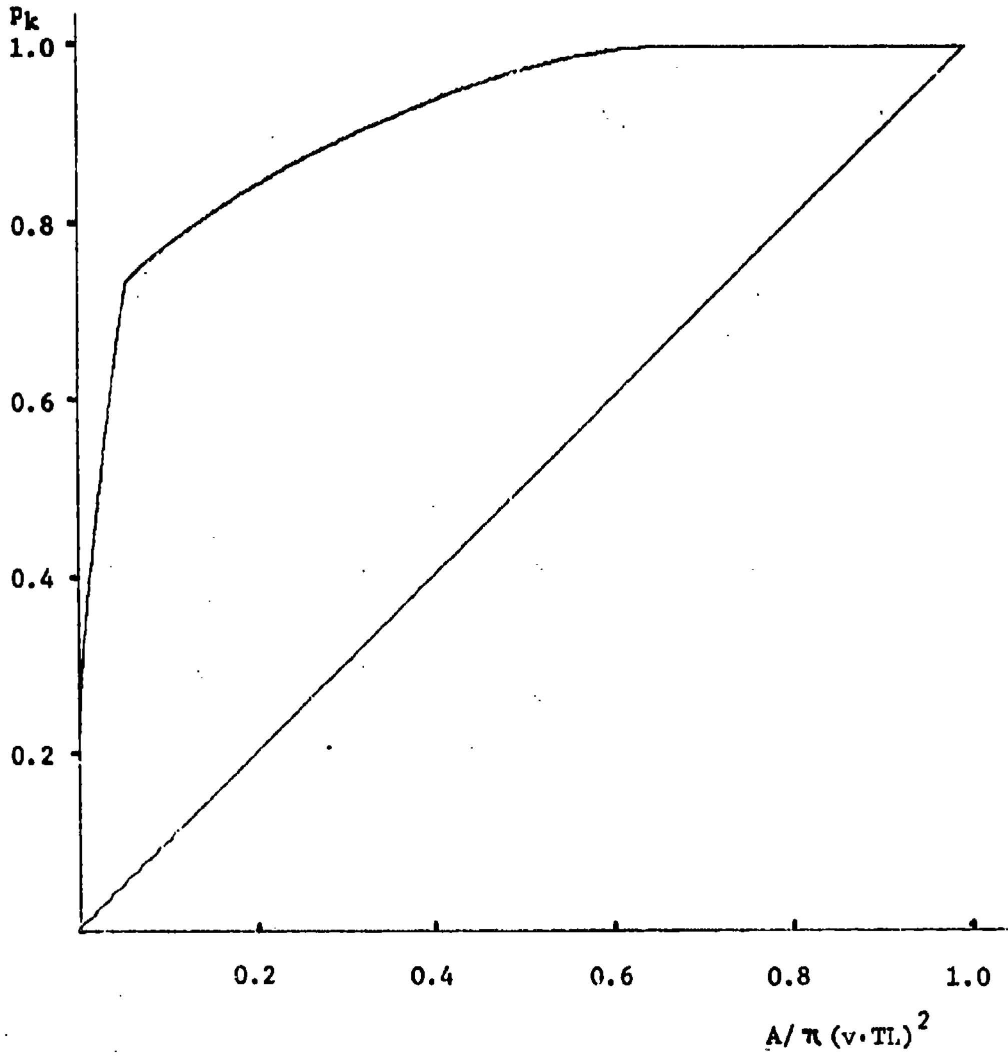
P_k versus Attacker Strength

Course Change Rule: Reverse Course ET = 0.5



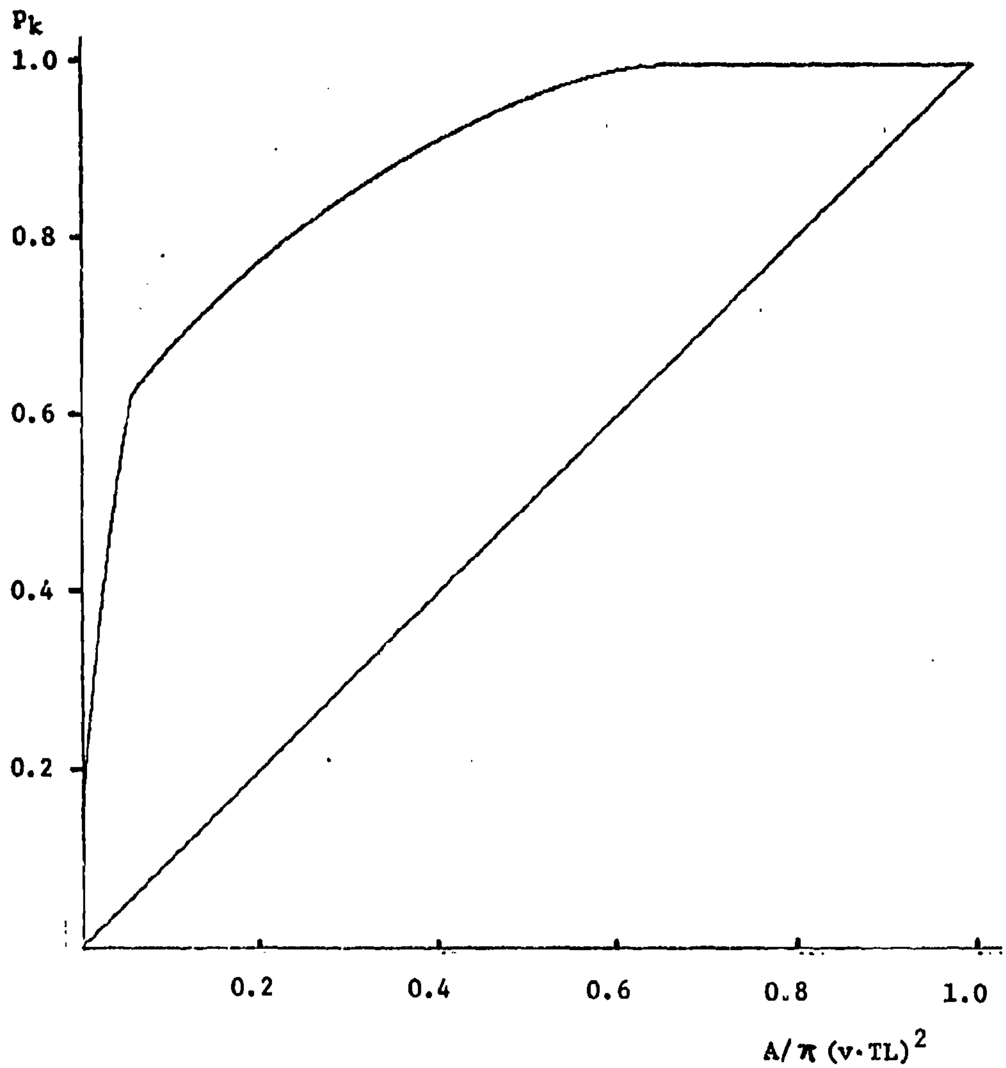
P_k versus Attacker Strength

Course Change Rule: Left-Right ET = 2.0



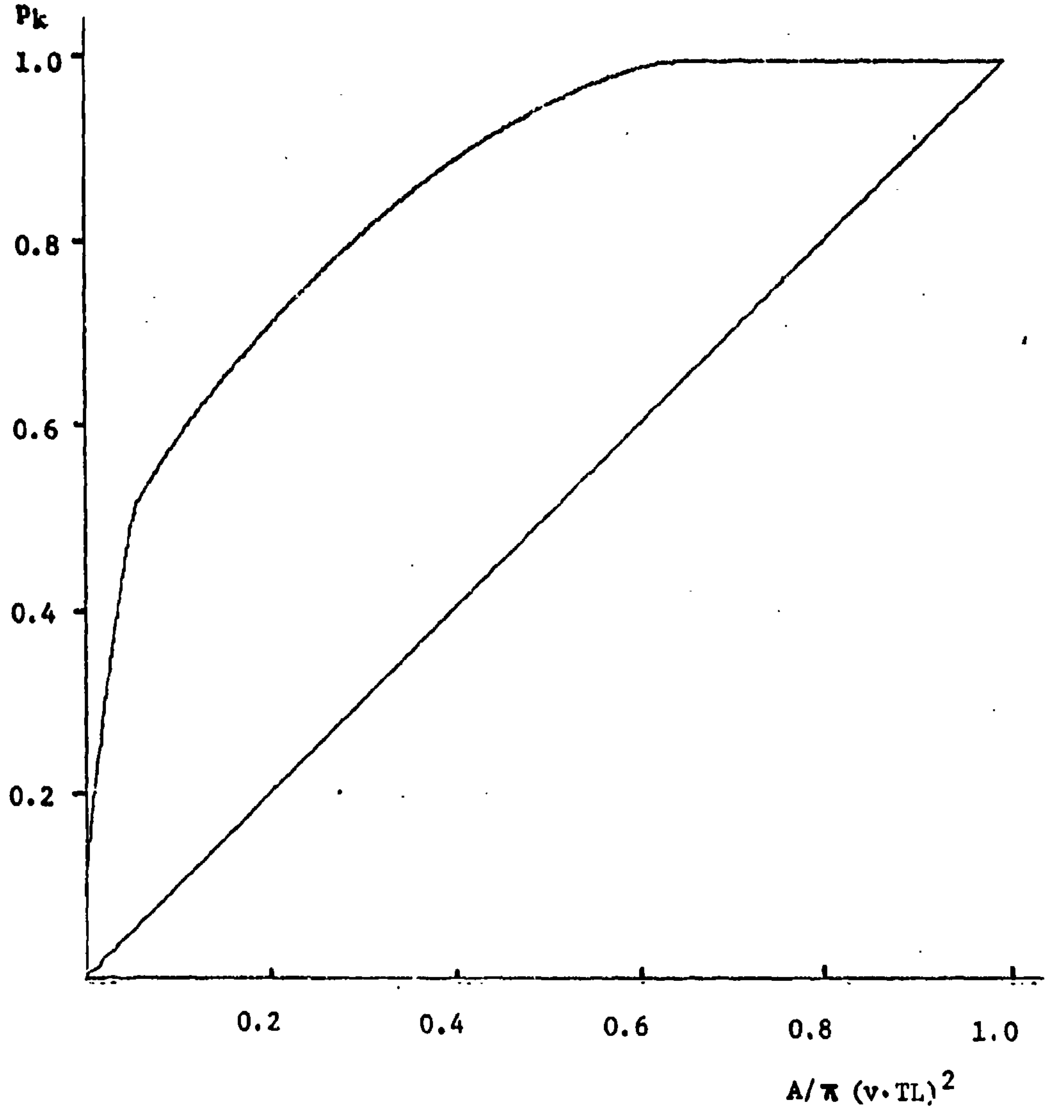
p_k versus Attacker Strength

Course Change Rule: Left-Right ET = 1.33



P_k versus Attacker Strength

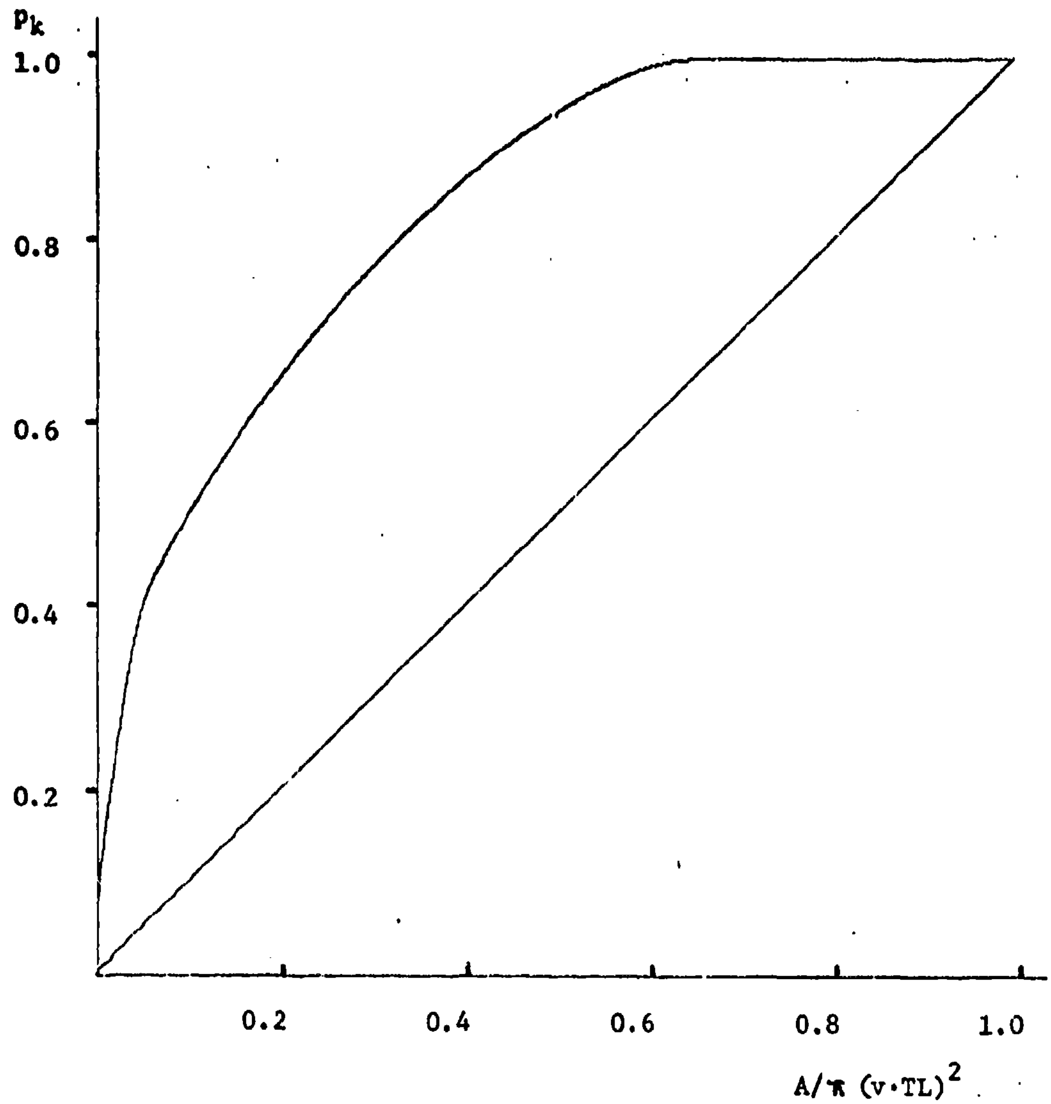
Course Change Rule: Left-Right ET = 1.0



p_k versus Attacker Strength

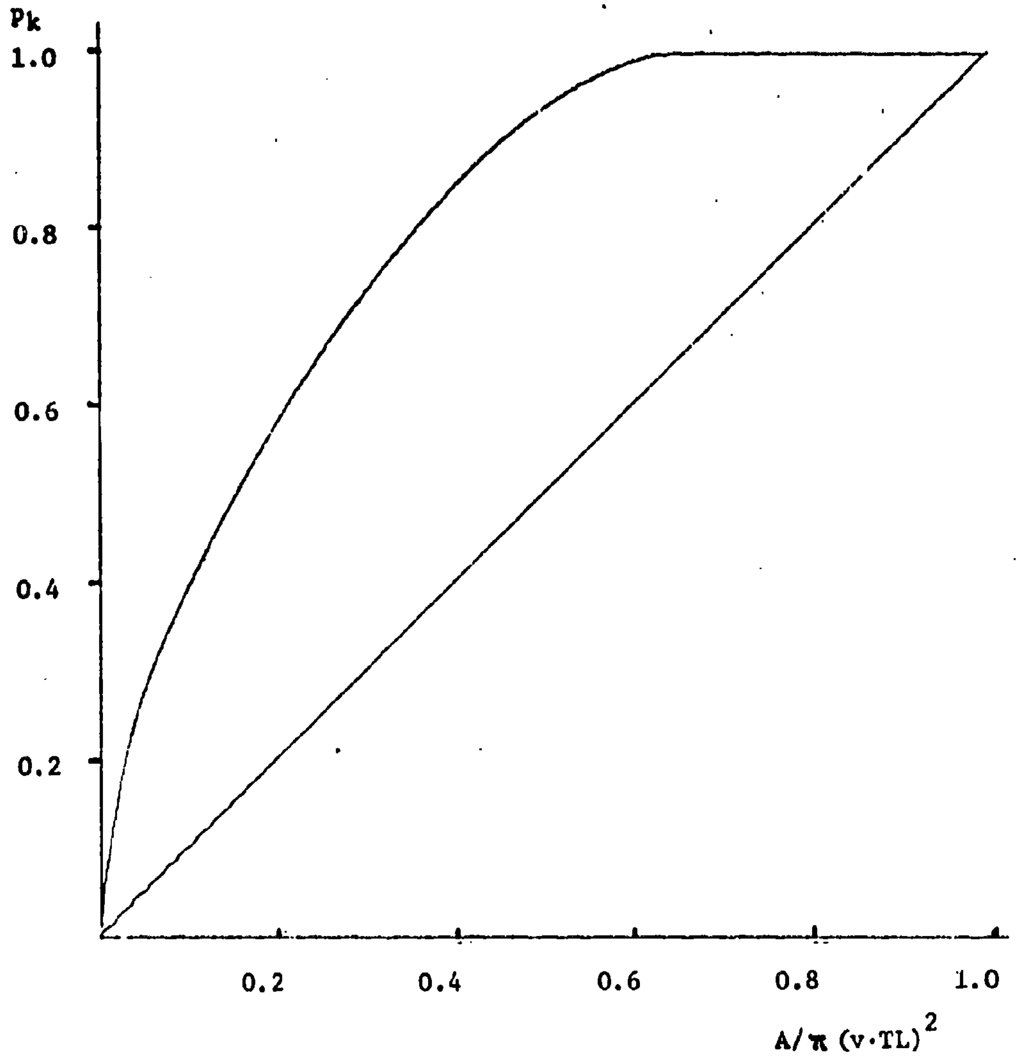
Course Change Rule: Left-Right

ET = 0.8



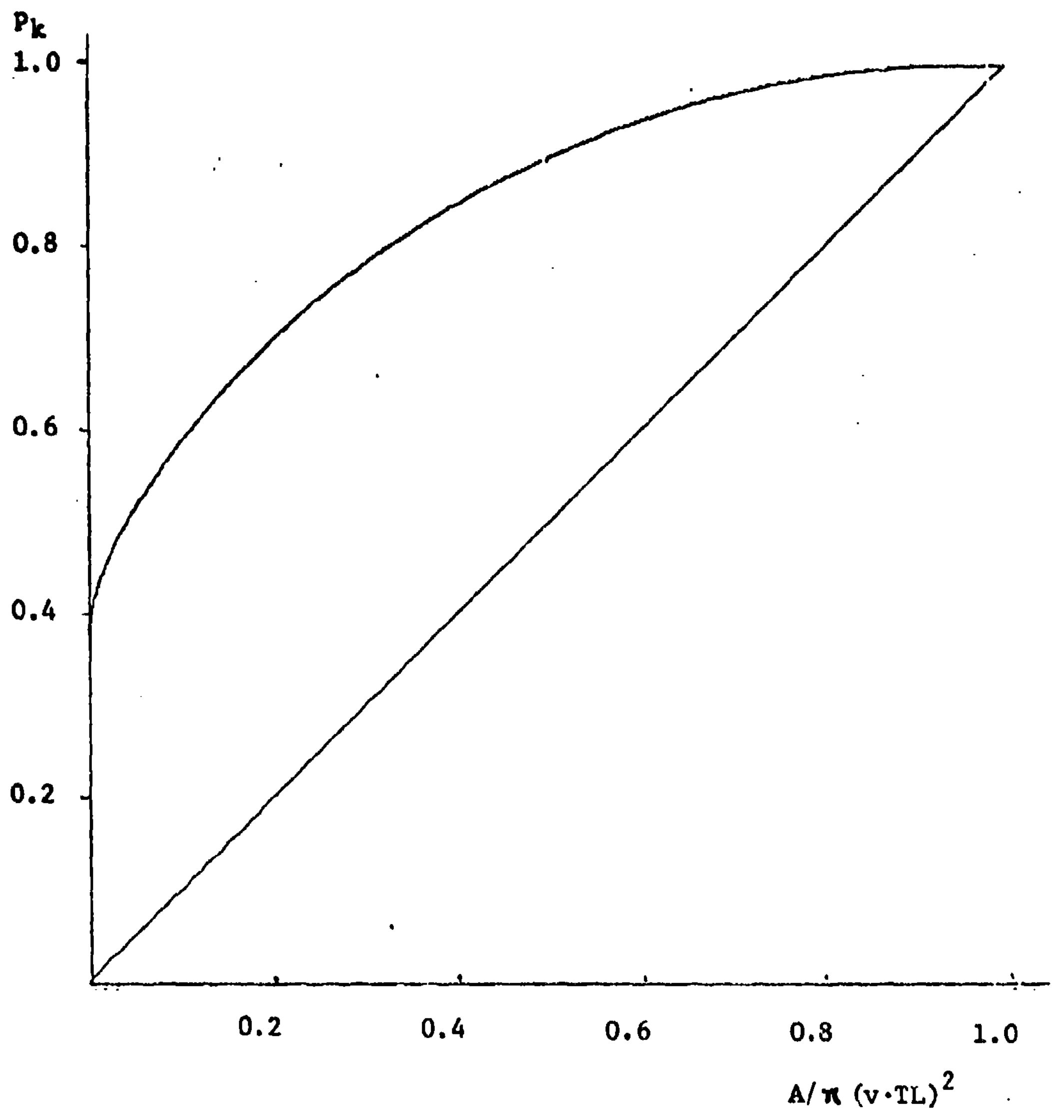
p_k versus Attacker Strength

Course Change Rule: Left-Right ET = 0.66



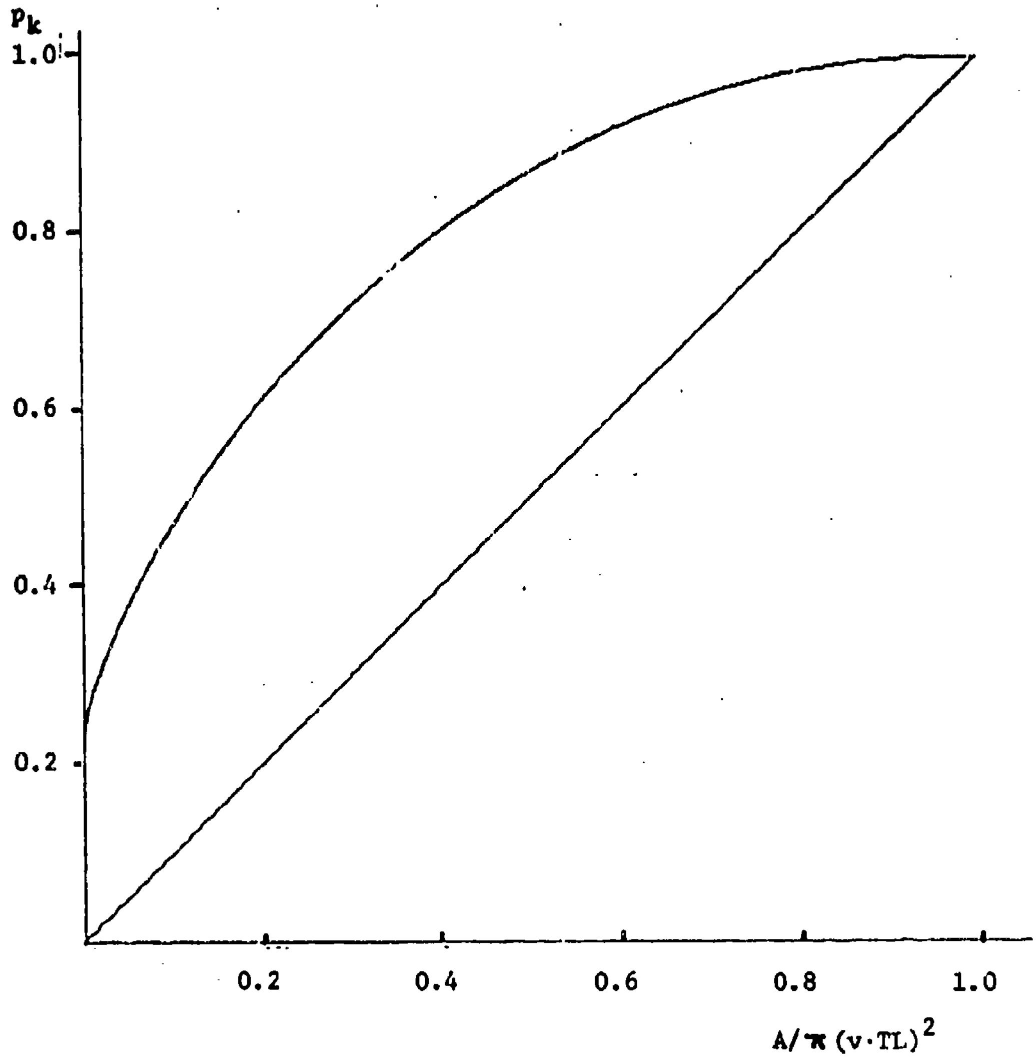
p_k versus Attacker Strength

Course Change Rule: Left-Right ET = 0.5



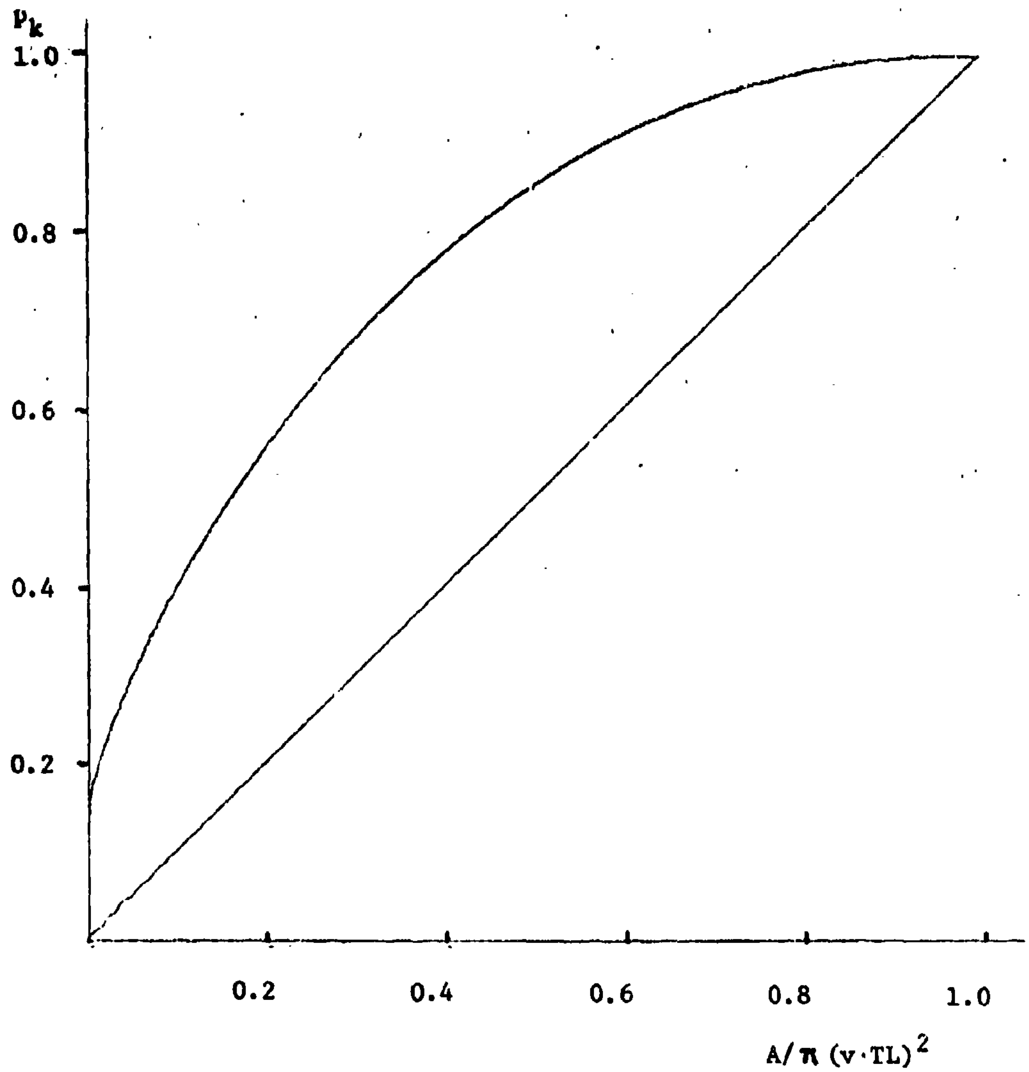
P_k versus Attacker Strength

Course Change Rule: Truncated Uniform FT = 2.0



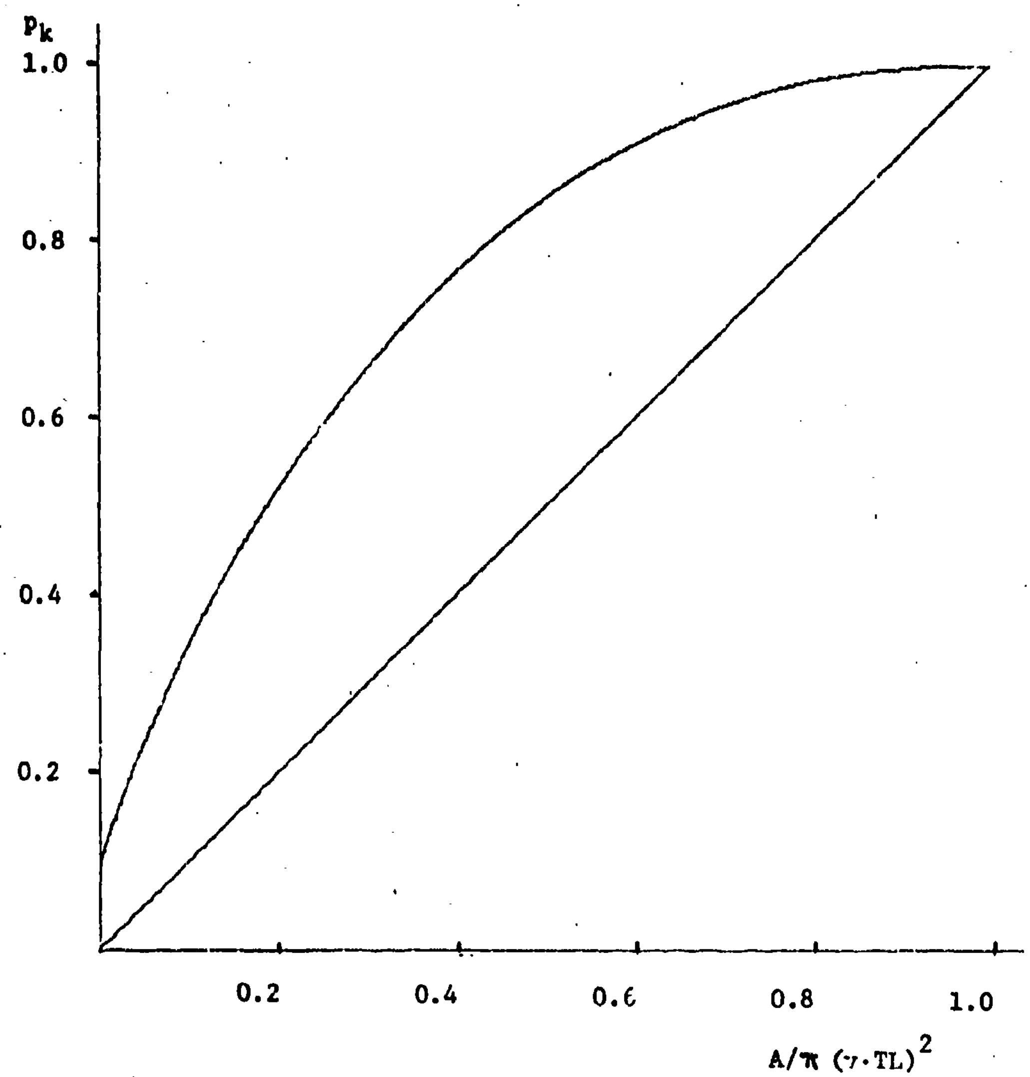
p_k versus Attacker Strength

Course Change Rule: Truncated Uniform ET = 1.33



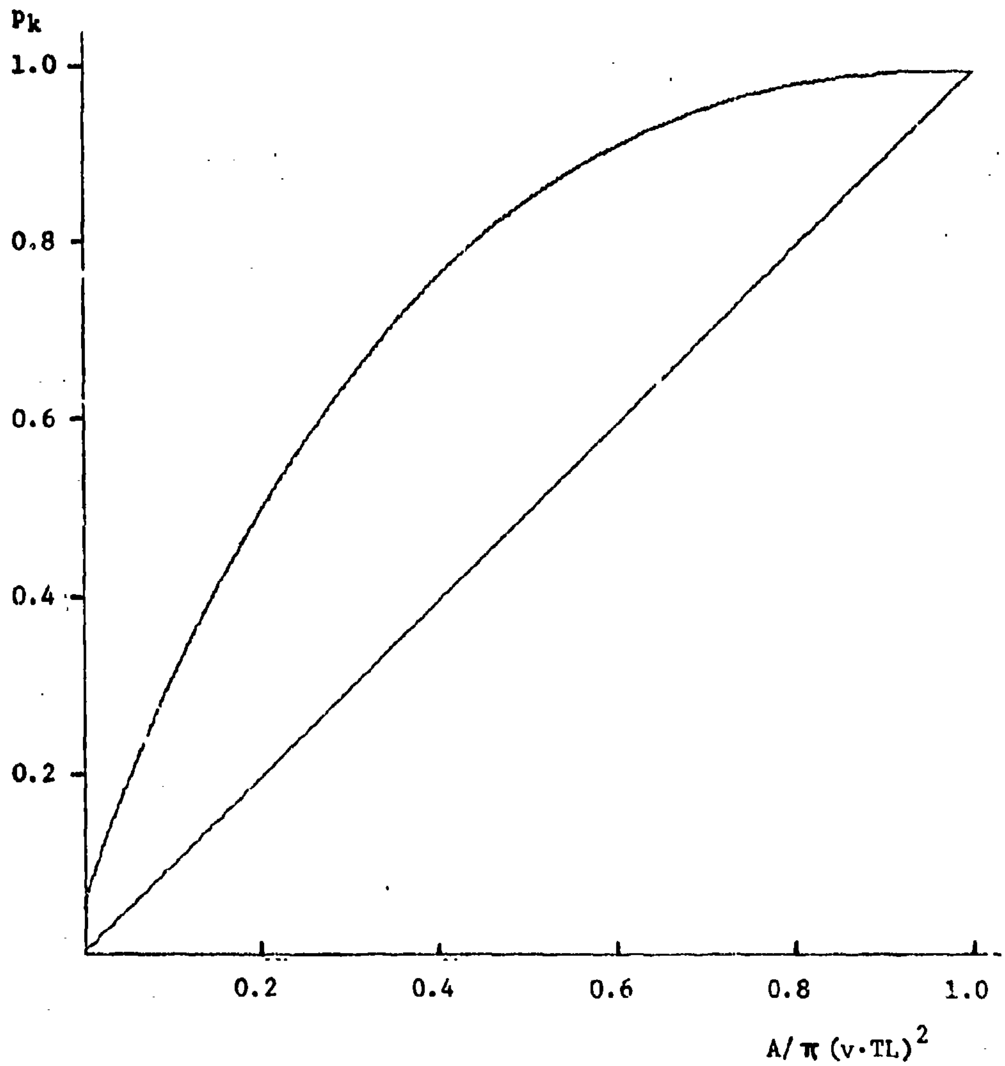
P_k versus Attacker Strength

Course Change Rule: Truncated Uniform ET = 1.0



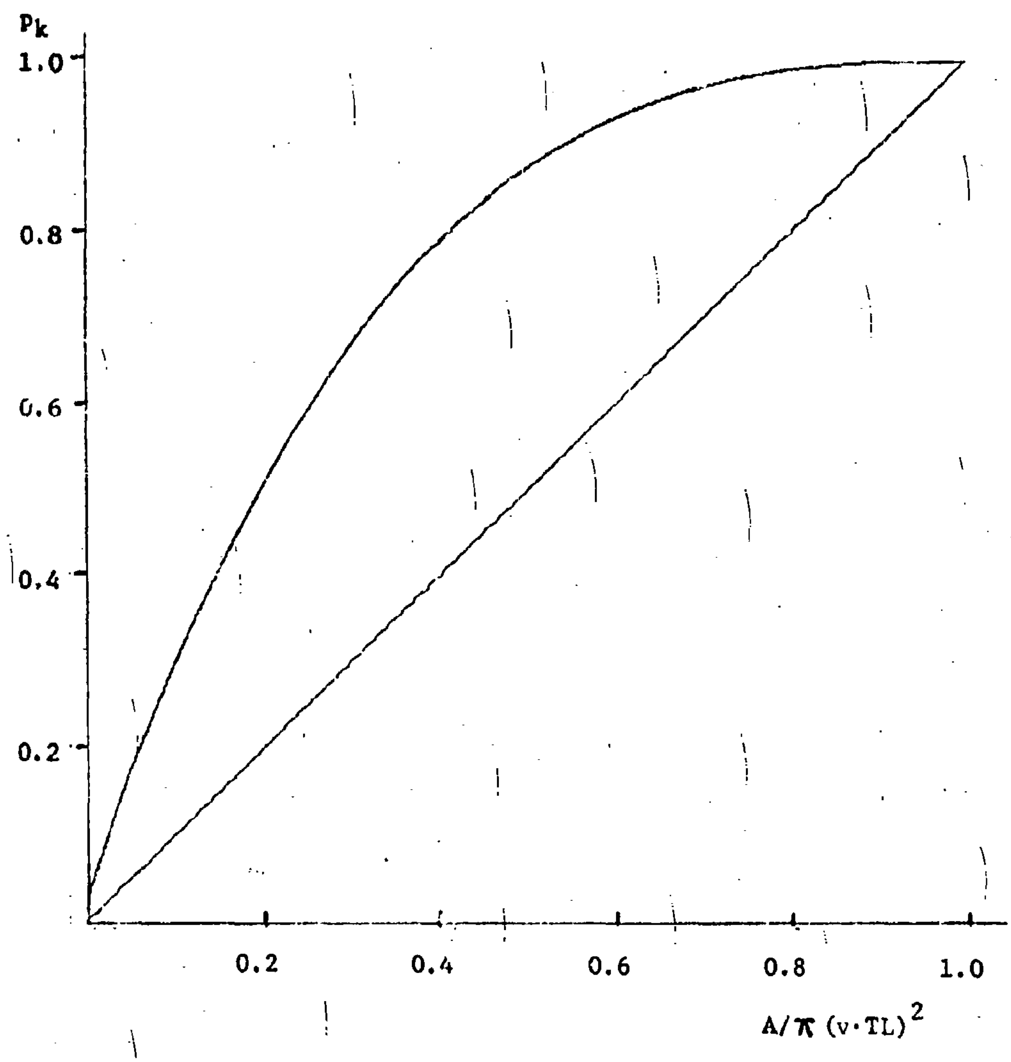
P_k versus Attacker Strength

Course Change Rule: Truncated Uniform ET = 0.8



P_k versus Attacker Strength

Course Change Rule: Truncated Uniform ET = 0.66



P_k versus Attacker Strength

Course Change Rule: Truncated Uniform ET = 0.5

COMPUTER PROGRAM

A SIMULATION OF THE CONTINUOUS EVASIVE GAME FOR THE POISSON CLASS OF STRATEGIES

```
      INTEGER * 2 PP,INDEX
      INTEGER * 2 CUMSUM
      INTEGER * 2 NCOUNT
      INTEGER *2 XP,QQ
      INTEGER * 4 CRITE3
      INTEGER * 4 TOTCEL
      REAL * 4 IN
      REAL LABEL/'    '/
      REAL*8 ITITLE(12)/'CLOTHIER','  THOMAS','  J,     ','3*'
     X      ','STRATEGY',' TRNCRS',' (LAMDA',' * TL)= ',
