A MONTE CARLO STUDY OF FAST NEUTRON KINETICS IN SMALL METAL ASSEMBLIES

by

George Fergus Flanagan

A Thesis Submitted to the
Graduate Faculty in Partial Fulfillment of
The Requirements for the Degree of
MASTER OF SCIENCE

Major Subject: Nuclear Engineering

Iowa State University
Ames, Iowa

1967

TABLE OF CONTENTS

I. INTRODUCTION

In the last fourteen years, the use of the pulsed neutron source techniques for the investigation of the kinetic behavior of neutrons in matter has been wide spread (25, 26, 27). Diffusion and slowing down parameters can be calculated from the measurement of the migration, thermalization, and absorbtion of neutrons as a function of time. The pulsed assemblies are of one of two general categories. The first consist of or include moderator material, whereas the second consist mainly of a non-moderating material usually of a metallic nature. The latter are generally referred to in the literature as fast assemblies. It is these fast assemblies which will be considered in this work.

In the study of fast material assemblies subjected to a neutron pulse, it becomes important to know the density of neutrons as a function of position, velocity, angle, and time in the medium. Solutions for the neutron density can be found by analytical methods such as the solution of the equation

$$\frac{1}{v}\frac{d\emptyset}{dt} = H\emptyset + S\delta(t)$$

This is the time dependent Boltzman equation for a neutron flux from an impulse source. For an exact solution, certain simplifying assumptions are required, and these assumptions then apply only to specific cases. Some of the usual assumptions made are those of a constant mean free path and isotropic elastic scattering.

The neutron density in fast systems can also be measured experimentally. The experiments require sophisticated, expensive equipment, and few have been performed.

One way to tackle the problem of determining the density of neutrons in a fast assembly resulting from a pulsed source is the use of the Monte Carlo method. The probabilistic nature of neutron interactions makes the Monte Carlo technique useful.

This method was used in the investigations which are presented in this thesis. The computer code "PULSE" written by A. E. Profio (18, 19) was utilized for the computations with some modifications and alterations dictated by (a) the need to modernize and complete the code and (b) by the specific requirements of the IBM 360 model-50 computer available at Iowa State University.

Small heavy metal assemblies were used in the investigation. The geometrical shape of the assembly can be either a rectangular block, a cylinder, or a sphere. The purpose was to find the time dependence of the density and to investigate the possibility of expressing such a dependence as a simple decay constant. The results were also compared, whenever possible, with experimental results which are scarce but which are currently being investigated. With increasing emphasis on fast reactor systems, the investigation of die-away times in small uranium assemblies becomes important. Hence, uranium was the primary material in the assemblies considered.

Due to the small size of these assemblies, the neutron makes only a few collisions during a lifetime. The computation time using the Monte Carlo technique is then not too long, a fact that makes it attractive for application in such assemblies. In addition, the transport calculation may not be reliable in these cases if drastic simplifying assumptions are made. It seems therefore that the Monte Carlo method is well fitted for investigations of time dependent neutron density calculations in metallic assemblies of relatively small size.

II. THE MONTE CARLO METHOD AND ITS APPLICATION IN THE "PULSE" CODE

A. Historical Background

The Monte Carlo Method originated during the early 1940's as a result of suggestions advanced by J. von Neumann and S. Ulam at Los Alamos. However, virtually nothing appeared in print until about 1949. In that year, the first symposium on Monte Carlo was held at Los Angeles under the sponsorship of the Rand Corporation and the National Bureau of Standards in cooperation with Oak Ridge National Laboratory. The proceedings of this conference were published by the N.B.S. (17) in 1951.

A second symposium was held at the University of Florida in 1954. It was sponsored by Wright Air Development Center of the Air Research and Development Command. A. W. Marshall in the introduction to the proceedings of this second symposium (16, p. 4) says the following:

> "The most important practical applications thus far have had a probabilistic basis; the influence of the original Monte Carlo idea has been to suggest treating them directly as probabilistic problems rather than attempting a difficult, if not impossible, analytic solution. The translation and later retranslation of problems from probabilistic terms to non-probabilistic mathematical problems and back again has been bypassed."

There are many references which describe both theoretical and applied work that has been done in the field (2, 4, 5, 13, 14, 15, 16, 17) and no further background will be given here.

B. The Monte Carlo Method as Applied to the Physical Problem

Since neutron interactions within a material are described by neutron cross sections, which in essence are probabilities of interactions, the Monte Carlo technique can be applied to investigate the neutron transport process.

The problem which is to be solved here is, to find the number of neutrons leaking from the surface of an assembly as a function of time. The assembly is composed of one or more heavy metal isotopes. The neutrons arise from a neutron pulse occurring at time t = 0. The pulse of neutrons may be considered as incident on one face of the assembly as in the case of a cube or as generated inside the assembly as in the case of a sphere.

The Monte Carlo technique, as employed here, follows one neutron at a time through the assembly. The neutron's path length between interactions, and the type of interactions (fission, capture, or scattering) which it undergoes is determined by the material's neutron cross sections and angular distributions for various reactions. These data must be obtained by experiment or theory for the materials in the assembly and must be supplied to the code by the user.

The individual neutron is followed until it crosses the boundary of the assembly, or is absorbed, or is scattered over 100 times, or until its energy falls below a certain minimum. These latter two restrictions are used to prevent a neutron from being followed for too long and are not pertinent to the physical problem.

The above process is repeated over and over for a large number of neutrons, each of which produce a history. By combining the results of all histories, it is possible to approximate the actual physical behavior of the assembly under pulsed conditions. This probabilistic treatment does not have the generality of an analytical solution but it corresponds closely to the process of neutron interactions in matter which is probabilistic in nature.

The time dependence is incorporated into the Monte Carlo code by setting the time t equal to zero at the source. After calculating the distance d to the next collision from an exponential distribution of free paths about a mean free path, the time of flight t is calculated using

$$t = d/v$$

where v is the velocity of the neutron. If the neutron should leak out of the assembly, the neutron is placed at the boundary, and the distance D to the boundary from the last collision is found. The time of flight is then

$t = D/v$

In the event of fission the starting time for the new particle becomes the lifetime to that point of the original neutron. The time to exit, e.g. by leakage, absorption, falling below a certain minimum speed, by exceeding a specified maximum number of collisions, is printed in the output. The first time moment can then be calculated by

$$[t] = \sum_{i=1}^{I} t_i/I$$

where t_i is the lifetime of the i-th neutron and I is the total number of neutrons.

C. The "PULSE" Code

1. General description

A Monte Carlo Code named "PULSE" to handle the physical processes described above was written by A. E. Profio (18, 19). This code was used in this work with certain modifications added, including the translation from Fortran II to Fortran IV. The program is listed in Appendix F.

A. E. Profio sums up the usefulness of the Monte Carlo technique when he states (19, p. 1)

> "The use of straight analog Monte Carlo is feasible because the program is designed for small highly absorbing systems excited by fast neutrons, where the neutron makes only a few collisions on the average. The time of computation is essentially

> proportional to the mean life time in the medium, and computations of long lifetime systems is restricted by the computation center."

Since the laws of scattering, absorbtion, fission, and their cross sections are known for a single reaction (microscopic level) the "PULSE" code follows each individual neutron through the fast assembly until it is absorbed or leaks out. It does this for a large number of neutrons, giving a statistical approximation to what physically can be expected to happen when a burst of neutrons from a pulse source enters a fast assembly.

An overall description of the program follows with specific details related in the Appendices.

A simplified flow diagram is included in Figure 1 for reference. The MAIN program first reads each data card and prints the information for future reference. These data cards include neutron source coordinates, atomic density (of either one or two materials or isotopes), the microscopic cross sections, fission velocities, limiting velocities, anisotropic distributions for elastic scattering, and other specific data required by the program. The MAIN program multiplies the microscopic cross sections by the atomic density before storing and printing them. The MAIN also starts the computation of each neutron history and continues until all the source histories are run. It then checks to see if fission neutrons are present. These are

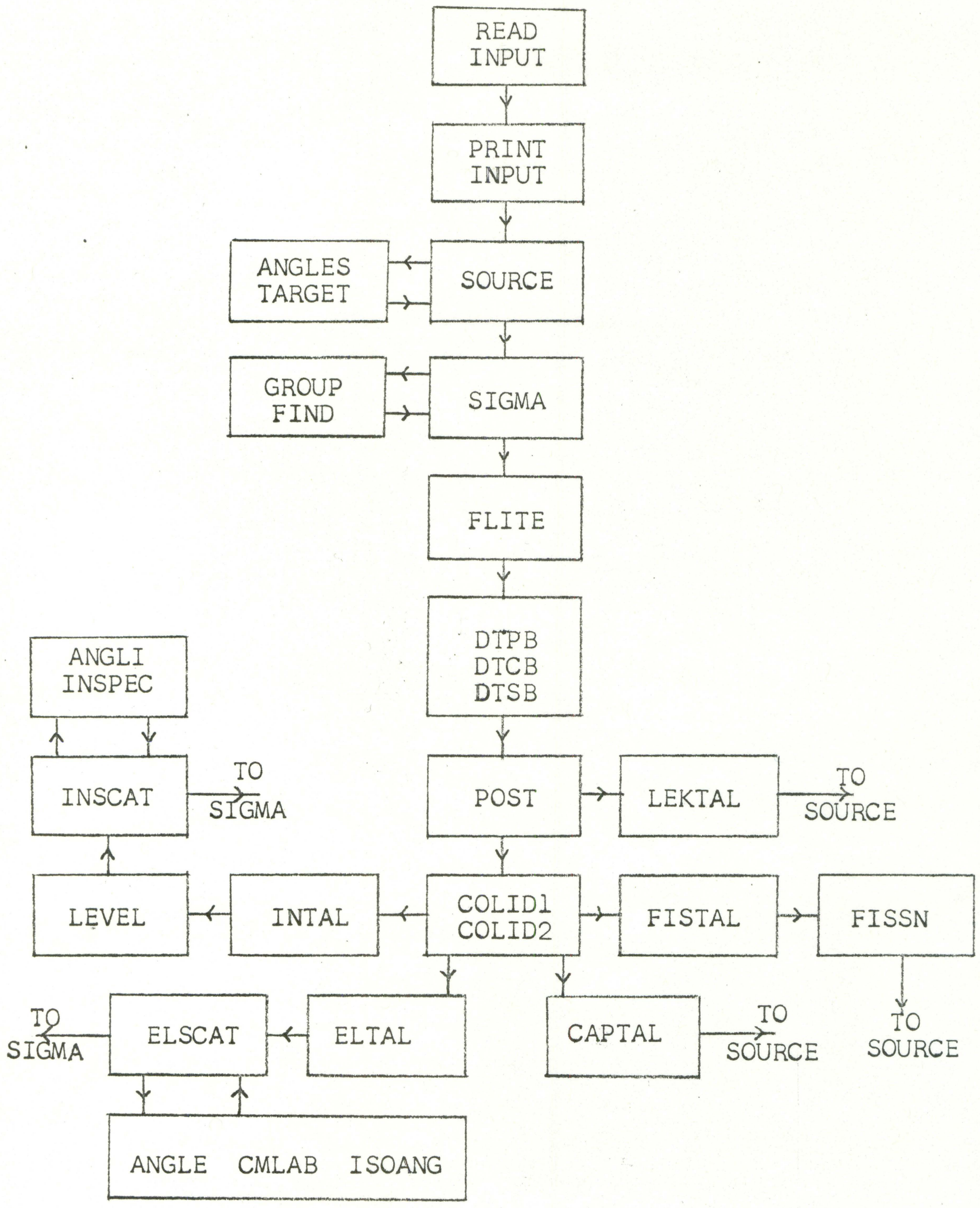


Figure 1. "PULSE" flow diagram

a result of fission taking place, and each of them is followed as if they were generated by the source.

Considering only source neutrons now, the MAIN begins the computation by calling the subroutine SOURCE. This is provided input which includes the type of source (plane, point within the target, point outside the target, and an option for including a configuration of the user's choice). This routine returns the x, y, z coordinates, velocity, time, and direction cosines of the neutron.

The MAIN now calls the subroutine SIGMA. SIGMA calculates the cross sections for elastic scattering, inelastic scattering, fission, and capture for each nuclide present, the total mean free path, and the probabilities for elastic, inelastic, fission, and capture interactions. The description of the methods used for calculation of these probabilities is given in Appendix C. SIGMA also calls an auxilliary subroutine GROUP to determine in which of a possible twenty groups, the velocity lies. The velocities at the lower limit of each of these groups is included in the input in units of 10^9cm/sec. These limits have been chosen arbitrarily in the range of 0.3-2.8 MeV. SIGMA also uses the subroutine FIND to linearly interpolate between the cross sections. These cross sections were input at each velocity group boundary mentioned above. Control is returned to the MAIN program which calls FLITE.

FLITE uses a pseudorandom number generated by the routine RANDU provided by the I.S.U. Computation Center (12). A detailed explanation of this routine is included in Appendix A. This pseudorandom number is used to select an exponential distribution of free paths which a neutron will travel before suffering another collision. FLITE checks the pseudorandom number to see if it is less than 0.0000454, and if it is another pseudorandom number is generated. This corresponds to the rejection of any free paths greater than 10 times the mean free path calculated in SIGMA. In addition a time variable (ITIME) is computed by dividing the free flight distance (DIST) by the velocity (VEL). Control is again returned to the MAIN program.

The MAIN now calls the subroutines DTPB, DTCB, or DTSB. These subroutines compute the distance to the nearest boundary of a plane, cylinder, or sphere respectively. Only one of these is called depending on the shape of the target specified in the input. The MAIN now calls POST.

POST compares the distance to the nearest boundary with the mean free flight distance (DIST) to see if the neutron is within the boundaries of the assembly. If it is not, the time of flight is updated by a quantity equal to the distance to the nearest boundary (DISTB) divided by the velocity of the neutron (VEL) and control is returned to the MAIN. If the neutron is within the boundary the time is updated by the quantity (DIST/VEL) and control is the returned to the MAIN program.

The MAIN now decides, using the information from POST, if the neutron leaked out or is still in the assembly. If it leaked out the subroutine LEAKTAL is called. This routine sets up a two dimensional array which categorizes the neutron according to its energy when it leaks out and the time since the neutron left the source. The MAIN adds one to tally of the number leaking out of the target (NL). If the neutron is within the boundaries the main calls COLIDI of COLID2 depending on the number of isotopes or elements present in the target.

COLID1 or COLID2, hereafter referred to as COLIDX, causes a pseudorandom number to be generated. Using this number the type of interaction (elastic, inelastic, fission, or capture) is determined as is the nuclide which took part in the reaction if more than one nuclide or isotope is present in the target assembly. Control is returned to the MAIN which calls the appropriate subroutine ELSCAT, INSCAT, FISSIN, or CAPTAL depending on the type of interaction determined by the COLIDX subroutine. The method employed for determination of the type of interaction is further explained in Appendix C.

ELSCAT is called if the reaction determined by the COLIDX subroutine is elastic scattering. This routine compares the velocity with an input parameter to determine if the scattering was isotropic or anisotropic. If isotropic, the center of mass direction cosine (GAMMAC) is computed by the formula CAMMAC=2R-1 where R is a pseudorandom number generated by RANDU. Now a check is made of the mass of the scattering nuclide to see if a conversion from the center of mass to laboratory system is necessary. This is done by comparing the atomic mass of the target nuclide to that specified by the input constant (ALIMX). If a conversion is to be made a subroutine CMLAB is called and the new velocity and direction cosines are computed as described in Appendix E. If no conversion is necessary ISOANG is called. This is a subroutine which computes the new direction cosines (alpha, beta, and gamma). If the scattering is anisotropic a subroutine (ANGLE) is called which computes a new direction cosine (GAMMA) from an angular distribution which is provided as input. Appendix E describes the methods used for determining such data from a given distribution. The subroutine ELTAL is then called which tallies the neutron in a two dimensional array. One dimension is time, and the other is space. So the neutron is registered in a certain time interval and in a certain coordinate interval. Also, one is added to the tally (NS) which is a tally of the number scattered (elastically or inelastically).

If inelastic scattering has taken place, the routine (INSCAT) is called. Scattering is assumed isotropic for this work. If the velocity is such that discrete level scattering takes place, the subroutine LEVEL is used to determine the probabilities of scattering from each of the levels for each energy group. These are used to determine which level does the scattering and the new energy is the incident neutron energy minus the level energy.

For high energy incident neutrons, scattering is assumed to take place in the continuum region.

The level distribution in the continuum region can be described by the evaporation model of the nucleus (8, 23, 27). Here the value of the nuclear temperature is usually assumed to vary as $\sqrt{E}$, a quantity represented by VEL in the "PULSE" code. The distribution of the fixed energy E' for the scattered neutrons is calculated from

$$\begin{cases} \dfrac{E'\exp(-E'/\Theta)}{\int E'\exp(-E'/\Theta)dE'}, & 0 < E' < E \\ 0 & \text{elsewhere} \end{cases}$$

where Θ = nuclear temperature = constant $\sqrt{E}$. The value of Θ depends on the nuclide and may be found in the references (8, 23, 27).

Profio has reduced the above model for computer use in the subroutine INSPEC. The new velocity (V_r) is computed using a probable distribution for the quantity V_r/V_{max}.[1]

[1]Profio, A. E., General Atomic, San Diego, California. Input constants for "PULSE". Private communication. 1967.

V_{max} is the square root of E_{max}. E_{max} is an input constant (CIN) times the incident velocity (VEL). The new direction cosines are computed by the subroutine (ISOANG). Again, one is added to the tally (NS) and control is returned to the MAIN program.

If fission occurs the subroutine FISSN is called. The average number of fission neutrons is calculated by use of the following formula:

$$\bar{\nu} = \nu_f + \delta v^2$$

ν_f and δ are input constants corresponding to the particular nuclide present in the assembly. A whole number for $\bar{\nu}$ is then chosen with the help of a pseudorandom number. A cumulative probability table is used to determine the velocity of each of the fission neutrons. Their coordinates, velocity, and the times are recorded on tape for running after all the source neutrons have been run. Also, the whole number closest to the value $\bar{\nu}$ is added to the tally (NF) which is the number of fission neutrons. Control is returned to the MAIN for the continuation of the source histories.

If capture takes place CAPTAL is called and one is added to the array KAPT in the appropriate time interval.

When all the source histories have been run, the tapes containing the fission neutron data, mentioned previously in the discussion of FISSN, are rewound. The program runs using the data on the tapes instead of the source data. Any new neutrons are again recorded on tapes. These tapes are then rewound and above process continues until there are no more fission neutrons generated.

The MAIN now outputs the requested data and the program is ended.