     X2*'    4.0'/
C     BY MAKING THE HISTOGRAM INTEGER STAR TWO THE AMOUNT
C     OF STORAGE REQUIRED FOR THE PROGRAM WILL BE REDUCED.
C
C
      DIMENSION PP(41,41),INDEX(41,41)
      DIMENSION XP(1681),QQ(41,41)
      EQUIVALENCE (PP(1,1),XP(1))
      DIMENSION EEX(2),EEY(2),EX(692),EY(692),EXX(692)
      DIMENSION EYY(692)
C
C
C     PP IS THE PLAYING AREA AND HISTOGRAM.  EACH CELL IN
C     PP REPRESENTS AN AREA OF SIZE CELSIZ.  INDEX IS A
C     BOOK-KEEPING ARRAY USED TO DETERMINE WHICH CELLS IN
C     PP ARE FEASIBLE, E.G. IF INDEX(I,J)=0 THEN PP(I,J) IS
C     AN INFEASIBLE CELL.  SIMILARLY IF INDEX(I,J)=1 OR 2
C     THEN PP(I,J) IS A FEASIBLE CELL.
C
C     PP IS A SQUARE WITH THE TIME-LATE CIRCLE INSCRIBED IN
C     IT.  THEREFORE PP HAS SOME CELLS THAT ARE NOT INSIDE
C     THE TIME-LATE CIRCLE AND FOR THOSE CELLS THEIR INDEX
C     VALUE IS ZERO.  FOR THOSE CELLS IN PP THAT ARE CON-
C     TAINED ENTIRELY IN THE CIRCLE THE ASSOCIATED INDEX
C     VALUE IS ONE.  AND LASTLY THERE ARE THE CELLS IN PP
C     THAT ARE ON THE EDGE OF THE CIRCLE, THESE ARE CALLED
C     EDGE CELLS AND HAVE THE INDEX VALUE OF TWO.
C
C
      NCOUNT=0
      PI=3.1415926
      DATA IX,V,TL,ET/37915,05.0,02.00,00.5/
C     V IS PARTICLE SPEED
C     TL IS THE TIME-LATE
C     ET IS THE MEAN TIME BETWEEN COURSE CHANGES
C     IX IS THE SEED FOR THE PSEUDO-RANDOM NUMBER GENERATORS
C
C
      DELTAX=(2.0*V*TL)/41.0
C     DELTAX IS THE HORIZONTAL DIMENSION OF A CELL.
C
      CELSIZ=DELTAX**2
C     CELSIZ IS THE AMOUNT OF REAL AREA CONTAINED IN A CELL.
C
      K=20
C     THE NUMBER OF CELLS IN EITHER DIMENSION IS ALWAYS A
C     CONSTANT 41.  THIS CONSTANT IS EXPRESSED AS AN ODD
C     INTEGER AND WRITTEN IN THE FOLLOWING MANNER,
C     41 = 2 * (K) + 1.  THEREFORE K = 20.
C
C
C     ZERO OUT THE HISTOGRAM
      DO 5   I=1,41
      DO 4   J=1,41
      INDEX(I,J)=0
```

```
      PP(I,J)=0
      QQ(I,J)=0
    4 CONTINUE
    5 CONTINUE
C
C
C     NOW TO CHECK EACH OF THE CELLS IN THE HISTOGRAM AND
C     DETERMINE ITS ASSOCIATED INDEX VALUE.
C
C
C     QUADRANT I
      DO 42 J=21,41
      DO 41 I=1,21
      OUT=(V*TL-J*DELTAX)**2+((I-1)*DELTAX-V*TL)**2
      IN =(V*TL-(J-1)*DELTAX)**2+(I*DELTAX-V*TL)**2
      RADSQ=(V*TL)**2
      IF((OUT.LE.RADSQ).AND.(IN.LE.RADSQ))INDEX(I,J)=1
      IF((OUT.GT.RADSQ).AND.(IN.GT.RADSQ))INDEX(I,J)=0
      IF((OUT.GT.RADSQ).AND.(IN.LE.RADSQ))INDEX(I,J)=2
   41 CONTINUE
   42 CONTINUE
C
C
C
C     QUADRANT II CHECK
      DO 44 J=1,21
      DO 43 I=1,21
      OUT=(V*TL-(I-1)*DELTAX)**2+(V*TL-(J-1)*DELTAX)**2
      IN=(V*TL-I*DELTAX)**2+(V*TL-J*DELTAX)**2
      RADSQ=(V*TL)**2
      IF((OUT.LE.RADSQ).AND.(IN.LE.RADSQ))INDEX(I,J)=1
      IF((OUT.GT.RADSQ).AND.(IN.GT.RADSQ))INDEX(I,J)=0
      IF((OUT.GT.RADSQ).AND.(IN.LE.RADSQ))INDEX(I,J)=2
   43 CONTINUE
   44 CONTINUE
C
C
C     QUADRANT III
      DO 46 J=1,21
      DO 45 I=21,41
      OUT=(V*TL-(J-1)*DELTAX)**2+(I*DELTAX-V*TL)**2
      IN=(V*TL-J*DELTAX)**2+((I-1)*DELTAX-V*TL)**2
      RADSQ=(V*TL)**2
      IF((OUT.LE.RADSQ).AND.(IN.LE.RADSQ))INDEX(I,J)=1
      IF((OUT.GT.RADSQ).AND.(IN.GT.RADSQ))INDEX(I,J)=0
      IF((OUT.GT.RADSQ).AND.(IN.LE.RADSQ))INDEX(I,J)=2
   45 CONTINUE
   46 CONTINUE
C
C
C     QUADRANT IV CHECK
      DO 48 J=21,41
      DO 47 I=21,41
      OUT=(V*TL-I*DELTAX)**2+(V*TL-J*DELTAX)**2
      IN =(V*TL-(I-1)*DELTAX)**2+(V*TL-(J-1)*DELTAX)**2
      RADSQ=(V*TL)**2
      IF((OUT.LE.RADSQ).AND.(IN.LE.RADSQ))INDEX(I,J)=1
      IF((OUT.GT.RADSQ).AND.(IN.GT.RADSQ))INDEX(I,J)=0
      IF((OUT.GT.RADSQ).AND.(IN.LE.RADSQ))INDEX(I,J)=2
   47 CONTINUE
   48 CONTINUE
C
C
C     NOW TO SIMULATE THE PARTICLE MOTION UNDER THE
C     SPECIFIED RULES.  THIS IS THE MAIN DO LOOP IN THE