Certain computational "tricks" have been incorporated in the program to economize on computer time. For example, the entire output is recorded after every 500 histories in addition to the final recording which occurs after all the source histories have been run. Thus, in case the program is dumped prior to the final output some information is salvaged. The above tricks may or may not be used and elimination of these will in no way interfere with the running of the program.

2. Output description

The output of the "PULSE" code consists of the following tallies:

NL	The number of neutrons leaking out of the target assembly
NC	The number of neutrons captured within the assembly
NS	The total number of scattering interactions, both elastic and inelastic

NF	The number of fission neutrons resulting from the fission reactions
NLTD	The number of interactions which took place in less than the specified time delay input constant (TD)
NGTR	The number of interactions which took place in time greater than 100 time intervals
NGZR	The number of neutrons which suffer elastic collisions and end up outside the range of the Z coordinate interval
NLME	The number of neutrons ending up with an energy less than a minimum specified energy
NGER	The number of neutrons ending up with energies greater than 10 energy intervals
NOSL	The number of neutrons suffering more than 100 scattering interactions and therefore dropped from the program

Also included in the output are the following arrays:

LEAK	A two dimensional array (time, energy) specifying the time and energy of each of the neutrons which cross the surface of the assembly
NELS	A two dimensional array (time, Z-coordinate) specifying the time and position of each of the neutrons whenever they suffer an elastic collision

NIMS	A one dimensional array (time) specifying the time for each inelastic collision
NFIS	A one dimensional array (time) specifying the time for each fission reaction
KAPT	A one dimensional array (time) specifying the time for each capture interaction

In addition to the above tallies and arrays, the variable ITOT is output after every 500 histories. ITOT is a running tally of the number of histories which have been run. In this way if the program should hang up or if the machine should fail the spot in the program can be determined where a failure occurred and the program can be resumed from there. Also, since all results are recorded on tape as well as printed, the variable ITOT will be the total number of histories retained on the output tape.

All input data are also printed out for reference as is the variable number which initiates the random number generating routine explained in Appendix A.

3. Input description

A number of input variables are required for the code "PULSE". The order of appearance in the data deck and a short description of each variable are given below. A more detailed description of these parameters can be found in two reports by A. E. Profio (18, 19). If the variable is an array, the dimensions of the array are given in parenthesis following the variable.

Card #1 XS, YS, ZS, PARA, PARB, PARC, THETA, KS, NEUT

XS, YS, and ZS are the source coordinates; PARA, PARB, and PARC specify the source velocity; THETA is the source time (usually 0.0); KS is a code integer giving the source option as mentioned in section II; and NEUT is the number of histories being run.

Card #2 SP (10)

SP is an array which specifies an anisotropic source distribution. It consists of value of the Cosine Θ. In this work, the source was considered isotropic and values of 1.0 were used for all the SP data.

Card #3 XMAX, YMAX, ZMAX, RMAX, KAS

XMAX, YMAX, ZMAX and RMAX give the dimensions of the assembly in units of cm. The first three are used if the assembly is rectangular, and RMAX is used if it is a cylinder or a sphere. KAS specifies the shape of the target (1-block, 2-cylinder, 3-sphere).

Card #4 TD, TCH, EMIN, ECH, KT1, KT2

TD is the time delay in the source; TCH is the time channel with; EMIN is the minimum tallied energy (MeV); KT1 and KT2 are tape number used in the FISSN routine which are supplied by the computation center.

Cards #5-6 P (20)

This is an array specifying a Maxwell-Boltzmann distribution for inelastic scattered velocities from the continuum. The values are normalized velocities for an index K.

Cards #7-8 VBOUND (20)

VBOUND is an array of velocities. The units are 10^9cm/sec. It is at each of these twenty velocities that the cross sections used in "PULSE" are evaluated.

Card #9 AD1, A1, ALIM1, SLIM1, CIN1, VST1, FNU1, DELNU1, KIA1

In the above the "1" following each variable signifies nuclide #1 in the target. AD1 is the atomic density (10^{24}/cm^3); A1 is the mass number; ALIM1 is the mass below which a center of mass to laboratory reference system conversion must be made; SLIM1 is the velocity above which anisotropic center of mass elastic scattering can be assumed to occur; CIN1 is a decimal number used in the routine INSPEC to determine the most probable velocity from the input velocity when inelastic scattering from the continuum is assumed (7, 19, 22, 23, 26); VST1 is the velocity below which individual level inelastic scattering occurs; FNU1 and DELNU1 are decimal numbers used in the FISSN routine (see section II); and KIA1 is a constant used to determine isotropic or anisotropic inelastic scattering (1-isotropic, 2-anisotropic).

Cards #10-11	SBE1 (20)
	This array consists of elastic cross sections ($10^{-24}cm^2$) evaluated at the velocities given in VBOUND. Again the "1" signifies that the values are for nuclide 1.
Cards #12-13	SBI1 (20)
	Included in this array are the inelastic cross sections ($10^{-24}cm^2$). Each evaluated at the velocities in VBOUND.
Cards #14-15	SBF1 (20)
	The values of these cards are thos of the fission cross sections ($10^{-24}cm^2$).
Cards #16-17	SBC1 (20)
	These are the cross sections for neutron capture ($10^{-24}cm^2$).
Cards #18-37	AP1 (10, 20)
	AP1 is a two dimensional array specifying the

angular distribution in anisotropic elastic scattering. The values are those of the cosine Θ for each of the twenty velocities given in VBOUND.

Cards #38-39	VL (20)
	VL is an array for up to twenty inelastic scattering level velocities (10^9cm/sec).
Cards #40-79	SBL1 (20, 20)
	SBL1(L,J) is a two dimensional array specifying the cross sections (10^{-24}cm^2) for inelastic level scattering where L is the level number, and J is the velocity group number from VBOUND.
Cards #80-81	FP1 (22)
	This array specifies fission neutron velocities (10^9cm/sec) and is used in the FISSN routine.
Card #82	This card contains the same variables as card #9 except that the values are for nuclide #2. If there is only one nuclide in the assembly zeroes are punched for the values on this card and it is then the third from the last card in the data deck.
Cards #83-154	These cards contain the variable data for nuclide #2. The arrangement is the same as for cards #10-81. If only one nuclide is used these cards are not needed in the data deck and are

therefore left out of the pack.

Card #155 IX

IX is a number of one to nine digits and must be odd. It is used to initiate the random number routine RANDU. This is always the second to the last card in the data deck and is needed regardless to the number of nuclides used.

Card #156 JJ

This variable is used to specify the number of different energies of the source neutrons. If a monoenergetic source is used, JJ is equal to 1. If a spectrum is used, there must be a card containing the same information as is contained on card #1 for each of the energy groups. These cards will follow this card in the data pack.

III. THE DATA USED IN THE COMPUTATIONS

Initially in this work, the material used in the assembly was U-238 with a monoenergetic (1MeV) source.

Uranium has an atomic density of $0.0472 \times 10^{24}/\text{cm}^3$ and a mass of 238. The 1 MeV neutrons have a velocity of 1.385×10^9cm/sec.

The angular distribution for inelastic scattering was assumed to be isotropic and this was confirmed using BNL-400 (10). The velocity was found to be isotropic below a velocity of 0.875×10^9cm/sec. Above this velocity the differential distributions in BNL-400 (10) for elastic scattering in U-238 at various energies were integrated. From the integrated curves, values for the array AP1 (anisotropic distribution for elastic scattering) were obtained as explained in Appendix E.

The first 16 values of the velocity group array VL1 range from 0.3 MeV to 1.6 MeV at 0.1 MeV intervals. The next 6 values are at 0.2 MeV intervals giving an energy range of 0.3 to 2.8 MeV.

The cross sections needed for input into the "PULSE" code included the elastic scattering, inelastic scattering, capture, and fission cross sections. In this work these need only be evaluated over the energy range 0.3 MeV to 2.8 MeV as 1 MeV monoenergetic source is used. However, the code can be run with a source energy spectrum, and in this case the energy range must be extended.

Various sources were used to obtain the best possible values for the cross sections.

For the capture reactions, the values used were from BNL-325 (11). These were compared to those given in ENDF/B[1] which were supplied by Brookhaven Sigma Center and were found to be in agreement.

For the fission values, BNL-325 was again used and these data correlated with those supplied by ENDF/B.

None of the references used listed the elastic scattering cross sections for the isotope U-238. Therefore, the values were taken from the natural uranium listings. In doing this one must assume most of the scattering is due to U-238. This is a reasonable assumption due to the fact that the concentration of U-235 in natural uranium is small, and its scattering cross section is small. The values were obtained by subtracting the non-elastic values from the total cross sections. BNL-325 was used again. There were no values for these values in ENDF/B.

The inelastic cross sections were obtained from the non-elastic values for the uranium cross sections. In doing this

---

[1]May, V. Brookhaven National Lab., Sigma Center, Upton, New York. ENDF/B nuclear cross sections. Private communication. 1967.

one assumed three possible non-elastic processes (fission, capture, and inelastic scattering). The fission and capture values were obtained as explained above, and their sum was subtracted from the non-elastic cross sections. The difference was taken as the value for inelastic scattering cross sections. Again, BNL-325 was used as the reference. Some of the values obtained agree with those in ENDF/B. This latter reference was far too incomplete to be of much value in this case except as a check for other sources.

For inelastic scattering from individual levels, there is much discrepancy as to the level energies, the number of levels, and the cross sections at each level. In the present work experimental data supplied by Dr. D. A. Lind[1] was used; these data were deemed as the most complete set. Some of these values obtained by Lind are in agreement with those in BNL-325. However, the latter contains an incomplete set of data and was not used as a reference for level scattering.

The fission spectrum for U-238 was taken to be the same as that of U-235. The spectrum used was taken from an article by R. L. Henkel (9). This was integrated and values for the array FP1 were obtained as described in Appendix E.

FNU1 and DELNU1 were obtained from ANL-5800 (22).

The constant CIN1 was obtained by private communication

[1]Lind, D. A., University of Colorado, Boulder, Colorado. Inelastic cross section levels for U-238. Private communication. 1967.

from A. E. Profio[1] with the aid of the data found in the references (8, 23, 24, 27).

The final run of this work was made with natural uranium. This metal consists of 99.3% U-238 and 0.7% U-235. Data were found for U-235 from the same sources as mentioned above for U-238.

[1]Profio, A. E., General Atomic, San Diego, California. Input constants for "PULSE". Private communication. 1967.

IV. RESULTS AND DISCUSSION

The primary purpose of this work, as stated earlier, was to investigate the leakage of neutrons following the injection of a fast pulse from metallic assemblies of various shapes by the Monte Carlo method.

Also, an investigation was made as to the possibility of expressing the leakage in the form

$$N = C\, e^{-\lambda t}$$

In the above expression, N is the number of neutrons leaking out of the assembly after time t; C is a constant; λ is a time delay constant with units of inverse time; and t is the time after the pulse injection. If the expression is valid, then log N plotted versus time should be a straight line with a slope λ.

As an attempt to investigate the afore stated postulates, three runs were made with a spherical assembly of U-238. A 1 MeV monoenergetic source of neutrons was assumed to be at the center of the spheres. The three runs were made with spheres 15 cm, 20 cm and 30 cm in diameter.

The resulting leakage from the sphere surface is shown plotted versus time in Figures 2, 4, and 6. The number of histories were 8,000, 7,000, and 8,000 for the 15 cm, 20 cm, and 30 cm spheres respectively.

A plot of the data was also made on log paper, and the least squares fit technique was applied assuming an equation of the form

$$\ln N = \ln C + \lambda t$$

The values found for λ were 0.305 nsec^{-1}, 0.179 nsec^{-1}, and 0.1475 nsec^{-1} for the 15 cm, 20 cm, and 30 cm diameter spheres respectively.

The logarithmic plots along with the line having the least squares fitted slope are shown in Figures 3, 5, and 7 for the 15, 20, and 30 cm diameter spheres.

A second set of three runs was made on a cube of U-238 with 1 MeV neutrons uniformly incident on one of the faces. The leakage is plotted as a function of time in Figures 8, 10, and 12 for the 15 cm, 20 cm, and 25 cm cubes. The number of histories for each were 8,000, 10,000, and 8,000 respectively.

Again, logarithmic plots were made as shown in Figures 9, 11, and 13, and a least squares fit was applied. The results were the straight lines shown in the above mentioned figures. The slopes are 0.1695 nsec^{-1}, 0.156 nsec^{-1}, and 0.101 nsec^{-1} respectively.

Finally one run was made using natural uranium. The assembly was a 15 cm diameter sphere with a 1 MeV source at the center. The leakage is plotted versus time in Figure 14 and the log plot showing a least squares fitted line with a

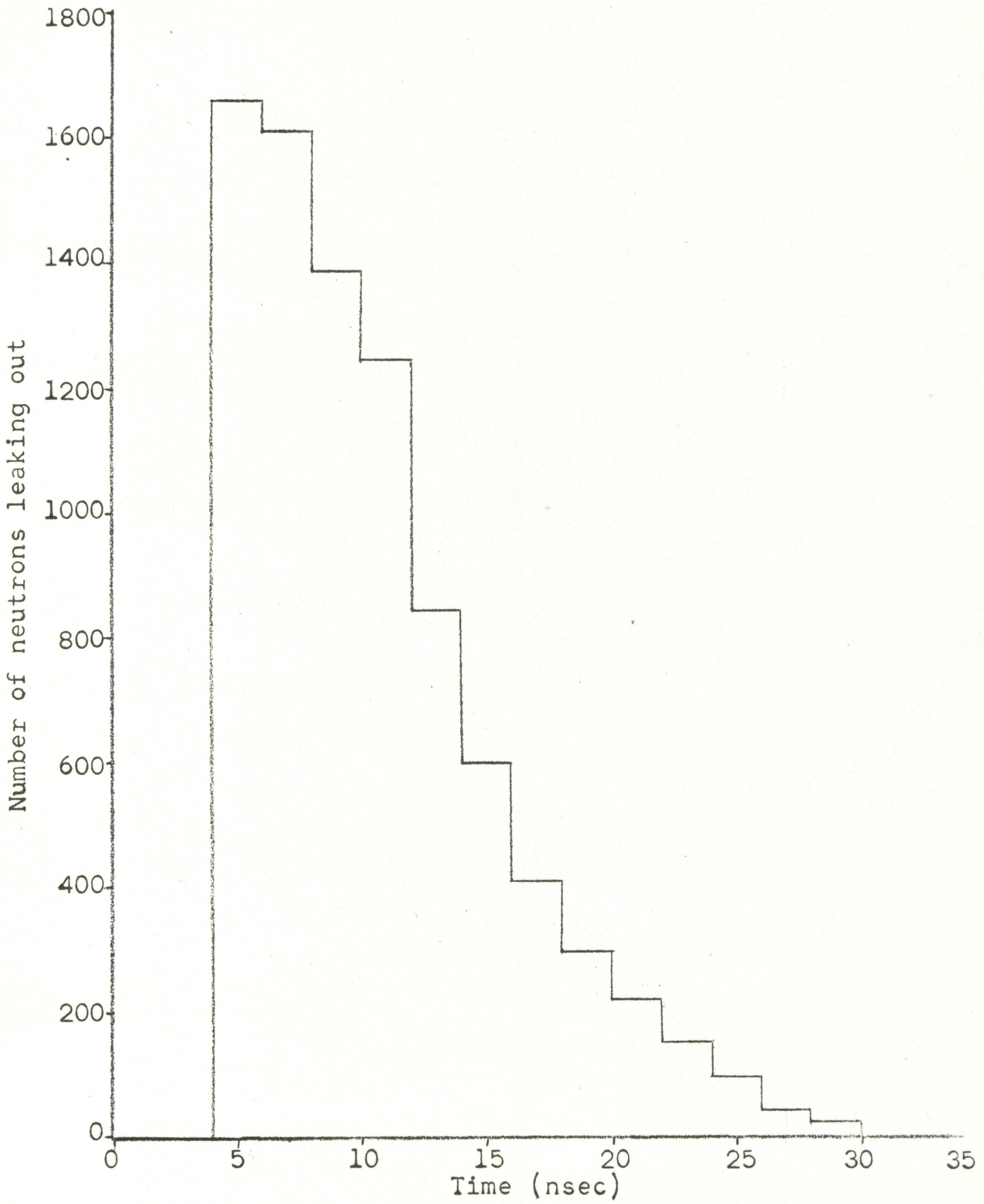


Figure 2. A 15 cm diameter U-238 sphere

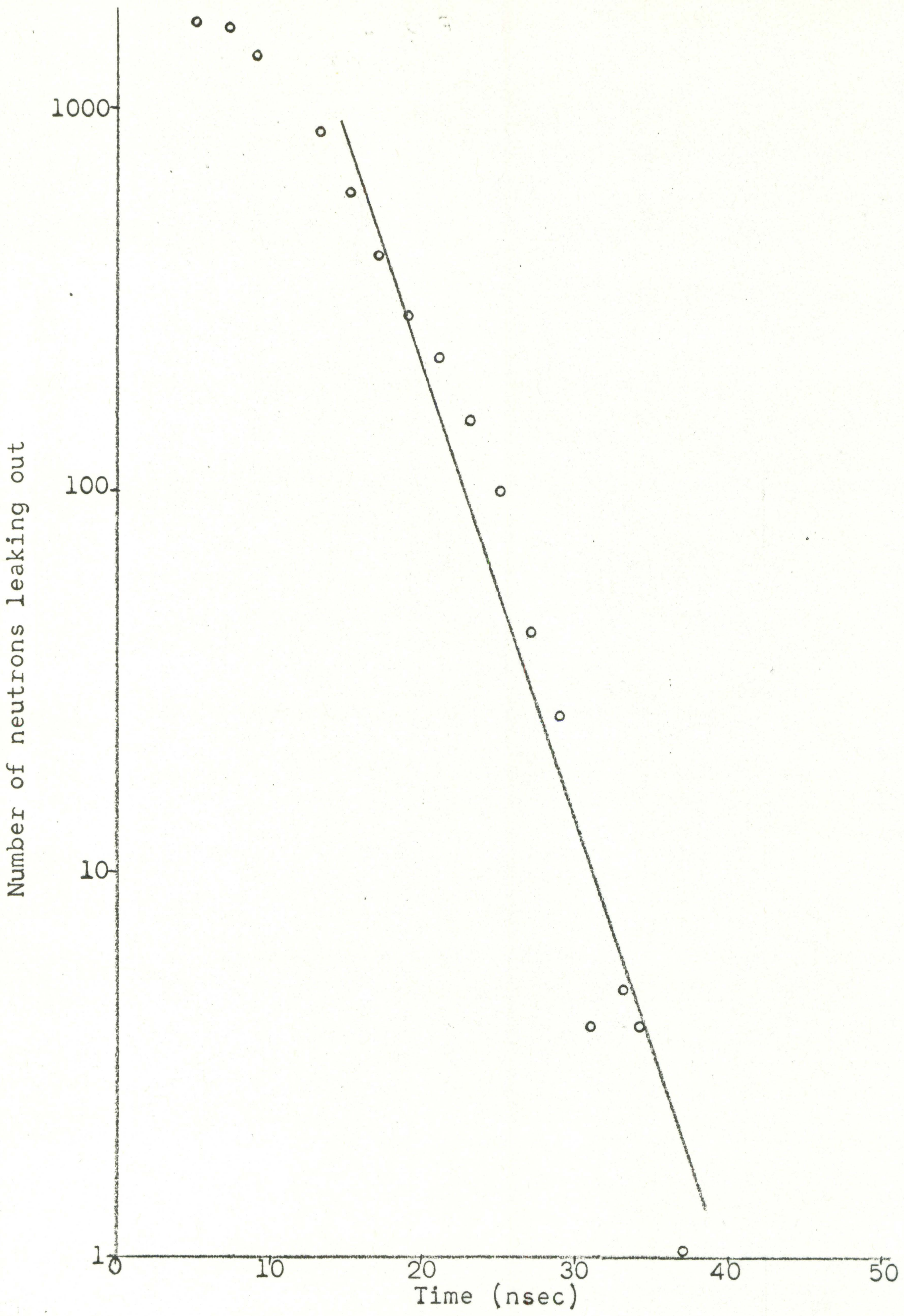


Figure 3. A 15 cm diameter U-238 sphere

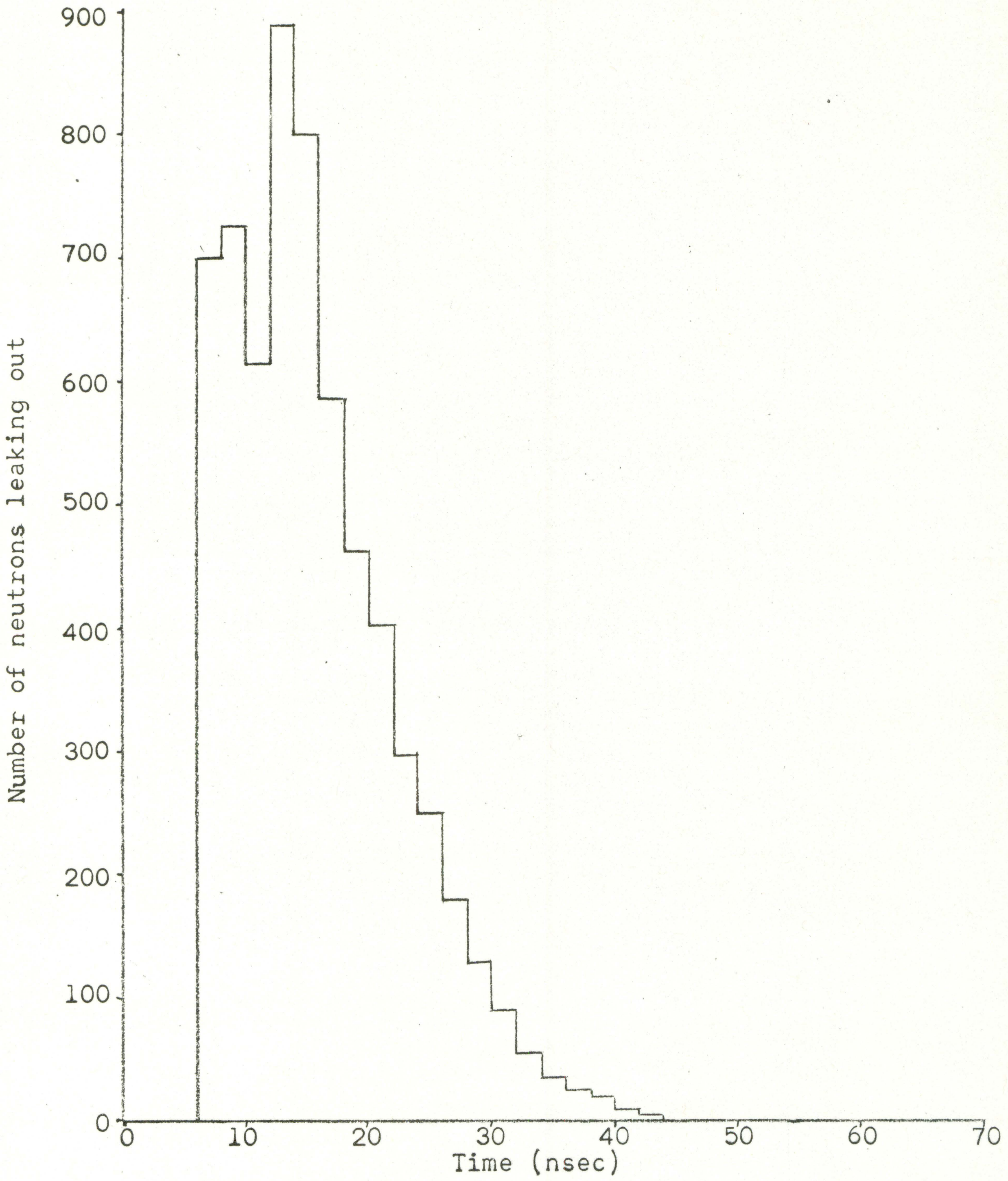


Figure 4. A 20 cm diameter U-238 sphere

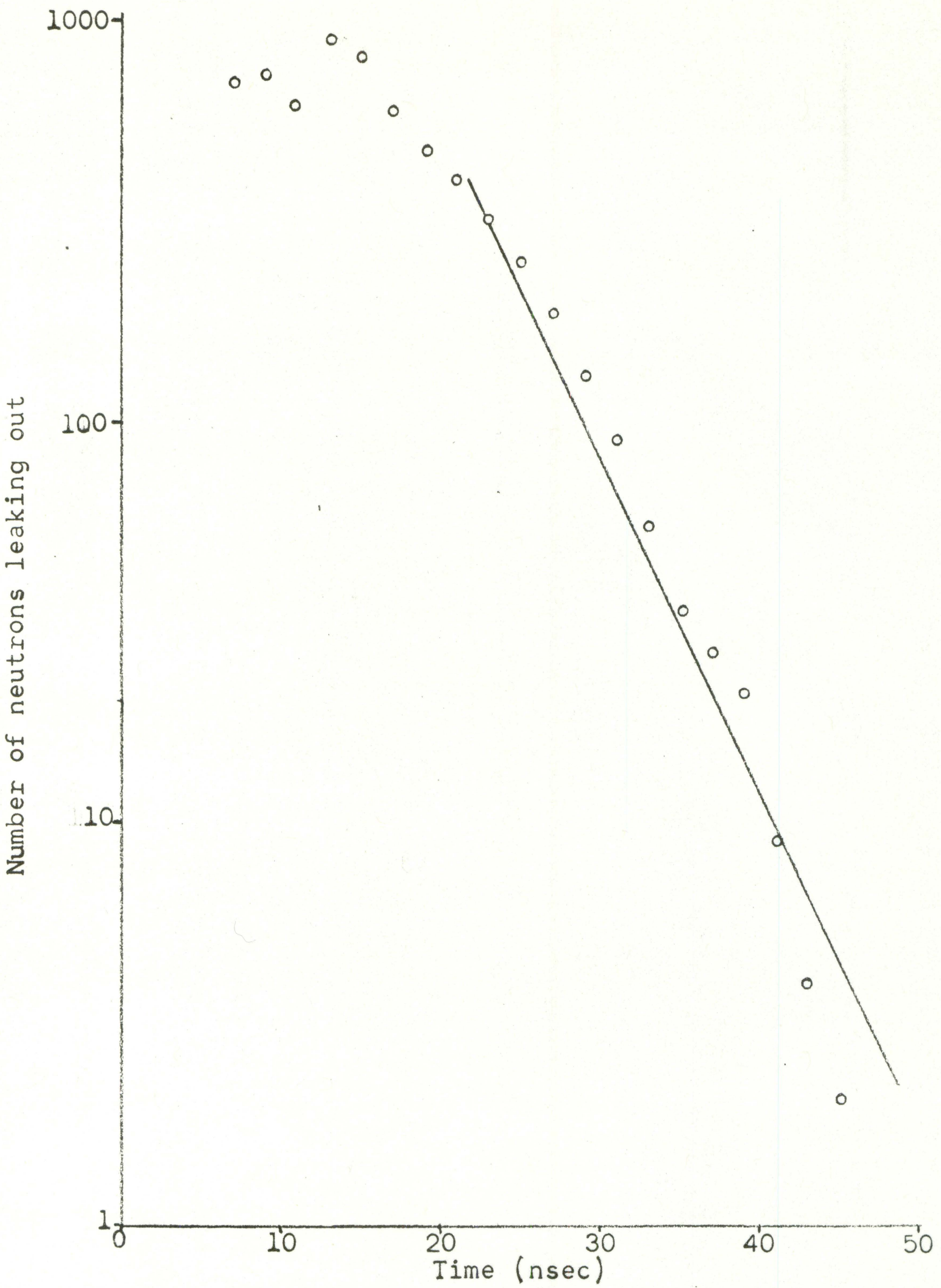


Figure 5. A 20 cm diameter U-238 sphere

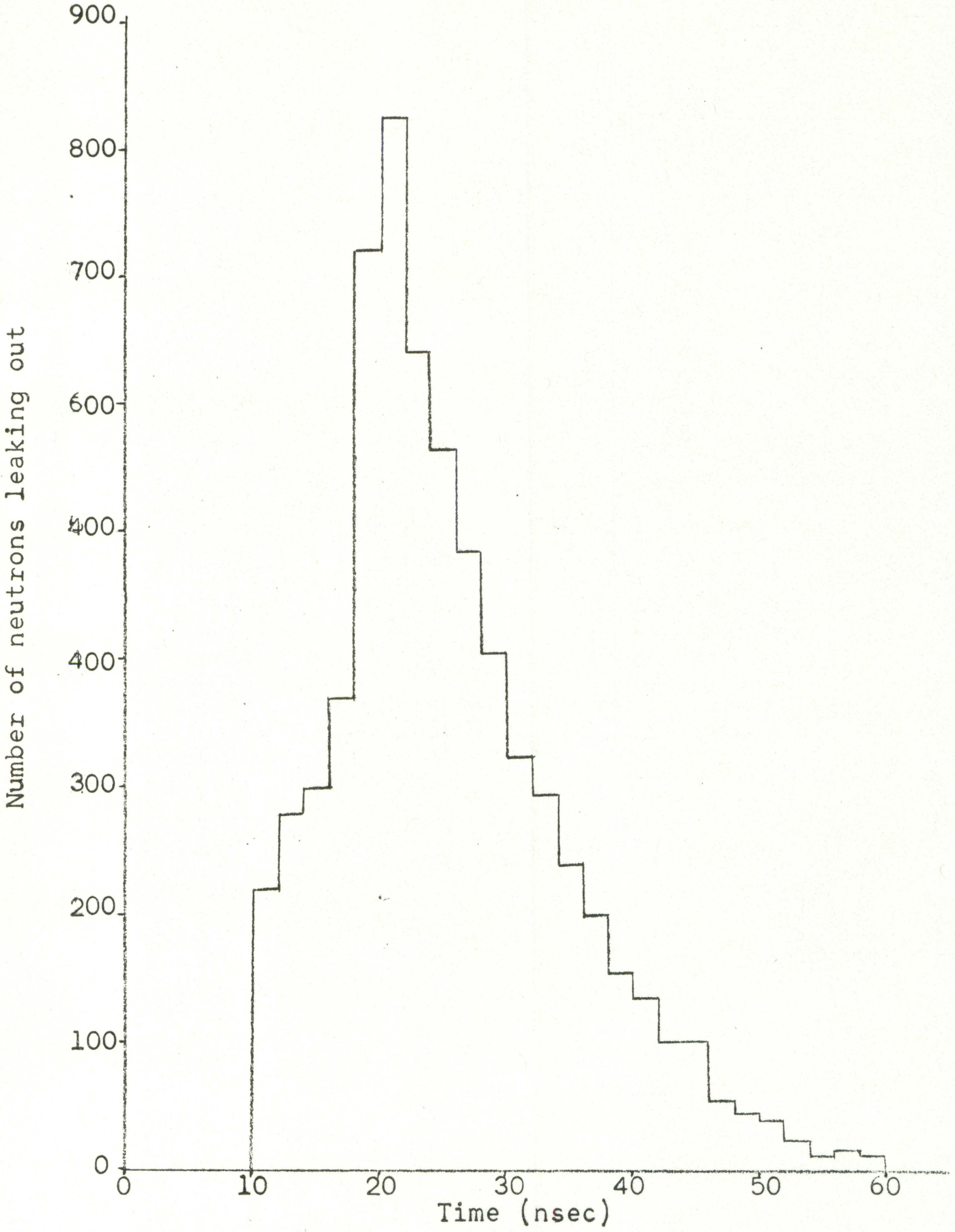


Figure 6. A 30 cm diameter U-238 sphere

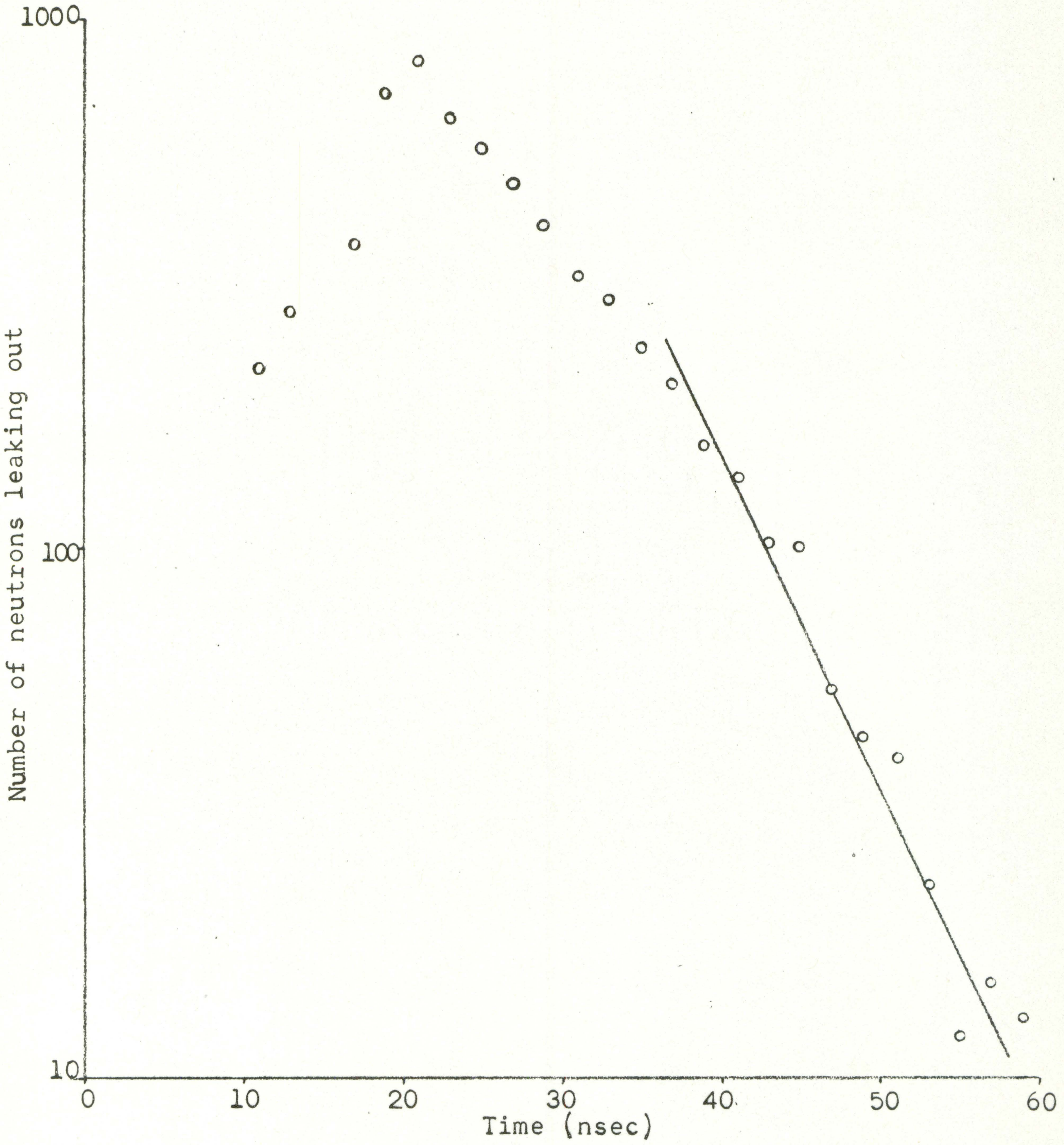


Figure 7. A 30 cm diameter U-238 sphere

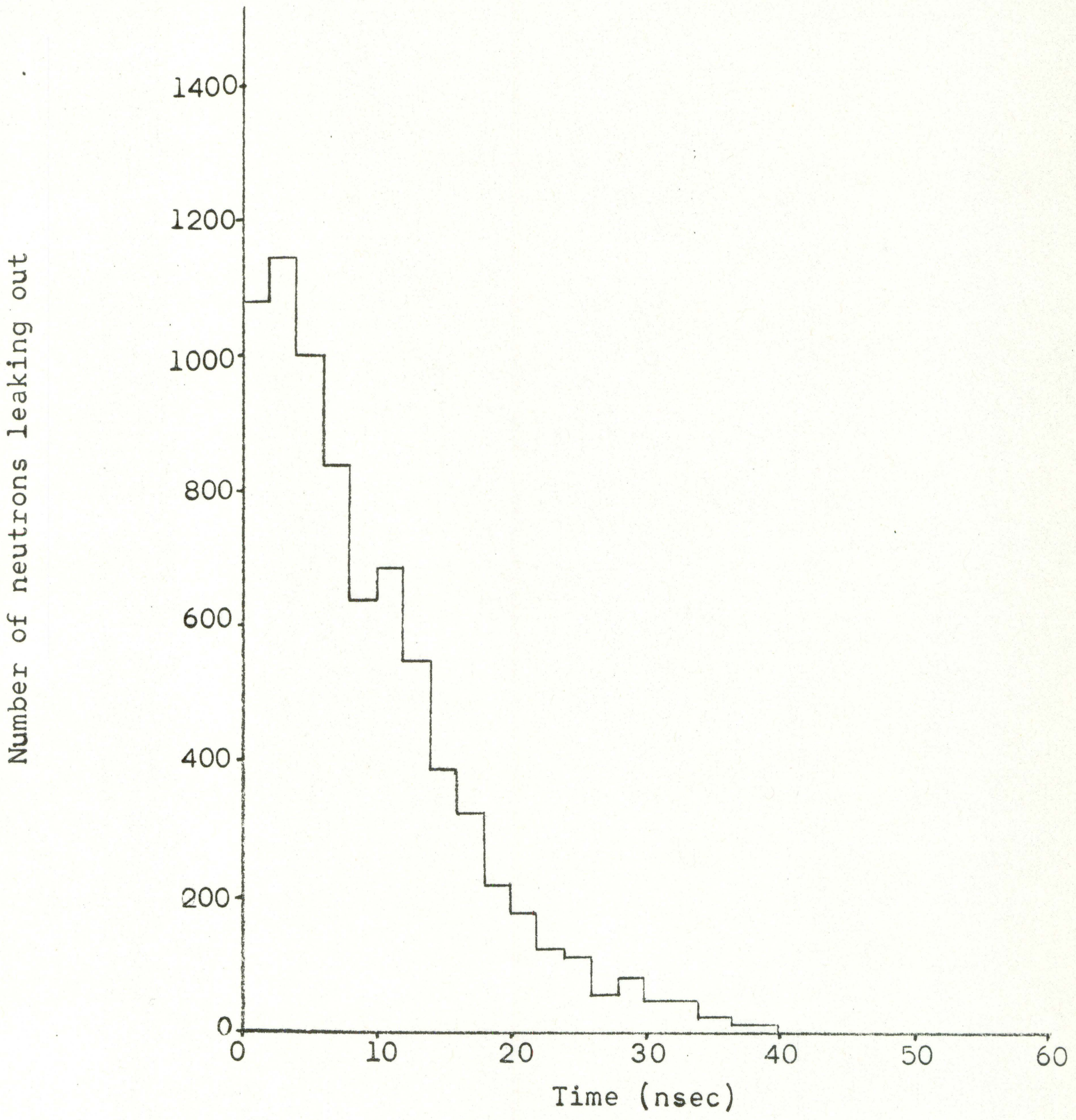


Figure 8. A 15 cm U-238 cube

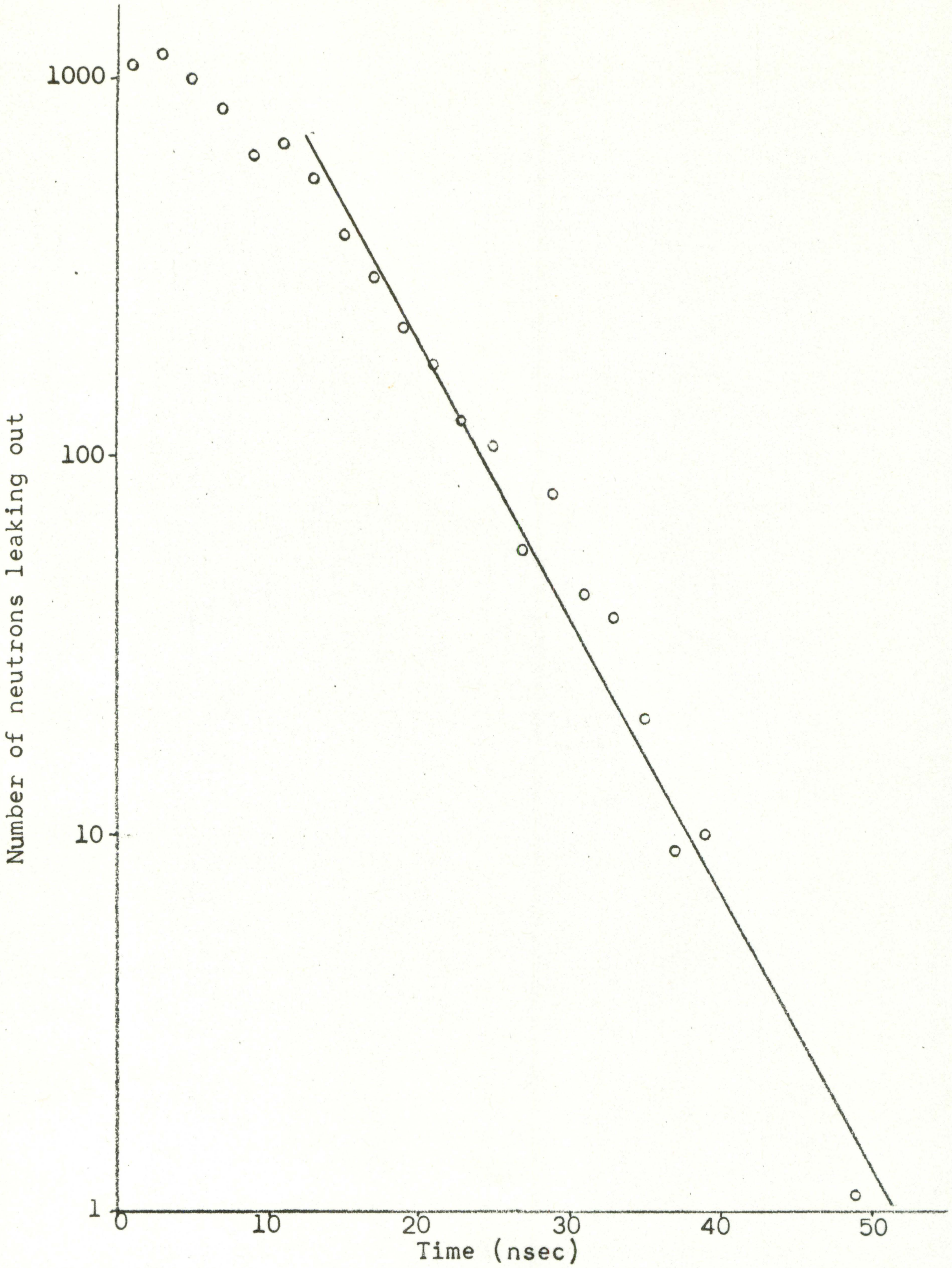


Figure 9. A 15 cm U-238 cube

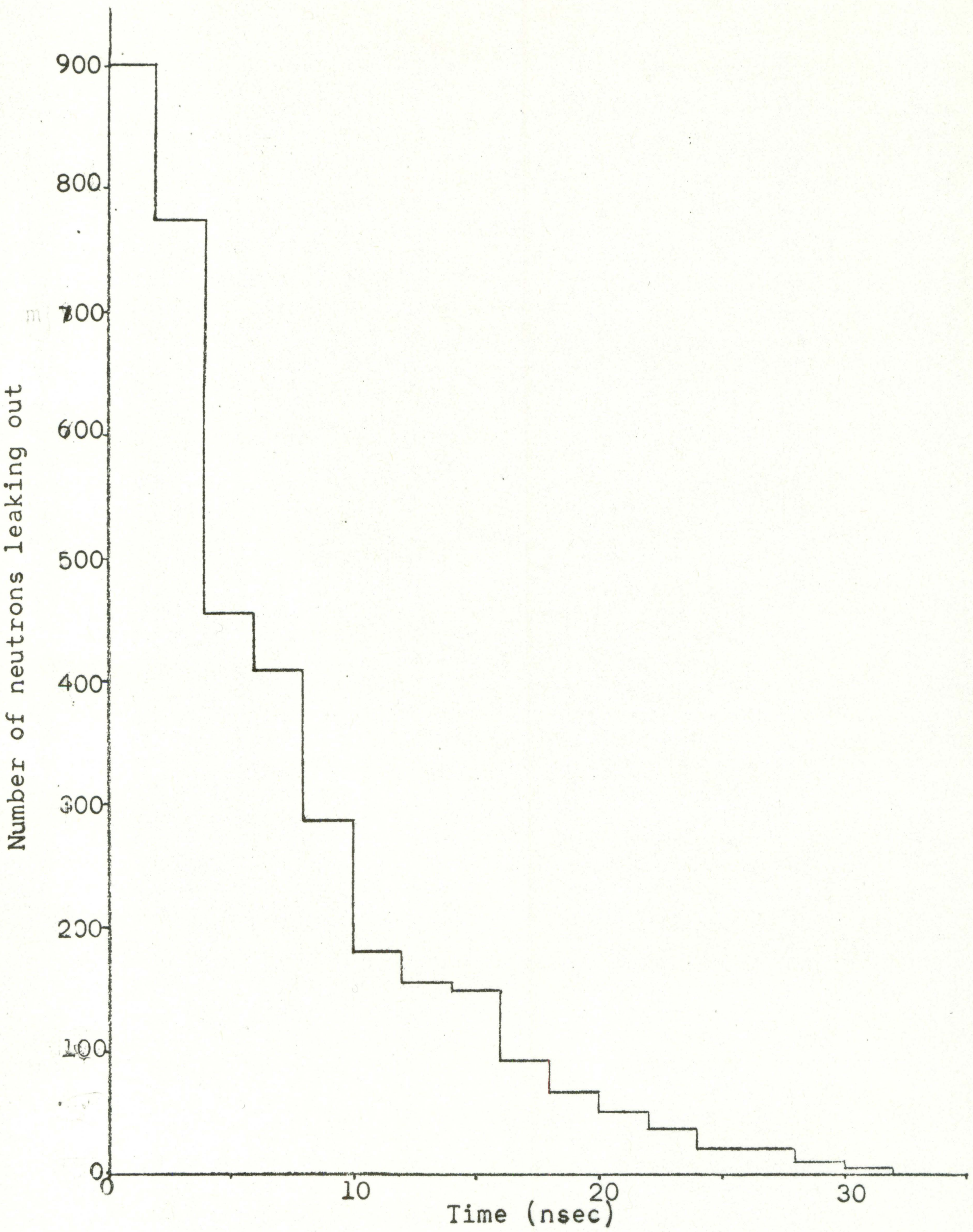


Figure 10. A 20 cm U-238 cube

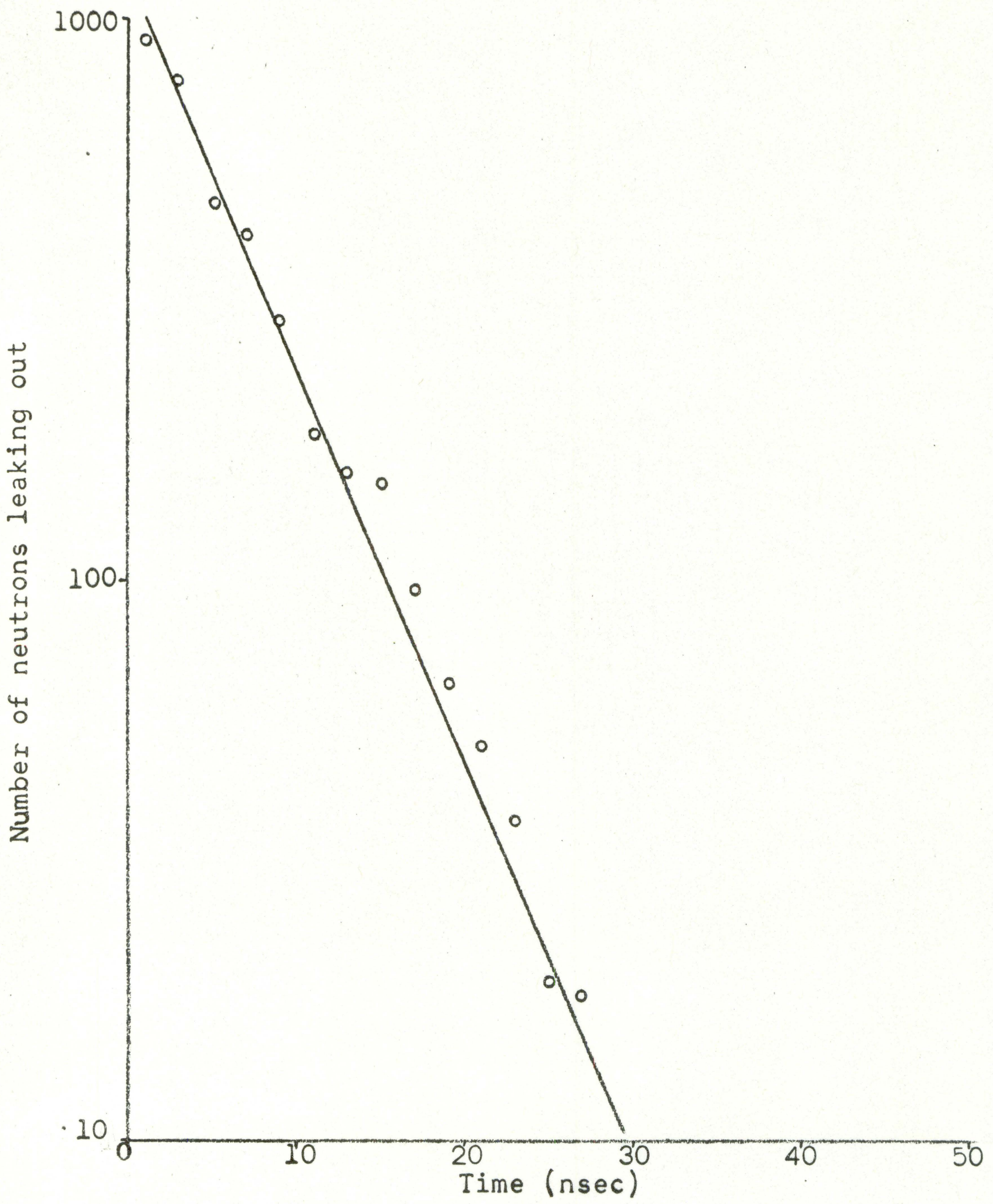


Figure 11. A 20 cm U-238 cube

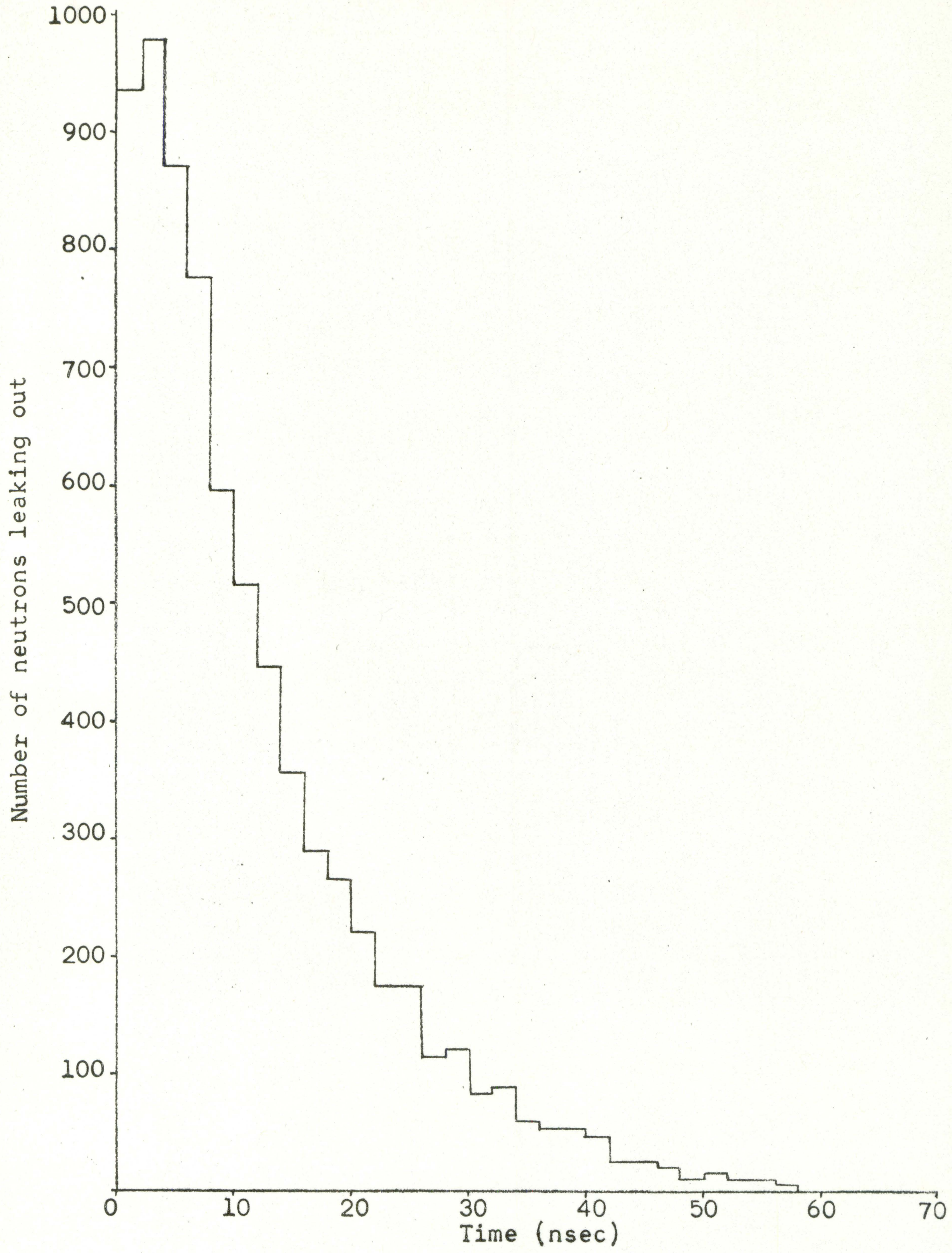


Figure 12. A 25 cm U-238 cube

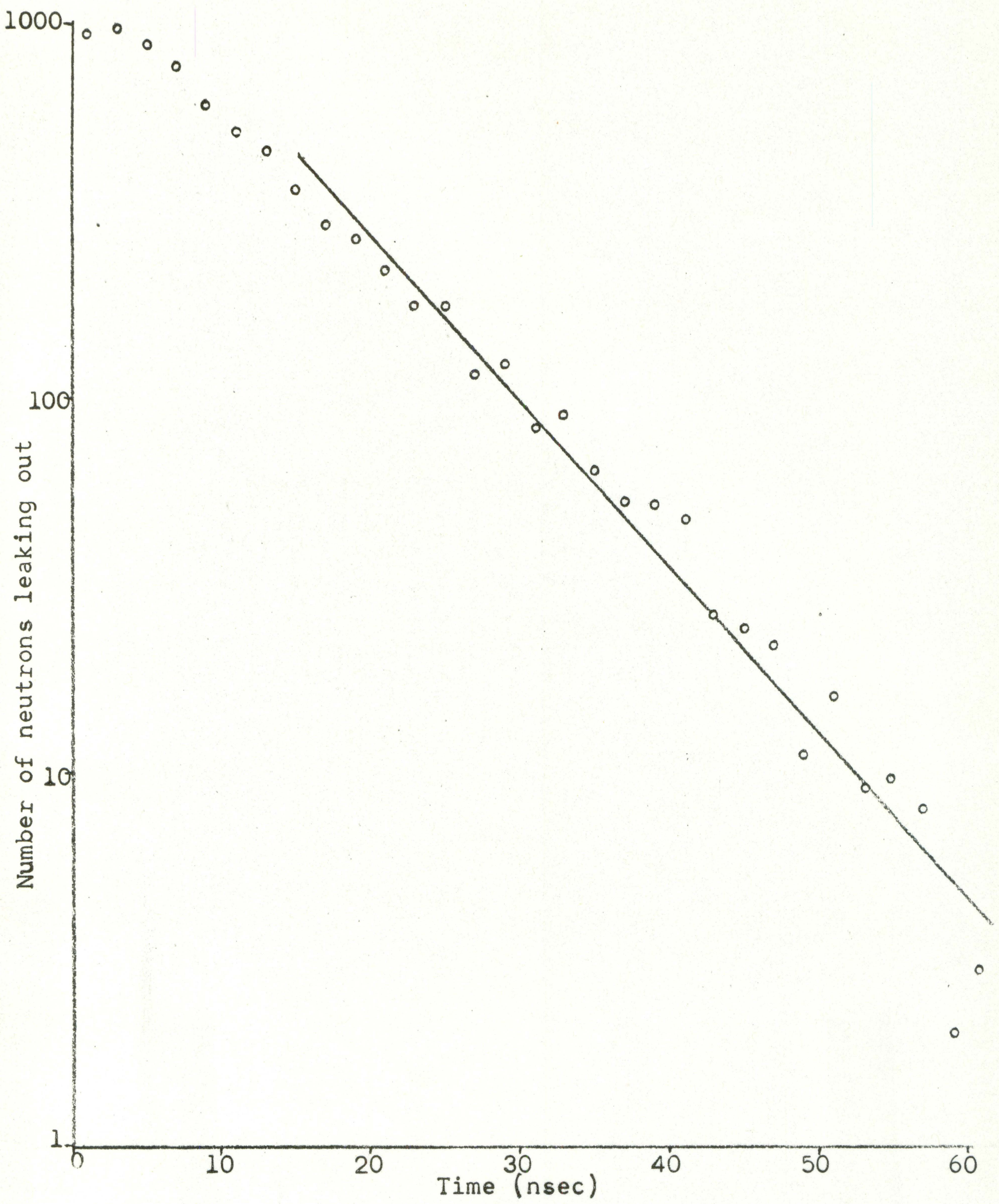


Figure 13. A 25 cm U-238 cube

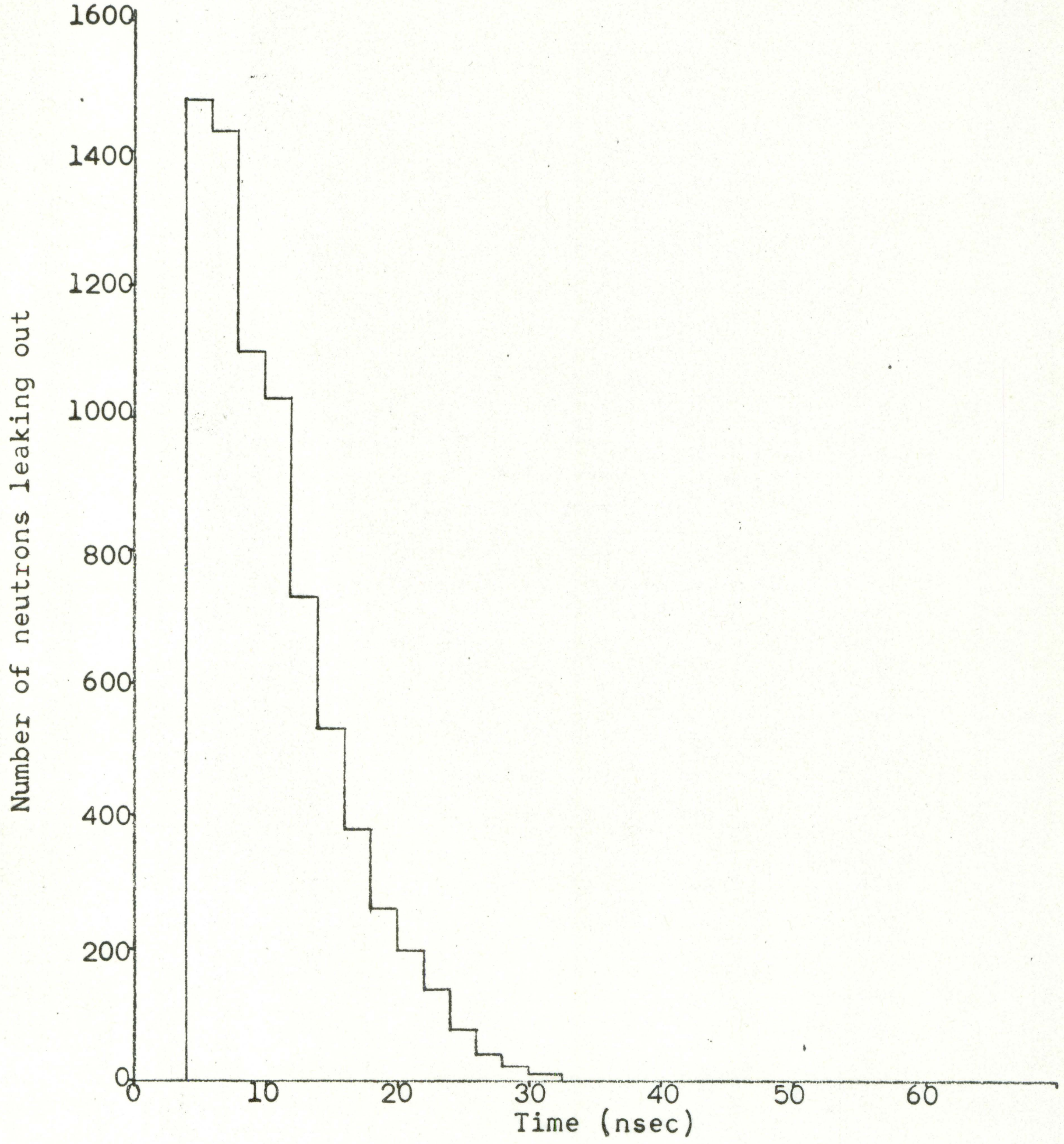


Figure 14. A 15 cm diameter natural uranium sphere

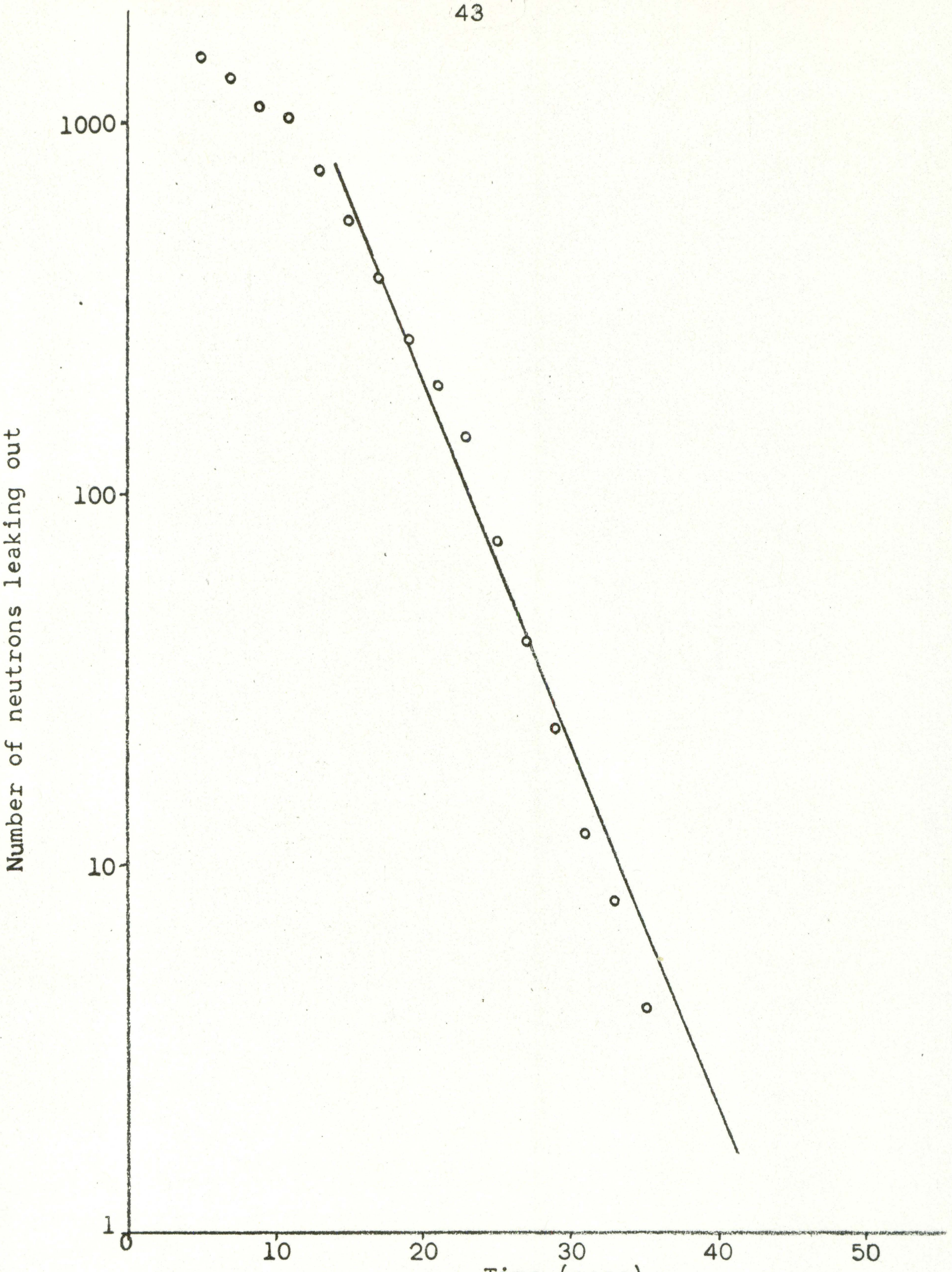


Figure 15. A 15 cm diameter natural uranium sphere

slope of 0.219 $nsec^{-1}$ is shown in Figure 15. This run consisted of 8,000 histories.

A mean die-away time τ can be found by

$$\tau = 1/\lambda$$

Table 1 is a summary of the results obtained from the runs.

Certain trends are shown in Table 1. The mean die-away time increases as the volume of the target assembly increases. This follows naturally where one considers the physical situation. With increased volume, there is an increase in material. This in turn