C     PROGRAM.  FIRST A TIME UNTIL THE NEXT COURSE CHANGE
C     IS GENERATED.  THEN A COURSE TO BE STEERRED IS GEN-
C     ERATED.  XX AND YY ARE THE TWO COMPONENTS OF POSITION
C     ADDED DUE TO A COURSE AND TIME  SEGMENT.   X AND Y ARE
C     THE UPDATED POSITION FROM DATUM.   AT TIME-LATE THE
C     UPDATED POSITION IS CONVERTED TO A CELL POSITION AND
C     THAT CELL VALUE IS INCREMENTED BY ONE.
```

```
C
C
      DO 3 J=1,16810
      X=0.0
      Y=0.0
      CLOCK=0.0
      C=0.0
      TR=TL-CLOCK
      CALL RANDU(IX,IY,R)
      IX=IY
      CALL CORSE(R,C)
      THETA=C
      GO TO 66
    1 TR=TL-CLOCK
      CALL RANDU(IX,IY,R)
      IX=IY
      CALL TRNCRS(R,C)
   66 CALL RANDU(IX,IY,R)
      IX=IY
      CALL TRKTM(ET,R,T)
      TT=AMIN1(T,TR)
      CALL MOVE(V,TT,C,XX,YY)
      CLOCK=CLOCK+T
      X=X+XX
      Y=Y+YY
      IF(CLOCK.LT.TL)GO TO 1
      XPRIME=X*COS(THETA)+Y*SIN(THETA)
      YPRIME=Y*COS(THETA)-X*SIN(THETA)
      X=XPRIME
      Y=YPRIME
      CALL RANDU(IX,IY,R)
      IX=IY
      CORR=1.0
      CALL HISTO(K,DELTAX,R,CORR,X,I)
      L=I
      CALL RANDU(IX,IY,R)
      IX=IY
      CORR=-1.0
      CALL HISTO(K,DELTAX,R,CORR,Y,I)
      M=I
C     THE PARTICULAR CELL I,J IS DETERMINED BY TWO  SUCCESS-
C     IVE CALLS OF HISTO.  IT IS POSSIBLE A POSITION X,Y
C     COULD BE DETERMINED TO BE IN AN INFEASIBLE CELL.
C     IF THIS HAPPENS THEN RESOLV WILL BE CALLED TO CHANGE
C     THAT CELL ASSIGNMENT IN A PRESCRIBED MANNER TO A
C     FEASIBLE CELL.
      IF(INDEX(M,L).EQ.0)GO TO 31
   30 PP(M,L)=PP(M,L)+1
      GO TO 3
   31 CALL RESOLV(L,M,I,N)
      NCOUNT=NCOUNT+1
      L=I
      M=N
      GO TO 30
    3 CONTINUE
C
      NUPCEL=0
      NUPEDC=0
      CRITE2=0.0
      DO 52 I=1,41
      DO 51 J=1,41
      IF(PP(I,J).EQ.0)GO TO 51
      NUPCEL=NUPCEL+1
      IF(INDEX(I,J).NE.2)GO TO 51
      NUPEDC=NUPEDC+1
      CRITE2=CRITE2+PP(I,J)/1681..0