increases the number of reactions that a neutron can have while in the assembly. The increase in the number of reactions increases the mean time it takes to leak out of the target assembly. A plot was made of the mean time versus the volume of the sphere and cubes. The results are shown in Figure 16. From this figure no conclusions as to functional behavior of die-away time with respect to volume can be drawn.

Attempts to correlate the die-away time behavior to surface area were made in Figure 17 and Figure 18 respectively. As in the case of the time-volume correlating, no conclusions can be drawn about functional behavior of the die-away time and the surface area or volume to surface area ratios.

Other trends observed in Table 1 and Figure 16 are that

Table 1. Compiled results

	Dia. of U-238 Spheres			Size of U-238 Cube			Dia. of Nat. U. Sphere
	15 cm	20cm	30cm	15cm	20cm	25cm	15cm
$\lambda(\text{nsec}^{-1})$	0.305	0.179	0.1475	0.1695	0.156	0.101	0.219
$\tau(\text{nsec})$	3.28	5.558	6.78	5.90	6.41	9.90	4.57

for equal volumes of material, the die-away time in the cube are considerably longer than in the spheres. In reactor theory (6) it is learned that a spherical assembly gives the lowest leakage due to the low surface to volume ratio. The apparent contradiction can be explained by considering the manner in which the spherical and cubical assemblies were pulsed. The sphere was pulsed at the center, therefore for neutrons to leak from the assembly the vector sum of their path lengths must be that equivalent to one radius length. However in the cubical assembly the neutrons were