   51 CONTINUE
   52 CONTINUE
      CRITE3=NUPEDC
      RNUPED=NUPEDC
      RNUPCE=NUPCEL
      CRITE1=RNUPED/RNUPCE
```

```
      NUPINC=NUPCEL-NUPEDC
  212 WRITE(6,203)
  203 FORMAT('1',60X,'PP   PAGE 1')
      DO 205 I=1,41
      WRITE(6,204) (PP(I,J),J=1,15)
  204 FORMAT (' ',15I7)
  205 CONTINUE
      WRITE (6,206)
  206 FORMAT('1',60X,'PP   PAGE 2')
      DO 208 I=1,41
      WRITE(6,207)(PP(I,J),J=15,29)
  207 FORMAT(' ',15I7)
  208 CONTINUE
      WRITE (6,209)
  209 FORMAT('1',60X,'PP   PAGE 3')
      DO 211 I=1,41
      WRITE(6,210)(PP(I,J),J=29,41)
  210 FORMAT(' ',13I7)
  211 CONTINUE
C
C     NOW THAT THE HISTOGRAM IS BUILT ITS ELEMENTS  ARE
C     ORDERED SO THE HIGHEST VALUES ARE IN THE LOWEST NUM-
C     BERED CELLS.
C
C
 1000 IC=0
      DO 1010 I=1,1680
      IF(XP(I).GE.XP(I+1)) GO TO 1010
      MAX=XP(I+1)
      XP(I+1)=XP(I)
      XP(I)=MAX
      IC=10
 1010 CONTINUE
      IF(IC.NE.0)GO TO 1000
      DO 2000 I=1,41
      DO 2010 J=1,41
      QQ(I,J)=PP(J,I)
 2010 CONTINUE
 2000 CONTINUE
      DO 1030 I=1,41
      DO 1020 J=1,41
      PP(I,J)=QQ(I,J)
 1020 CONTINUE
 1030 CONTINUE
C
C
C     AFTER ORDERING THE CELL VALUES THEY ARE ACCUMLATED SO
C     EACH CELL CONTAINS THE SUM OF ITS OWN VALUE AND ALL
C     THE VALUES PRECEDING IT.
C
C
      CUMSUM=0
      DO 11 I=1,41
      DO 10 J=1,41
      CUMSUM=CUMSUM+PP(I,J)
      PP(I,J)=CUMSUM
   10 CONTINUE
   11 CONTINUE
C
C
      NUMEXC=0
      NUMINC=0
      NUMEDC=0
      DO 33 I=1,41
      DO 32 J=1,41
      IF (INDEX(I,J).EQ.0)NUMEXC=NUMEXC+1
      IF (INDEX(I,J).EQ.1)NUMINC=NUMINC+1
      IF (INDEX(I,J).EQ.2)NUMEDC=NUMEDC+1
   32 CONTINUE
   33 CONTINUE
      NUMCEL=NUMINC+NUMEDC
      TOTCEL=NUMEXC+NUMEDC+NUMINC
```

```
C
C
C       NOW TO LOAD THE FIRST 900 CELL VALUES INTO THE PLOT-
C       TING VECTOR EY.  THE CUMULATIVE NUMBER OF LOOKS OR
C       KILL POWER WILL BE PLOTTED ON THE X AXIS.
C
C
        EEX(1)=0.0
        EEX(2)=1.0
        EEY(1)=0.0
        EEY(2)=1.0
C
C
        DO 15 I=1,16
        DO 14 J=1,41
        N=(I-1)*41+J
        EY(N)=PP(I,J)/16810.0
        EX(N)=N/1385.0
     14 CONTINUE
     15 CONTINUE
        DO 16 J=1,36
        I=17
        N=(I-1)*41+J
        EY(N)=PP(I,J)/16810.0
        EX(N)=N/1385.0
     16 CONTINUE
C
C
        DO 17 J=37,41
        I=17
        N=(I-1)*41+J
        L=N-692
        EYY(L)=PP(I,J)/16810.0
        EXX(L)=N/1385.0
     17 CONTINUE
        DO 19 I=18,33
        DO 18 J=1,41
        N=(I-1)*41+J
        L=N-692
        EYY(L)=PP(I,J)/16810.0
        EXX(L)=N/1385.0
     18 CONTINUE
     19 CONTINUE
        DO 20 J=1,31
        I=34
        N=(I-1)*41+J
        L=N-692
        EYY(L)=PP(I,J)/16810.0
        EXX(L)=N/1385.0
     20 CONTINUE
        WRITE(6,606) NCOUNT
    606 FORMAT(//,25X,'NCOUNT EQUALS   ',1I6)
        WRITE (6,100) NUMEXC,NUMEDC,NUMINC,TOTCEL
    100 FORMAT (/,10X,'NUMEXC=',1I6,5X,'NUMEDC=',1I6,5X,'NUMIN
       XC=',1I6,5X,'TOTCEL=',1I6)
        WRITE(6,101) CRITE3,CRITE1,CRITE2
    101 FORMAT (//,10X,'NUMBER OF POSITIVE EDGE CELLS IS   ',1I
       X6,//,10X,'THE PERCENTAGE OF POSITIVE CELLS WHICH ARE E
       XDGE CELLS IS   ',1F4.3,//,10X,'THE AMOUNT OF PROBABILIT
       XY IN THE EDGE CELLS IS   ',1F4.3)
        CALL DRAW (2,EEX,EEY,1,0,LABEL,ITITLE,0,0,0,0,0,0,9,9,
       X0,LAST)
        CALL DRAW (692,EX,EY,2,0,LABEL,ITITLE,0,0,0,0,0,0,9,9,
       X0,LAST)
        CALL DRAW (692,EXX,EYY,3,0,LABEL,ITITLE,0,0,0,0,0,0,9,
       X9,0,LAST)
        STOP
        END
```

```
      SUBROUTINE RANDU(IX,IY,YFL)
      IY=IX*65539
      IF(IY)5,6,6
    5 IY=IY+2147483647+1
    6 YFL=IY
      YFL=YFL*.4656613E-9
      RETURN
      END

      SUBROUTINE TRKTM(ET,R,T)
      RA=1.0-R
      T1=-ET*ALOG(R)
      T2=-ET*ALOG(RA)