uniformly incident on one face. The angle of scattering at 1 MeV energies is an isotropic with a preference toward the forward direction. Therefore, except for those neutrons which are scattered backward and leak out upon arriving, the majority must travel a greater distance, of the order of the cube side, in order to reach a surface. This accounts for the longer die-away time in the cubical cases.

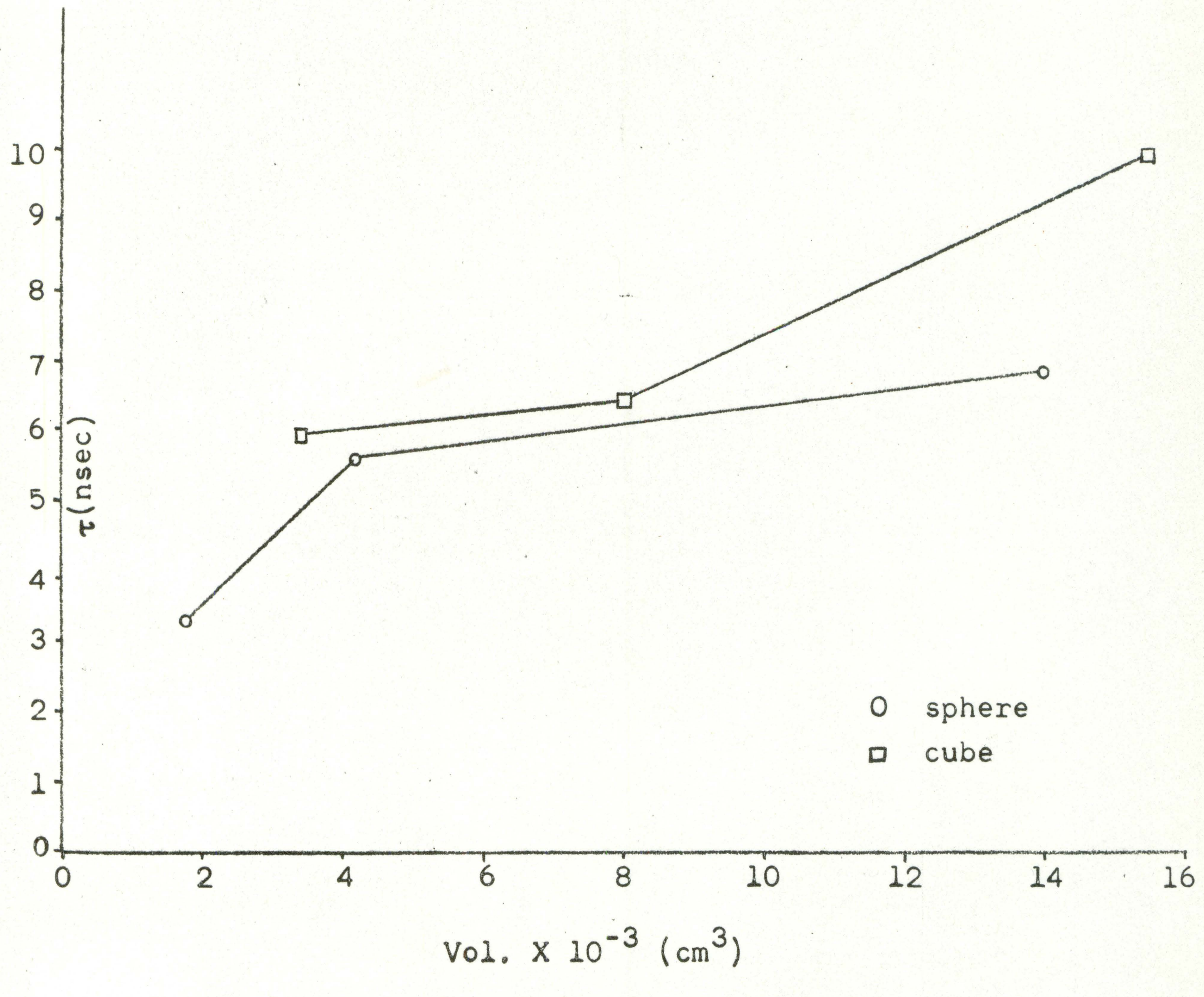
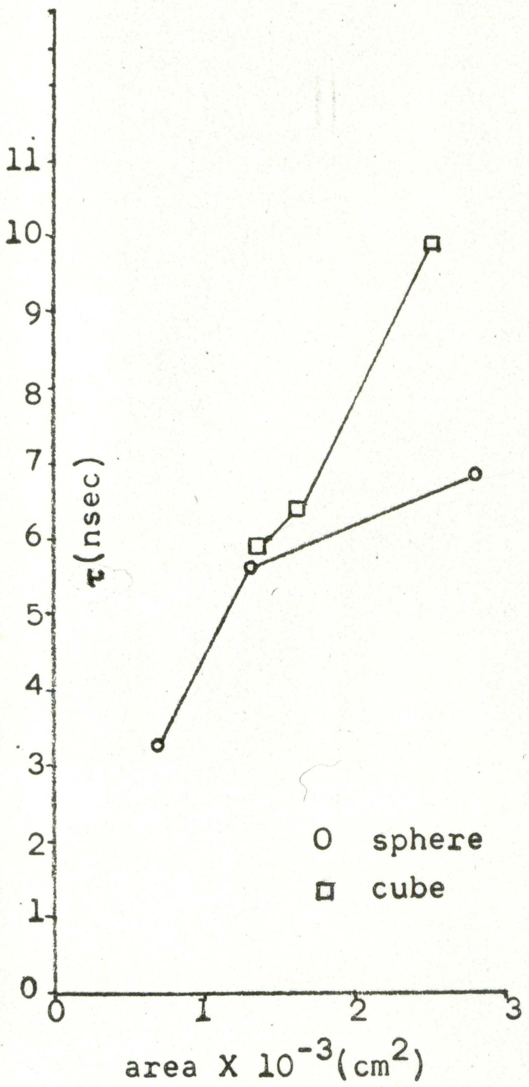


Figure 16. Vol. delay time correlation



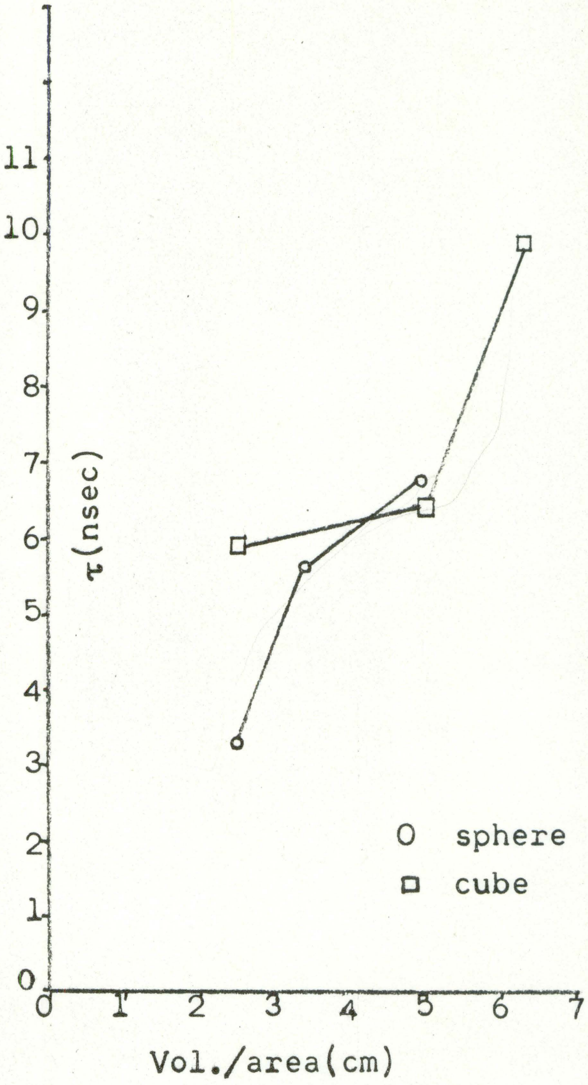


Figure 17. Area decay time correlation

Figure 18. Vol./area ratio decay time correlation

From the log N plots, the data at the larger times appears widely scattered. This should be expected since the number of neutrons is very small making the statistics of the problem poor. On the other hand at short times after pulse injection, the slope of the log N curves is smaller than the slope at later times. These neutrons have had no or at most a very few collisions before leaking out of the assembly. Therefore, it is doubtful they will obey any type of exponential behavior. Only those suffering a number of collisions can be thought of as likely to obey an exponential decay. Besides, since the neutron population in the assembly (and hence also the leakage) must build up from a zero level it is only natural that some sort of peak must be exhibited in the histograms. The question arises, however, why the slower slope continues for quite a time after the peak is reached. A physical argument is offered for this phenomenon.