      T=T1
      RETURN
      END

      SUBROUTINE MOVE(VEL,TIME,ANG,X,Y)
      X=VEL*TIME*COS(ANG)
      Y=VEL*TIME*SIN(ANG)
      RETURN
      END

      SUBROUTINE HISTO(K,DELTAX,R,CORR,X,I)
      UPBND=((2*K+1)*DELTAX)/2.0
      L=2*K+1
      XX=(X+(CORR*UPBND))/DELTAX
      XX=CORR*XX
      X1=AMOD(XX,1.0)
      M=INT(XX)
      IF(M.EQ.0)GO TO 1
      IF(M.EQ.L)GO TO 2
      IF(X1.NE.0.0)GO TO 3
      IF(R.GT.0.5)GO TO 3
      I=M
      GO TO 4
    1 I=1
      GO TO 4
    2 I=L
      GO TO 4
    3 I=M+1
    4 RETURN
      END

      SUBROUTINE RESOLV(I,J,L,M)
      IF(I.GT.21)GO TO 2
      IF(J.GT.21)GO TO 1
      L=I+1
      M=J+1
      GO TO 4
    1 L=I+1
      M=J-1
      GO TO 4
    2 IF(J.GT.21)GO TO 3
      L=I-1
      M=J+1
      GO TO 4
    3 L=I-1
      M=J-1
    4 RETURN
      END
```

```
      SUBROUTINE CORSE(R,C)
      PI=3.1415926
      A=0.0
      B=2.0*PI
      F=2.0*PI
      DELTAC=R*(B-A)+A
      C=C+DELTAC
      C=AMOD(C,F)
      RETURN
      END
```

```
      SUBROUTINE TCORSE(R,C)
C     R IS THE UNIFORM VARIATE USED TO GENERATE THE NEXT
C     COURSE CHANGE.
C
C     C IS THE PRESENT COURSE AND WILL BE RETURNED AFTER
C     ADDING A NEW HEADING CHANGE.
C
C     DELTAC IS THE AMOUNT OF THE NEW HEADING CHANGE
C
      S=0.20
C
C     S IS THE TEEPEE SLOPE AND IS BETWEEN 0 AND
C     2/(PI**2) = 0.202.  NOTE , S CANNOT EQUAL 0.
C
      PI=3.1415927
C
      IF(0.0.LE.R.AND.R.LE.0.25)GO TO 1
C
      IF(0.25.LT.R.AND.R.LE.0.5)GO TO 2
C
      IF(0.5.LT.R.AND.R.LE.0.75)GO TO 3
C
      IF(0.75.LT.R.AND.R.LE.1.0)GO TO 4
C
C
    1 A=S/2.0
      B=(1.0/(2.0*PI))-(S*PI/4.0)
      D=-1.0*R
      E=B**2-4.0*A*D
      DELTAC=(-1.0*B+SQRT(E))/(2.0*A)
      GO TO 5
C
C
    3 RPRIME=R-0.5
      A=S/2.0
      B=(1.0/(2.0*PI))-(S*PI/4.0)
      D=-1.0*RPRIME
      E=B**2-4.0*A*D
      DELTAC=(-1.0*B+SQRT(E))/(2.0*A)
      DELTAC=DELTAC+PI
      GO TO 5
C
C
    2 A=(-1.0*S)/2.0
      B=(1.0/(2.0*PI))+(3.0*S*PI/(4.0))
      D=-1.0*((S*PI**2)/4.0)-R
      E=B**2-4.0*A*D
      DELTAC=(-1.0*B+SQRT(E))/(2.0*A)
      GO TO 5
C
C
    4 RPRIME=R-0.5
      A=(-1.0*S)/2.0
      B=(1.0/(2.0*PI))+(3.0*S*PI/(4.0))
      D=-1.0*((S*PI**2)/4.0)-RPRIME
      E=B**2-4.0*A*D
```

```
      DELTAC=(-1.0*B+SQRT(E))/(2.0*A)
      DELTAC=DELTAC+PI
      GO TO 5
C
    5 C=C+DELTAC
C
      F=2.0*PI
C
      C=AMOD(C,F)
C
      RETURN
      END

      SUBROUTINE PCORSE(R,C)
      PI=3.1415926
      F=2.0*PI
      S=1.0/(PI**[illegible])
      IF(R.GT.0.5)GO TO 1
      A=2.0*R/S
      DELTAC=SQRT(A)
      GO TO 3
    1 A=-1.0*S/2.0
      B=(1.0/(2.0*PI[illegible]+(3.0*PI*S/2.0)
      D=-1.0*S*(PI**[illegible])-R
      E=(B**2)-4.0*A*D
      DELTC1=(-B+SQRT(E))/(2.0*A)
      DELTC2=(-B-SQRT(E))/(2.0*A)
      IF(PI.LT.DELTC1.AND.DELTC1.LE.F)GO TO 2
      DELTAC=DELTC2
      GO TO 3
    2 DELTAC=DELTC1
    3 C=C+DELTAC
      C=AMOD(C,F)
      RETURN
      END

      SUBROUTINE LRCORS(R,C)
      PI=3.1415926
      F=2.0*PI
      IF(R.GT.0.5)GO TO 1
      C=C+(PI/2.0)
      GO TO 2
    1 C=C+((3.0*PI)/2.0)
    2 C=AMOD(C,F)
      RETURN
      END

      SUBROUTINE TRNCRS(R,C)
      PI=3.1415926
      F=2.0*PI
      A=PI/6.0
      B=(11.0*PI)/6.0
      DELTAC=R*(B-A)+A
      C=C+DELTAC
      C=AMOD(C,F)
      RETURN
      END
```

BIBLIOGRAPHY

[1] Washburn, A.R., "Probability Density of a Moving Particle," ORSA Vol. 17 (1969), pp. 861-871.

[2] Washburn, A.R., "An Introduction to Evasion Games," United States Naval Postgraduate School, September, 1971.

[3] Isaacs, R., "A Game of Aiming and Evasion: General Discussion and The Marksman Strategies," RAND Meno RM-1385, 24 November 1954.

[4] Danskin, J.M., "A Helicopter Versus Submarine Search Game," ORSA Vol. 16 (1968), pp. 509-517.