At times very shortly after $t = 0$ the distribution is flat as in Figure 19(b) whereas at later times it curves as shown in Figure 19(c). Since the leakage rate is proportional to the gradient of the flux just inside the boundary the leakage rate must be larger in the case of Figure 19(c) than in Figure 19(b).

It is also important to note, in Figures 2, 4, and 6 showing the number of neutrons leaking from the spherical assemblies, that no neutrons leak out until 4 sec, 6 sec,

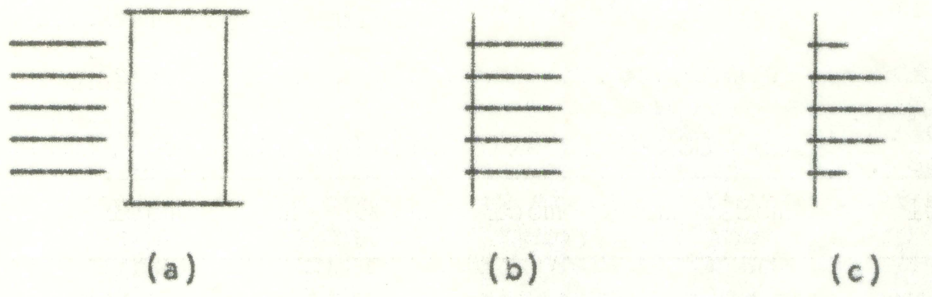


Figure 19. Neutron distribution across cube face at times very close to t = 0 and at later time

and 10 sec respectively. These are the times it takes for a 1 MeV neutron to travel a distance of one radius for each of the spheres. This must be true as the neutrons start at the center of the spheres, and the fact that this feature appeared in the results as expected provides an additional check on the reliability of the code.

At the time of this writing there were no final experimental values for comparison. T. Gozani of General Atomic is at the present working on a 51 cm diameter sphere of U-238, but his final results are not available. In preliminary results (7) he was apparently not finding an exponential die-away time, which is contrary to earlier reports (20).

Work has been done on moderating materials, e.g. beryllium (25) for which the leakage did not obey a single exponential law, but a correlation to a heavy metal is not possible in this case.

V. CONCLUSIONS AND RECOMMENDATIONS

A. Conclusions

From the results expressed in the preceeding section it can be concluded that it is feasible to study neutron leakage and die-away times by the Monte Carlo technique provided the neutron energies are high, and the assembly used as the target is small and consists of a heavy metallic isotope. These stipulations make computer time for this code reasonable. The results indicate the leakage may be grossly expressed by an exponential decay law of the type

$$N = C\ e^{-\lambda t}$$

where N is the number of neutrons leaking out, C is a constant, and λ is an exponential decay constant.

The exponential decay constant λ is a function of both the geometry and size of the assembly. Its functional dependence on size cannot be clearly determined from the data obtained in this work. It is, however, observed that the time decay constant does decrease with increasing volume for both spherical and cubical shapes. It is also smaller for cubical shapes than for spherical shapes of equal volume.

At large times, as the number of neutrons leaking out of the assembly becomes smaller and smaller, a high degree of data scattering and statistical fluctuation is observed just as in the case of low counting rates. Therefore, the Monte Carlo technique and the experimental technique have large statistical deviations at low count rates.

At very small times, the leakage seems to depart from an exponential decay law. The exponential law begins after a certain "stability" has been reached in the leakage process, and this can be considered in terms of simple physical considerations.

B. Recommendations for Future Work

There are numerous possibilities for investigation by use of the Monte Carlo technique.

The dependence of leakage on energy could be found by running a number of cases for the same material and geometrical conditions with variable monoenergetic neutron sources. Also, since neutrons are seldom monoenergetic, the code should be run with an energy spectrum. This would make future comparison with experimental data much more meaningful. An extra facility was added to the "PULSE" code to enable it to handle a spectrum of incident neutron energies, but it was not used in actual runs.

The variation of λ with material is also left to be explored. Various heavy metals, e.g. iron, bismuth, lead, or combinations of metals can be used while keeping energy and geometrical shape constant.

In addition to the leakage output, the "PULSE" code provides capture, fission, and scattering information which may be of interest.

Other codes have been developed, e.g. O5R (13). This code has been developed and used at Oak Ridge. It is a very general neutron transport code. It may be run as a check on the "PULSE" code, or part of O5R may be used in combination with "PULSE" to write an improved code which will handle more complicated geometrical shapes, e.g. reactor cores, or shields on space vehicles.

In addition to the running of a Monte Carlo code, another possibility for future work consists of doing further theoretical work in the behavior of a neutron pulse in a small assembly based on transport equation solutions.

Finally, an experiment can be developed using the I.S.U. neutron generator. This would presumably be similar to the type being performed by T. Gozani (7) at General Atomic which was mentioned earlier. The experimental results could be correlated with the results predicted by Monte Carlo.

In the use of this code and obtaining data for its use other theoretical and experimental problems arose which should be investigated. These include such topics as the inelastic scattering in the continuum region, inelastic scattering from levels, and neutron cross section data evaluation.

VI. BIBLIOGRAPHY

1. Batchelor, R., Gilboy, W. B., and Towle, J. H. Neutron interactions with U-238 and Th-232 in the energy region 1.6 MeV to 7 MeV. Nuclear Physics 65: 236-256. 1965.

2. Cashwell, E. D., Everet, C. J., and Rechard, O. W. Practical manual on the Monte Carlo method for random walk problems. U.S. Atomic Energy Commission Report LA-2120 [Los Alamos Scientific Lab., N. Mex.]. 1957.

3. Clark, M., Jr. and Hansen, K. E. Methods of numerical analysis. New York, New York, Academic Press. 1964.

4. Cupini, E., Molinari, V. G., and Solinas, G. Time-dependent neutron thermalization by Monte Carlo method. Nukleonik 9: 295-300. 1967.

5. Fleck, C. M., Hejtmanek, H., and Kopitsch, F. Ein Monte Carlo Programm für den zeitabhängigen Neutronentransport eines thermischen Pulses in Leichtwasser. Nukleonik 9: 157-160. 1967.

6. Glasstone, S. and Edlund, M. C. The elements of nuclear reactor theory. Princeton, New Jersey, D. Van Nostrand Company, Inc. 1952.

7. Gozani, T., Moore, R. A., Neill, J. M., and Main, G. Experimental kinetic studies on depleted uranium sphere. Transactions of the American Nuclear Society 10: 280-281. 1967.

8. Hanna, G. C., and Clarke, R. L. Neutron evaporation in the 14 MeV neutron fission of uranium. Canadian Journal of Physics 39: 967-973. 1961.

9. Henkel, R. L. Fission by fast neutrons. In Marian, J. B. and Fowler, J. L., eds. Fast neutron physics. Vol. 2. pp. 2001-2050. New York, New York, Interscience Publishers. 1963.

10. Hughes, D. J. and Carter, R. S. Neutron cross sections, angular distributions. 2nd ed. U.S. Atomic Energy Commission Report BNL-400 (Brookhaven National Lab., Upton, N.Y.). 1956.

11. Hughes, D. J. and Schwartz, R. B. Neutron cross sections. U.S. Atomic Energy Commission Report BNL-325 (Brookhaven National Lab., Upton, New York) Supplement Number 2. 1958.

12. Iowa State University. Computer Center. Catalogued programs. Ames, Iowa, Computer Center, Iowa State University. 1967.

13. Irving, D. C., Freestone, R. M., Jr., and Kam, F. B. K. O5R: a general Monte Carlo neutron transport code. U.S. Atomic Energy Commission Report ORNL-3622 (Oak Ridge National Lab., Tenn.). 1965.

14. Kahn, H. Applications of Monte Carlo. U.S. Atomic Energy Commission Report AECU-3259 (Division of Technical Information Extension, AEC). 1956.

15. Mayne, A. J. Monte Carlo methods for solving neutron problems. U.S. Atomic Energy Commission Report AWRE-18/55 (Great Britain Atomic Weapons Research Establishment, Aldermaston, Berks, England). 1955.

16. Meyer, H. A., ed. Symposium on Monte Carlo methods: held at the University of Florida, conducted by Wright Air Development Center of the Air Research and Development Command, 1954. New York, New York, John Wiley and Sons, Inc. c1956.

17. Monte Carlo method: proceedings of a symposium held at Los Angeles, conducted by RAND Corp. and the National Bureau of Standards, with the cooperation of the Oak Ridge National Lab. U.S. Department of Commerce National Bureau of Standards Applied Mathematics Series No. 12. 1951.

18. Profio, A. E. PULSE, an IBM 7094 program for calculation of fast neutron kinetics by Monte Carlo; Addendum No. 1, May 1964. Cambridge, Massachusetts, Department of Nuclear Engineering, Massachusetts Institute of Technology. 1964.

19. Profio, A. E. PULSE, an IBM 7094 program for calculation of fast neutron kinetics by Monte Carlo: progress report Oct. 1963. Cambridge, Massachusetts, Department of Nuclear Engineering, Massachusetts Institute of Technology. 1963.

20. Profio, A. E., Koppel, J. U., and Adamantiades, A. Measurements and calculations of the slowing down and migration time. U.S. Atomic Energy Commission Report GA-6290 (General Atomic Div., General Dynamics Corp., San Diego, Calif.). 1965.

21. Pulsed neutron research: symposium held in Karlsruhe, Germany, conducted by the International Atomic Energy Agency, 1965. Vienna, Austria, International Atomic Energy Agency. 1965.

22. Reactor physics constants. U.S. Atomic Energy Commission Report ANL-5800 (Argonne National Lab., Lemont, Ill.). 1958.

23. Smith, A. B. Inelastic neutron scattering: a compendium; Conference on Neutron Cross Section Technology, 1966. U.S. Atomic Energy Commission Report CONF-660303 (Division of Technical Information Extension, AEC): 577-598. 1966.

24. Smith, A. B. Scattering of fast neutrons from natural uranium. Nuclear Physics 47: 633-651. 1963.

25. Wolberg, J. R. and Gozani, T. Measurement of leakage kernals and slowing down parameters using the pulsed source technique. Nukleonik 9: 180-187. 1967.

26. Wood, J. The decay of a neutron pulse in low temperature beryllium. Nukleonik 10: 3-6. 1967.

27. Zamiatnin, I. S., Safino, I. N., Gutnikora, E. K., and Ivanova, N. I. Spectra of neutrons produced by 14 MeV neutrons in fissile materials. Soviet Journal of Atomic Energy 4: 443-449. 1958.

VII. ACKNOWLEDGMENTS

The author wishes to thank the following for their assistance toward completion of this thesis. Dr. A. Adamantiades of the Iowa State Nuclear Engineering Department for his advice and consultation; Mr. H. Jesperson of the Iowa State Computation Center for his assistance in answering programming questions; and Dr. Glenn Murphy head of the Iowa State University Nuclear Engineering Department for his assistance and advice concerning this work. The author also wishes to thank the National Aeronautics and Space Administration for the fellowship which provided funds for his education.

In addition to the above mentioned people, the author wishes to acknowledge his parents, Mr. and Mrs. Fergus Flanagan, for their encouragement and assistance throughout his entire college career.

VIII. APPENDIX A

Random Number Generation

The pseudorandom numbers used in the running of this code were generated by a subroutine called RANDU. This routine was developed by IBM and was supplied by the Iowa State University Computation Center. The routine is called by the FORTRAN statement CALL RANDU(IX, IY, YFL). For the first calling, IX is supplied as an input variable. It is an interger of nine digits or less. IY is generated by the routine and is substituted for IX when ever the routine is used again. YFL is the pseudorandom number of nine digits uniformly distributed between 0 and 1.0

Following is a listing of the FORTRAN statements making up the RANDU code:

```
    SUBROUTINE RANDU(IX,IY,YFL)
    IY = IX* 65539
    IF (IY) 5,5,6
5   IY = IY + 2147483647 + 1
6   YFL = IY
    YFL = YFL*(0.4656613E-9)
    RETURN
```

The number of pseudorandom, uniformly distributed numbers which can be generated before a repetition is encountered is stated by the I.S.U. Computation Center as two raised to the twenty-ninth power or approximately five hundred million numbers.

IX. APPENDIX B

Directional Cosine Computation

The subroutine ISOANG is used to compute the direction cosines (α, β, γ) for isotropic elastic scattering. This routine is supplied with variable GAMMAC (the polar directional cosine) which is computed or chosen from a probability distribution. The Z-coordinate directional cosine γ is set equal to GAMMAC. Alpha and beta are chosen so that

$$\alpha^2 + \beta^2 + \gamma^2 = 1$$

The subroutine involves the solving of the following equations:

$$\alpha = \varepsilon_1 \frac{\sqrt{1 - \gamma^2}}{\eta}$$

$$\beta = \varepsilon_2 \frac{\sqrt{1 - \gamma^2}}{\eta}$$

ε_1, ε_2, and η are obtained as shown below where R_1 and R_2 are pseudorandom numbers generated by RANDU.

$$\varepsilon_1 = 2R_1 - 1$$

$$\varepsilon_2 = 2R_2 - 1$$

$$\eta = \varepsilon_1^2 + \varepsilon_2^2$$

Also, since isotropic center of mass scattering is presumed, the new velocity (VEL) is set equal to the incident velocity in this subroutine.

X. APPENDIX C

Probabilities of Occurrence of Interactions

The probabilities of occurrence of the various nuclear interactions are computed in the following manner. The macroscopic cross sections for each of the interactions (elastic scattering, inelastic scattering, fissions and capture) are computed using the atomic density and microscopic cross sections which have been supplied as inputs for each of the twenty velocity groups mentioned earlier. The following formula is employed in the calculation of these cross sections.

$$\Sigma_{i,j} = \sigma_{i,j} N_j$$

$\Sigma_{i,j}$ is the macroscopic cross section for the i-th interaction with nuclide j. $\sigma_{i,j}$ is the microscopic cross section for the i-th interaction with the nuclide j. N_j is the atomic density of the nuclide j.

These macroscopic cross sections are summed for all possible reactions with all the nuclides to get a total macroscopic cross section Σ_T.

$$\Sigma_T = \sum_{j=1}^{k} \sum_{i=1}^{n} \sigma_{i,j} N_j$$

where n is the number of possible interactions and k is the number of nuclides.

The probability for the i-th reaction with the j-th nuclide is then

$$P_{i,j} = \frac{\Sigma_{i,j}}{\Sigma_T}$$

The above equations are used in the subroutine SIGMA of the "PULSE" code.

In order to find which reaction has taken place, the subroutine COLIDX is called. COLIDX uses a random number R generated by RANDU and first compares it to $P_{1,1}$, if R is less than $P_{1,1}$ the first interaction is assumed to have taken place with the first nuclide. If R is greater than $P_{1,1}$, then a comparison is made to the sum $P_{1,1} + P_{1,2}$. Again if R is less than the sum then interaction 1 is assumed to take place with nuclide 2. If R is greater than the above sum it is compared to the sum $P_{1,1} + P_{1,2} + P_{2,1}$. This procedure continues, adding the probabilities $P_{i,j}$ one at a time checking after each addition to see if the sum is greater than R. If the sum of probabilities is found to be greater than R after the addition of $P_{i,j}$, the i-th reaction is taken to have occurred with the j-th nuclide.

From nuclear reactor theory (6) it is found that the probability P of a neutron traveling a distance x without being involved in a reaction is given by

$$P = e^{-x/\lambda_T}$$

where λ_T is the total mean free path. It is equal Σ_T^{-1} where Σ_T is the total macroscopic cross section calculated above. This relationship is used to find the distance traveled between interactions. Using a pseudorandom number R generated by RANDU, the subroutine FLITE computes the distance X between reactions by solving the following equation

$$X = -\lambda_T \ln R$$

XI. APPENDIX D

Conversion from Center of Mass to Laboratory System

The conversion from the center of mass to the laboratory coordinate system is accomplished by means of an intermediate coordinate system (21) whose coordinates have the subscript P in the following derivation. The center of mass coordinates have a subscript C, and the lab system has no subscript.

First, two pseudorandom numbers R_1 and R_2 (in the range of 0.0 to 1.0) are generated by RANDU. These are converted to pseudorandom numbers ε_1 and ε_2 respectively by the following equations.

$$\varepsilon_1 = 2R_1 - 1$$

$$\varepsilon_2 = 2R_2 - 1$$

ε_1 and ε_2 now have a range between -1 and +1.

The routine CMLAB is used to perform the calculations necessary for the conversion. The input to this routine includes the direction cosines $(\alpha_i, \beta_i, \gamma_i)$ all of which are in the lab system prior to the collision, and a variable (γ_c) which is obtained using the expression below.

$$\gamma_c = 2R_0 - 1$$

R_0 is again a pseudorandom number obtained from RANDU.

The center of mass direction cosines $(\alpha_c, \beta_c, \gamma_c)$ are found using the following:

$$\alpha_c = \varepsilon_1 \frac{\sqrt{1 - \gamma_c^2}}{\eta}$$

$$\beta_c = \varepsilon_2 \frac{\sqrt{1 - \gamma_c^2}}{\eta}$$

$$\gamma_c = \gamma_c$$

$$\eta = \varepsilon_1^2 + \varepsilon_2^2$$

A conversion is now made from the center of mass system to an intermediate system to obtain the directional cosines $(\alpha_p, \beta_p, \gamma_p)$.

$$\alpha_p = \frac{\alpha_i \gamma_i \alpha_c - \beta_i \beta_c}{\sqrt{1 - \gamma_i^2}} + \alpha_i \gamma_c$$

$$\beta_p = \frac{\beta_i \gamma_i \alpha_c - \alpha_i \beta_c}{\sqrt{1 - \gamma_i^2}} + \beta_i \gamma_c$$

$$\gamma_p = -\alpha_c \sqrt{1 - \gamma_i^2} + \gamma_i \gamma_c$$

The cosines (α, β, γ) in the lab system can now be calculated.

$$\alpha = \frac{\alpha_i + A\,\alpha_p}{\sqrt{1 + A^2 + 2A\gamma_c}}$$

$$\beta = \frac{\beta_i + A\,\beta_p}{\sqrt{1 + A^2 + 2A\gamma_c}}$$

$$\gamma = \frac{\gamma_i + A\,\gamma_p}{\sqrt{1 + A^2 + 2A\gamma_c}}$$

where A is the mass number of the target nuclide.

In addition to the directional cosines, the lab system velocity after the collision is also calculated by CMLAB.

$$V = \frac{V_i\sqrt{1 + A^2 + 2A\gamma_c}}{A + 1}$$

where V_i is the velocity of the neutron prior to the collision.

XII. APPENDIX E

Method Used to Select Values From a Given Density Distribution

The method employed in obtaining values for anisotropic scattering directional cosines and fission velocities is based on the probability distribution theory and the theory of cumulative probabilities.

If a continuous distribution of values is given as in Figure 20, such that the shaded area A can be thought of as representing the probability that a random variable X is less than or equal to x_i, then the probability that X is less than xmax is the entire area under the curve or a probability of 1.0. The probability that x is less than or equal to xmin is 0.

If the distribution in Figure 20 is integrated and normalized (Figure 21), then for any value x_i chosen on the abcissa, the probability that X is less than or equal to x_i is the value of the ordinate y_i corresponding to the point (x_i, y_i) on the integrated curve.

The above concept is used to find the directional cosine of the angle of exit for elastic scattering given an anisotropic angular distribution. It is also employed in determining the velocity of a neutron resulting from a fission reaction given the fission spectrum.

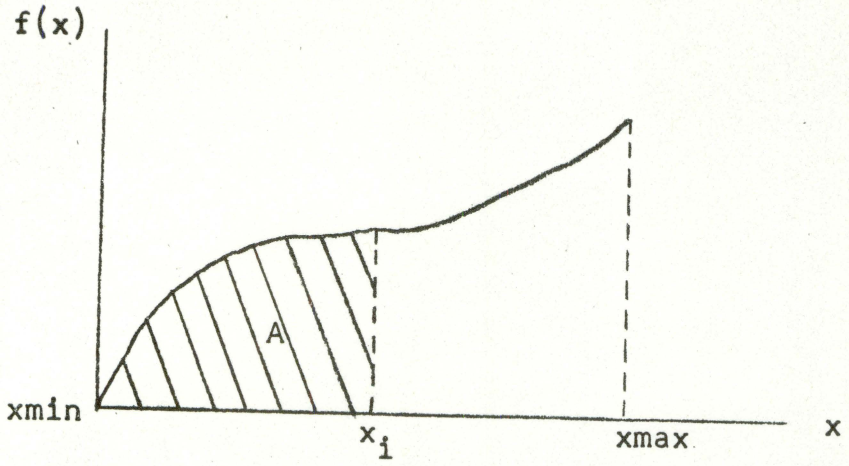


Figure 20. The distribution function as an area

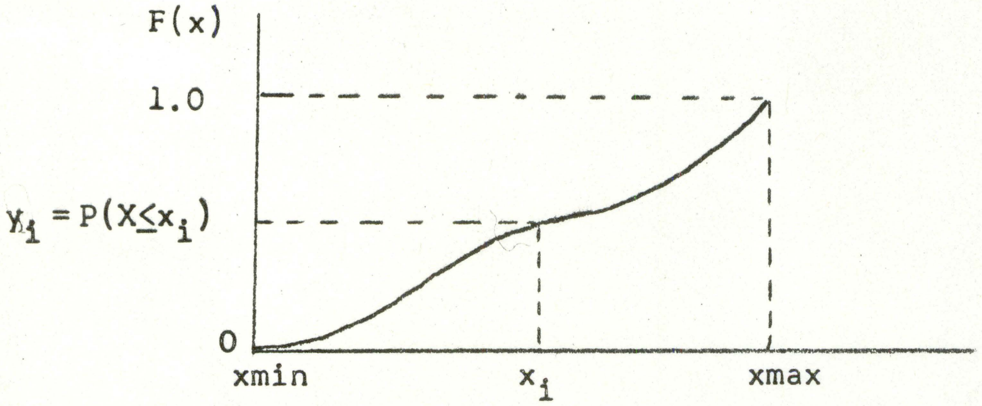


Figure 21. The cumulative distribution function

The angular distributions for elastic scattering at various energies (10) were graphically integrated and the ordinate was divided into 10 equal intervals ranging from 1 to 11. The lower limit of 1 is necessary due to the way arrays are indexed in FORTRAN. The value of the cosine θ for the point 11 is taken as +1. The values for the cosine θ for the other integer points, 1-10, are read from the integrated curve and stored in an array.

A pseudorandom number between 0 and 1.0 is generated in the code "PULSE" by the routine RANDU. This number is multiplied by 10 and added to 1.0 to give an integer between 1 and 11 and a remainder. The cosine is then obtained using the integer points from the array and the integers generated by RANDU. The remainder obtained from the random number is used to linearly interplate between the integer points in the array.

The fission neutron velocities are obtained in a similar manner except that the ordinate of the integrated distribution (9) is divided into 22 intervals 1-22. The pseudorandom number is multiplied by 20 and then 1 is added; the rest of the calculation proceeds as described above. The value of the integer point 22 is taken as the most energetic neutron of the spectrum.

XIII. APPENDIX F

A Listing of the "PULSE" Code

```
C       PULSE  MONTE CARLO  CODE
C       PROGRAMED BY A.E. PROFIO AT MIT IN 1963
C       REVISED AND UPDATED BY G.F. FLANAGAN  AT ISU IN 1967
C       USED FOR THE CALCULATION OF SLOWING DOWN PARAMETERS
C       IN FAST METAL ASSEMBLIES
        DIMENSION SP(10),SBE1(20),SBI1(20),SBF1(20),SBC1(20),
       1SBE2(20),SBI2(20),SBF2(20),SBC2(20),VBOUND(20),AP1(10
       2,20),AP2(10,20),SBL1(20,20),SBL2(20,20),P(22),VL1(20)
       3,VL2(20),SL(20),PL(20),FP1(22),FP2(22),LEAK(100,10),
       4NELS(100,10),NINS(100),NFIS(100),KAPT(100)
        PAUSE 1
        REWIND 10
        READ (1,1)  XS,YS,ZS,PARA,PARB,PARC,THETA,KS,NEUT
1       FORMAT(7F8.4,I2,I14)
        WRITE (3,2) XS,YS,ZS,PARA,PARB,PARC,THETA,KS,NEUT
2       FORMAT(1H1,3HXS=F8.4,2X,3HYS=F8.4,2X,3HZS=F8.4,2X,5HP
       1ARA=F8.4,2X,5HPARB=F8.4,2X,5HPARC=F8.4,2X,6HTHETA=F8.4,
       22X,3HKS=I2,2X,5HNEUT=I14)
        READ (1,3)SP
3       FORMAT(10F7.4)
        WRITE (3,4)SP
4       FORMAT(1H0,3HSP=10F7.4)
        READ  (1,5)XMAX,YMAX,ZMAX,RMAX,KAS
5       FORMAT(4F8.4,I2)
        WRITE (3,6)XMAX,YMAX,ZMAX,RMAX,KAS
6       FORMAT(1H0,5HXMAX=F8.4,2X,5HYMAX=F8.4,2X,5HZMAX=F8.4,2X
       1,5HRMAX=F8.4,2X,4HKAS=I2)
        READ(1,7)TD,TCH,EMIN,ECH,KT1,KT2
7       FORMAT(4F7.3,2I3)
        WRITE (3,8)TD,TCH,EMIN,ECH,KT1,KT2
8       FORMAT(1H0,3HTD=F7.3,2X,4HTCH=F7.3,2X,5HEMIN=F7.3,2X,4H
       1ECH=F7.3,2X,4HKT1=I3,2X,4HKT2=I3)
        READ(1,9) P
9       FORMAT(11F6.2)
        WRITE(3,10)P
10      FORMAT(1H0,2HP=11F6.2/3X,11F6.2)
        READ(1,11)VBOUND
11      FORMAT(10F7.4)
        WRITE(3,12)VBOUND
12      FORMAT(1H0,7HVBOUND=10F7.4/8X,10F7.4)
        READ(1,13) AD1,A1,ALIM1,SLIM1,CIN1,VST1,FNU1,DELNU1,KIA1
13      FORMAT(F7.5,2F7.2,5F8.4,I2)
        WRITE(3,14)AD1,A1,ALIM1,SLIM1,CIN1,VST1,FNU1,DELNU1,KIA1
14      FORMAT(1H0,4HAD1=F7.5,2X,3HA1=F7.2,2X,6HALIM1=F7.2,2X,6
       1HSLIM1=F8.4,2X,5HCIN1=F8.4,2X,5HVST1=F8.4,2X,5HFNU1=F8.
       24,2X,7HDELNU1=F8.4,2X,5HKIA1=I2)
        READ(1,15)SBE1
15      FORMAT(10F7.3)
        DO 16 J=1,20
16      SBE1(J)=AD1*SBE1(J)
```

```
      WRITE(3,17)SBE1
17    FORMAT(1H0,4HSBE=10F7.3/5X,10F7.3)
      READ(1,15)SBI1
      DO 18 J=1,20
18    SBI1(J)=AD1*SBI1(J)
      WRITE(3,19)SBI1
19    FORMAT(1H0,4HSBI=10F7.3/5X,10F7.3)
      READ(1,15)SBF1
      DO 20 J=1,20
20    SBF1(J)=AD1*SBF1(J)
      WRITE(3,21)SBF1
21    FORMAT(1H0,4HSBF=10F7.3/5X,10F7.3)
      READ (1,15)SBC1
      DO 22 J=1,20
22    SBC1(J)=AD1*SBC1(J)
      WRITE(3,23)SBC1
23    FORMAT(1H0,4HSBC=10F7.3/5X,10F7.3)
      READ (1,15)AP1
      WRITE(3,24)AP1
24    FORMAT(1H0,3HAP=10F7.3/4X,10F7.3/4X,10F7.3/4X,10F7.3/4X
     1,10F7.3/4X,10F7.3/4X,10F7.3/4X,10F7.3/4X,10F7.3/4X,10F7.
     23/4X,10F7.3/4X,10F7.3/4X,10F7.3/4X,107.3/4X,10F7.3/4X,1
     30F7.3/4X,10F7.3/4X,10F7.3/4X,10F7.3/4X,10F7.3)
      READ (1,15)VL1
      WRITE(3,25)VL1
25    FORMAT(1H0,3HVL=10F7.3/4X,10F7.3)
      READ (1,15)SBL1
      WRITE(3,26)SBL1
26    FORMAT(1H0,4HSBL=10F7.3/(5X,10F7.3))
      READ (1,27)FP1
27    FORMAT(11F6.3)
      WRITE(3,28)FP1
28    FORMAT(1H0,3HFP=11F6.3/4X,11F6.3)
      READ (1,13)AD2,A2,ALIM2,SLIM2,CIN2,VST2,FNU2,DELNU2,KIA2
      WRITE(3,29)AD2,A2,ALIM2,SLIM2,CIN2,VST2,FNU2,DELNU2,KIA2
29    FORMAT(1H0,4HAD2=F7.5,2X,3HA2=F7.2,2X,6HALIM2=F7.2,2X,6
     1HSLIM2=F8.4,2X,5HCIN2=F8.4,2X,5HVST2=F8.4,2X,5HFNU2=F8.
     24,2X7HDELNU2=F8.4,2X,5HKIA2=I2)
      IF(AD2)40,40,30
30    READ (1,15)SBE2
      DO 31 J=1,20
31    SBE2(J)=AD2*SBE2(J)
      WRITE(3,17)SBE2
      READ (1,15)SBI2
      DO 32 J=1,20
32    SBI2(J)=AD2*SBI2(J)
      WRITE(3,19)SBI2
      READ (1,15)SBF2
      DO 33 J=1,20
33    SBF2(J)=AD2*SBF2(J)
```

```
      WRITE (3,21)SBF2
      READ (1,15)SBC2
      DO 34 J=1,20
34    SBC2(J)=AD2*SBC2(J)
      WRITE (3,23)SBC2
      READ (1,15)AP2
      WRITE (3,24)AP2
      READ (1,15)VL2
      WRITE (3,25)VL2
      READ (1,15)SBL2
      WRITE (3,26)SBL2
      READ (1,27)FP2
      WRITE (3,28)FP2
      GO TO 50
40    DO 41 J=1,20
41    SBE2(J)=0.0
42    DO 43 J=1,20
43    SBI2(J)=0.0
44    DO 45 J=1,20
45    SBF2(J)=0.0
46    DO 47 J=1,20
47    SBC2(J)=0.0
50    REWIND KT1
      REWIND KT2
      KT=KT1
      MULT=1
      NL=0
      NC=0
      NS=0
      NT=0
      NF=0
      NLTD=0
      NGTR=0
      NGZR=0
      NLME=0
      NGER=0
      NOSL=0
      KSCAT=0
      IPU=0
      ITOT=0
      READ (1,90)IX
90    FORMAT(I9)
91    READ(1,95) JJ
95    FORMAT(I5)
100   DO 801 N=1,NEUT
107   IPU=IPU+1
      ITOT=ITOT+1
      IF(IPU-500)110,110,862
862   WRITE(10)ITOT,IX,NL,NC,NS,NF,NLTD,NGTR,NGZR,NLME,NGER,
     1NOSL,LEAK,NELS,NINS,NFIS,KAPT
```

```
      IPU=1
      WRITE(3,863)ITOT
  863 FORMAT(1H ,I8)
      REWIND 10
110   CALL SOURCE(ALPHA,BETA,GAMMA,VEL,X,Y,Z,TIME,PARA,PARB,
     1PARC,XS,YS,ZS,ZMAX,THETA,SP,KS,IX)
120   CALL SIGMA(VEL,SBE1,SBE2,SBI1,SBI2,SBF1,SBF2,SBC1,SBC2,
     1AD1,AD2,VBOUND,TMFP,PE1,PE2,PI1,PI2,PF1,PF2,PC1,J)
      IF(J)122,122,127
122   NT=NT+1
      IF(NT-5)123,123,124
123   GO TO (110,809,820),MULT
124   WRITE(3,125)NT
125   FORMAT(1H0,3HNT=I2)
      GO TO 900
127   NT=0
130   CALL FLITE(DIST,TIMET,TMFP,VEL,IX)
140   GO TO (145,150,155),KAS
145   CALL DTPB(ALPHA,BETA,GAMMA,X,Y,Z,XMAX,YMAX,ZMAX,DISTB)
146   CALL POST(ALPHA,BETA,GAMMA,X,Y,Z,DIST,DISTB,TIME,TIMET,
     1VEL,KGEO)
147   GO TO(160,600),KGEO
150   CALL DTCB(ALPHA,BETA,GAMMA,X,Y,Z,RMAX,ZMAX,DISTB)
151   GO TO 146
155   CALL DTSB(ALPHA,BETA,GAMMA,X,Y,Z,RMAX,DISTB)
156   GO TO 146
160   IF(AD2)161,161,165
161   CALL COLID1(PE1,PI1,PF1,KCOL,IX)
      GO TO 170
165   CALL COLID2(PE1,PE2,PI1,PI2,PF1,PF2,PC1,KCOL,IX)
170   KTYPE=KCOL/10
      KNUCL=KCOL-(10*KTYPE)
      GO TO (200,300,400,500),KTYPE
200   CALL ELTAL(TIME,TD,TCH,Z,ZMAX,KELS,NELS)
      NS=NS+1
      GO TO (203,205,207,207,209),KELS
203   NLTD=NLTD+1
      GO TO 209
205   NGTR=NGTR+1
      GO TO 800
207   NGZR=NGZR+1
      GO TO 800
209   KSCAT=KSCAT+1
      IF(KSCAT-100)211,211,225
211   GO TO (215,220),KNUCL
215   CALL ELSCAT(ALPHA,BETA,GAMMA,VEL,A1,ALIM1,SLIM1,AP1,J,
     1IX)
      GO TO 120
220   CALL ELSCAT(ALPHA,BETA,GAMMA,VEL,A2,ALIM2,SLIM2,AP2,J,
     1IX)
```

```
      GO TO 120
225   NOSL=NOSL+1
      KSCAT=0
      GO TO 800
300   CALL INTAL(TIME,TD,TCH,KINS,NINS)
      NS=NS+1
      GO TO (303,305,305,305,307),KINS
303   NLTD=NLTD+1
      GO TO 307
305   NGTR=NGTR+1
      GO TO 800
307   KSCAT=KSCAT+1
      IF(KSCAT-100)309,309,320
309   GO TO (310,315),KNUCL
310   IF(VEL-VST1)311,312,312
311   CALL LEVEL(VEL,SBL1,VBOUND,PL,J)
312   CALL INSCAT(ALPHA,BETA,GAMMA,VEL,A1,CIN1,P,PL,VL1,VST1
     1,KIA1,IX)
      GO TO 120
315   IF(VEL-VST2)316,317,317
316   CALL LEVEL(VEL,SBL2,VBOUND,PL,J)
317   CALL INSCAT(ALPHA,BETA,GAMMA,VEL,A2,CIN2,P,PL,VL2,VST2,
     1KIA2,IX)
      GO TO 120
320   NOSL=NOSL+1
      KSCAT=0
      GO TO 800
400   CALL FISTAL(TIME,TD,TCH,KFIS,NFIS)
      KSCAT=0
      GO TO (402,404,404,404,406),KFIS
402   NLTD=NLTD+1
      GO TO 406
404   NGTR=NGTR+1
      GO TO 800
406   GO TO (407,409),KNUCL
407   CALL FISSN(X,Y,Z,VEL,TIME,FP1,FNU1,DELNU1,NF,KT,IX)
      GO TO 800
409   CALL FISSN(X,Y,Z,VEL,TIME,FP2,FNU2,DELNU2,NF,KT,IX)
410   GO TO 800
500   CALL CAPTAL(TIME,TD,TCH,KCAP,KAPT)
      NC=NC+1
      KSCAT=0
      GO TO (504,506,506,506,507),KCAP
504   NLTD=NLTD+1
      GO TO 800
506   NGTR=NGTR+1
507   GO TO 800
600   CALL LEKTAL(TIME,VEL,TD,TCH,EMIN,ECH,KLEK,LEAK)
      NL=NL+1
      KSCAT=0
      GO TO (604,606,608,610,611),KLEK
604   NLTD=NLTD+1
      GO TO 800
606   NGTR=NGTR+1
      GO TO 800
608   NLME=NLME+1
      GO TO 800
610   NGER=NGER+1
611   GO TO 800
800   GO TO (801,809,820),MULT
801   CONTINUE
      KS=1
      GO TO 850
803   MULT=2
      REWIND KT1
      REWIND KT2
      IF(NF)850,850,807
807   N=NF
      WRITE(3,808)NF
808   FORMAT(1H0,3HNF=I8)
      NF=0
809   N=N-1
      IF(N)814,811,811
811   READ(KT1)XS,YS,ZS,PARA,THETA
      KT=KT2
      GO TO 107
814   MULT=3
      REWIND KT1
      REWIND KT2
      IF(NF)850,850,818
818   N=NF
      WRITE(3,808)NF
      NF=0
820   N=N-1
      IF(N)803,822,822
822   READ(KT2)XS,YS,ZS,PARA,THETA
      KT=KT1
      GO TO 107
849   READ(1,1) XS,YS,ZS,PARA,PARB,PARC,THETA,KS,NEUT
      WRITE(3,2) XS,YS,ZS,PARA,PARB,PARC,THETA,KS,NEUT
      GO TO 100
850   WRITE(3,851)NL,NC,NS,NF,NLTD,NGTR,NGZR,NLME,NGER,NOSL
851   FORMAT(1H1,3HNL=I8,2X,3HNC=I8,2X,3HNS=I8,2X,3HNF=I8/1H0,
     15HNLTD=I8,2X,5HNGTR=I8,2X,5HNGZR=I8,2X,5HNLME=I8,2X,5HNG
     2ER=I8,2X,5HNOSL=I8)
      WRITE(3,853)LEAK
853   FORMAT(1H0,5HLEAK=20I6/(6X,20I6))
      WRITE(3,855)NELS
855   FORMAT(1H4,5HNELS=20I6/(6X,20I6))
      WRITE(3,857)NINS
```

```
857    FORMAT(1H4,5HNINS=20I6/(6X,20I6))
       WRITE (3,859)NFIS
859    FORMAT(1H4,5HNFIS=20I6/(6X,20I6))
       WRITE (3,861)KAPT
861    FORMAT(1H4,5HKAPT=20I6/(6X,20I6))
865    WRITE(3,866)IX
866    FORMAT(1H0,I10)
       IF(NF) 871,871,870
870    IF(ITOT-NEUT) 803,803,871
871    JJ=JJ-1
       IF(JJ) 867,867,849
867    WRITE(3,868)
868    FORMAT(1H0,17HPROGRAM COMPLETED)
900    END
```

```
      SUBROUTINE SOURCE(ALPHA,BETA,GAMMA,VEL,X,Y,Z,TIME,PARA,
     1PARB,PARC,XS,YS,ZS,ZMAX,THETA,SP,KS,IX)
      DIMENSION SP(10)
      GO TO(10,20,30,40),KS
10    X=XS
      Y=YS
      Z=ZS
      CALL RANDU(IX,IY,YFL)
      IX=IY
      GAMMAC=2.0*YFL-1.0
      VEL=PARA
      CALL ISOANG(ALPHA,BETA,GAMMA,GAMMAC,VEL,IX)
      TIME=THETA
      RETURN
20    CALL RANDU(IX,IY,YFL)
      IX=IY
      X=XS*(2.0*YFL-1.0)
      CALL RANDU(IX,IY,YFL)
      IX=IY
      Y=YS*(2.0*YFL-1.0)
      Z=ZS
      GAMMA=1.0
      ALPHA=0.0
      BETA=0.0
      CALL RANDU(IX,IY,YFL)
      IX=IY
      VEL=PARA-PARB*YFL
      TIME=0.0
      RETURN
30    CALL ANGLS(SP,GAMMAC,IX)
      CALL RANDU(IX,IY,YFL)
      IX=IY
      VEL=PARA-PARB*YFL-PARC*(1.0-GAMMAC)
      CALL ISOANG(ALPHA,BETA,GAMMA,GAMMAC,VEL,IX)
      S=(-ZMAX-ZS)/GAMMA
      X=S*ALPHA
      Y=S*BETA
      Z=-ZMAX
      TIME=S/VEL
      RETURN
40    CALL TARGET(ALPHA,BETA,GAMMA,VEL,X,Y,Z,TIME,PARA,PARB,
     1PARC,IX)
      RETURN
      END
      SUBROUTINE ANGLS(SP,GAMMAC,IX)
      DIMENSION SP(10)
      CALL RANDU(IX,IY,YFL)
      IX=IY
      M=10.0*YFL+1.0
      REM=YFL-0.1*FLOAT(M-1)
      IF(10-M)30,10,20
10    GAMMAC=SP(10)+(REM/0.1)*(1.0-SP(10))
      RETURN
20    GAMMAC=SP(M)+(REM/0.1)*(SP(M+1)-SP(M))
      RETURN
30    GAMMAC=1.0
      RETURN
      END
      SUBROUTINE TARGET(ALPHA,BETA,GAMMA,VEL,X,Y,Z,TIME,PARA,
     1PARB,PARC,IX)
      X=X
      Y=Y
      Z=Z
      ALPHA=ALPHA
      BETA=BETA
      GAMMA=GAMMA
      TIME=0.0
      CALL RANDU(IX,IY,YFL)
      IX=IY
      VEL=PARA-PARB*YFL-PARC*ABS(GAMMA)
      RETURN
      END
```

```
       SUBROUTINE SIGMA(EN,SBE1,SBE2,SBI1,SBI2,SBF1,SBF2,SBC1,
      1SBC2,AD1,AD2,EBOUND,TMFP,PE1,PE2,PI1,PI2,PF1,PF2,PC1,J)
       DIMENSION SBE1(20),SBE2(20),SBI1(20),SBI2(20),SBF1(20)
      1SBF2(20),SBC1(20),SBC2(20),EBOUND(20)
10     CALL GROUP(EN,EBOUND,J,KGP)
       J=J
11     GO TO (12,14),KGP
12     J=0
13     RETURN
14     IF(20-J)60,60,20
20     SE1=FIND(EN,J,EBOUND,SBE1)
21     SI1=FIND(EN,J,EBOUND,SBI1)
22     SF1=FIND(EN,J,EBOUND,SBF1)
23     SC1=FIND(EN,J,EBOUND,SBC1)
24     IF(AD2)25,25,30
25     SE2=0.0
26     SI2=0.0
27     SF2=0.0
28     SC2=0.0
29     GO TO 40
30     SE2=FIND(EN,J,EBOUND,SBE2)
31     SI2=FIND(EN,J,EBOUND,SBI2)
32     SF2=FIND(EN,J,EBOUND,SBF2)
33     SC2=FIND(EN,J,EBOUND,SBC2)
40     TMFP=1.0/(SE1+SI1+SF1+SC1+SE2+SI2+SF2+SC2)
41     PE1=TMFP*SE1
42     PI1=TMFP*SI1
43     PF1=TMFP*SF1
44     IF(AD2)45,45,50
45     PC1=1.0-PE1-PI1-PF1
46     IF(PC1-0.0001)47,48,48
47     PC1=0.0
48     RETURN
50     PC1=TMFP*SC1
51     PE2=TMFP*SE2
52     PI2=TMFP*SI2
53     PF2=TMFP*SF2
54     RETURN
60     SE1=SBE1(20)
61     SI1=SBI1(20)
62     SF1=SBF1(20)
63     SC1=SBC1(20)
64     SE2=SBE2(20)
65     SI2=SBI2(20)
66     SF2=SBF2(20)
67     SC2=SBC2(20)
68     GO TO 40
       END
      SUBROUTINE GROUP(EN,EBOUND,J,KGP)
      DIMENSION EBOUND(20)
10    IF(EN-EBOUND(1))11,13,13
11    KGP=1
12    RETURN
13    J=20
14    IF(EN-EBOUND(J))15,91,91
15    J=10
16    IF(EN-EBOUND(J))17,91,29
17    J=5
18    IF(EN-EBOUND(J))19,91,25
19    J=2
20    IF(EN-EBOUND(J))90,91,21
21    J=J+1
22    IF(EN-EBOUND(J))90,91,23
23    J=J+1
24    IF(EN-EBOUND(J))90,91,91
25    J=7
26    IF(EN-EBOUND(J))27,91,21
27    J=J-1
28    GO TO 24
29    J=15
30    IF(EN-EBOUND(J))31,91,33
31    J=12
32    IF(EN-EBOUND(J))27,91,21
33    J=17
34    IF(EN-EBOUND(J))27,91,21
90    J=J-1
91    KGP=2
92    RETURN
      END
```

```
 FUNCTION FIND(EN,J,EBOUND,SBX)
 DIMENSION EBOUND(20),SBX(20)
 FIND=SBX(J)+(EN-EBOUND(J))*(SBX(J+1)-SBX(J))/(EBOUND(J+1
1)-EBOUND(J))
 RETURN
 END
```

```
      SUBROUTINE FLITE(DIST,TIMET,TMFP,VEL,IX)
10    CALL RANDU(IX,IY,YFL)
      IX=IY
12    IF(YFL-.0000454)10,10,13
13    C=ALOG(YFL)
      DIST=TMFP*(-C)
      IF(DIST)10,16,16
16    TIMET=DIST/VEL
      IF(TIMET)10,18,18
18    RETURN
      END
      SUBROUTINE POST(ALPHA,BETA,GAMMA,X,Y,Z,DIST,DISTB,TIME,
     1TIMET,VEL,KGEO)
      IF(DISTB-DIST)20,20,10
10    X=X+ALPHA*DIST
      Y=Y+BETA*DIST
      Z=Z+GAMMA*DIST
      TIME=TIME+TIMET
      KGEO=1
      RETURN
20    X=X+ALPHA*DISTB
      Y=Y+BETA*DISTB
      Z=Z+GAMMA*DISTB
      TIME=TIME+DISTB/VEL
      KGEO=2
      RETURN
      END
      SUBROUTINE DTPB(ALPHA,BETA,GAMMA,X,Y,Z,XMAX,YMAX,ZMAX,
     1DISTB)
      IF(ALPHA)2,1,2
1     D1=10000.0
      D2=10000.0
      GO TO 3
2     D1=(XMAX-X)/ALPHA
      D2=-(XMAX+X)/ALPHA
3     IF(BETA)5,4,5
4     D3=10000.0
      D4=10000.0
      GO TO 6
5     D3=(YMAX-Y)/BETA
      D4=-(YMAX-Y)/BETA
6     IF(GAMMA)8,7,8
7     D5=10000.0
      D6=10000.0
      GO TO 9
8     D5=(ZMAX-Z)/GAMMA
      D6=-(ZMAX+Z)/GAMMA
9     IF(D1)10,11,11
10    D1=10000.0
11    IF(D2)12,13,13
12    D2=10000.0
13    IF(D3)14,15,15
14    D3=10000.0
15    IF(D4)16,17,17
16    D4=10000.0
17    IF(D5)18,19,19
18    D5=10000.0
19    IF(D6)20,21,21
20    D6=10000.0
21    DISTB=AMIN1(D1,D2,D3,D4,D5,D6)
      IF(DISTB)23,24,24
23    DISTB=0.0
24    RETURN
      END
      SUBROUTINE DTCB(ALPHA,BETA,GAMMA,X,Y,Z,RMAX,ZMAX,DISTB)
      OMR=X*ALPHA+Y*BETA+Z*GAMMA
      R=SQRT(X**2+Y**2+Z**2)
      IF(GAMMA-1.0)9,8,9
8     D1=10000.0
      GO TO 20
9     D1=(Z*GAMMA-OMR+SQRT((Z*GAMMA-OMR)**2+(1.0-GAMMA**2)*(
     1RMAX**2+Z**2-R**2)))/(1.0-GAMMA**2)
      IF(D1)10,20,20
10    D1=-D1
18    IF(GAMMA)20,19,20
19    D2=10000.0
      D3=10000.0
      GO TO 33
20    D2=(ZMAX-Z)/GAMMA
      D3=-(ZMAX+Z)/GAMMA
      IF(D2)30,31,31
30    D2=10000.0
31    IF(D3)32,33,33
32    D3=10000.0
33    DISTB=AMIN1(D1,D2,D3)
      IF(DISTB)35,40,40
35    DISTB=-DISTB
40    RETURN
      END
      SUBROUTINE DTSB(ALPHA,BETA,GAMMA,X,Y,Z,RMAX,DISTB)
      OMR=X*ALPHA+Y*BETA+Z*GAMMA
      R=SQRT(X**2+Y**2+Z**2)
      DISTB=-OMR+SQRT(OMR**2+RMAX**2-R**2)
5     IF(DISTB)20,10,10
10    RETURN
20    DISTB=-DISTB
      GO TO 5
      END
      SUBROUTINE LEKTAL(TIME,VEL,TD,TCH,EMIN,ECH,KLEK,LEAK)
      DIMENSION LEAK(100,10)
10    ITIME=(TIME-TD)/TCH
      ITIME=ITIME+1
11    IF(ITIME-1)12,14,14
12    KLEK=1
13    RETURN
14    IF(100-ITIME)15,17,17
15    KLEK=2
16    RETURN
17    IEN=(0.5227*(VEL**2)-EMIN)/ECH
18    IF(IEN-1)19,21,21
19    KLEK=3
20    RETURN
21    IF(10-IEN)22,24,24
22    KLEK=4
23    RETURN
24    LEAK(ITIME,IEN)=LEAK(ITIME,IEN)+1
25    KLEK=5
26    RETURN
      END
      SUBROUTINE COLID1(PE1,PI1,PF1,KCOL,IX)
9     CALL RANDU(IX,IY,YFL)
      IX=IY
10    IF(YFL-PE1)20,11,11
11    IF(YFL-PE1-PI1)30,12,12
12    IF(YFL-PE1-PI1-PF1)40,13,13
13    KCOL=41
14    RETURN
20    KCOL=11
21    RETURN
30    KCOL=21
31    RETURN
40    KCOL=31
41    RETURN
      END
      SUBROUTINE COLI02(PE1,PE2,PI1,PI2,PF1,PF2,PC1,KCOL,IX)
9     CALL RANDU(IX,IY,YFL)
      IX=IY
10    IF(YFL-PE1)20,11,11
11    IF(YFL-PE1-PE2)30,12,12
12    IF(YFL-PE1-PE2-PI1)40,13,13
13    IF(YFL-PE1-PE2-PI1-PI2)50,14,14
14    IF(YFL-PE1-PE2-PI1-PI2-PF1)60,15,15
15    IF(YFL-PE1-PE2-PI1-PI2-PF1-PF2)70,16,16
16    IF(YFL-PE1-PE2-PI1-PI2-PE1-PE2-PC1)80,90,90
20    KCOL=11
21    RETURN
30    KCOL=12
31    RETURN
40    KCOL=21
41    RETURN
50    KCOL=22
51    RETURN
60    KCOL=31
61    RETURN
70    KCOL=32
71    RETURN
80    KCOL=41
81    RETURN
90    KCOL=42
91    RETURN
      END
      SUBROUTINE ELTAL(TIME,TD,TCH,Z,ZMAX,KELS,NELS)
      DIMENSION NELS(100,10)
10    ITIME=(TIME-TD)/TCH
      ITIME=ITIME+1
11    IF(ITIME-1)12,14,14
12    KELS=1
13    RETURN
14    IF(100-ITIME)15,17,17
15    KELS=2
16    RETURN
17    IZ=6.0+(5.0*Z)/ZMAX
18    IF(IZ-1)19,21,21
19    KELS=3
20    RETURN
21    IF(10-IZ)22,24,24
22    KELS=4
23    RETURN
24    KELS=5
25    NELS(ITIME,IZ)=NELS(ITIME,IZ)+1
26    RETURN
      END
      SUBROUTINE ELSCAT(ALPHA,BETA,GAMMA,VEL,A,ALIM,SLIM,AP,J,
     1IX)
      DIMENSION AP(10,20)
10    IF(VEL-SLIM)11,20,20
11    CALL RANDU(IX,IY,YFY)
      IX=IY
      GAMMAC=2.0*YFL-1.0
12    IF(A-ALIM)13,15,15
13    CALL CMLAB(ALPHA,BETA,GAMMA,GAMMAC,VEL,IX)
14    RETURN
15    CALL ISOANG(ALPHA,BETA,GAMMA,GAMMAC,VEL, IX)
16    RETURN
20    CALL ANGLE(J,AP,GAMMAC,IX)
21    GO TO 12
      END
      SUBROUTINE ANGLE(J,AP,GAMMAC,IX)
      DIMENSION AP(10,20)
      CALL RANDU(IX,IY,YFL)
      IX=IY
      M=10.0*YFL+1.0
      REM=YFL-0.1*FLOAT(M-1)
      IF(10-M)30,10,20
10    GAMMAC=AP(10,J)+(REM/0.1)*(1.0-AP(10,J))
      RETURN
20    GAMMAC=AP(M,J)+(REM/0.1)*(AP(M+1,J)-AP(M,J))
      RETURN
30    GAMMAC=1.0
      RETURN
      END
      SUBROUTINE CMLAB(ALPHA,BETA,GAMMA,GAMMAC,VEL,A,IX)
10    CALL RANDU(IX,IY,YFL)
      IX=IY
      R1=YFL
11    CALL RANDU(IX,IY,YFL)
      IX=IY
      R2=YFL
12    ETA=(2.0*R1-1.0)**2+(2.0*R2-1.0)**2
13    IF(ETA-1.0)14,14,10
14    ROOT=SQRT((1.0-GAMMAC**2)/ETA)
15    ALPHAC=(2.0*R1-1.0)*ROOT
16    BETAC=(2.0*R2-1.0)*ROOT
17    RTG=SQRT(1.0-GAMMA**2)
18    ALPHAP=((ALPHA*GAMMAXALPHAC-BETA*BETAC)/RTG)+ALPHA*
     1GAMMAC
19    BETAP=((BETA*GAMMA*ALPHAC+ALPHA*BETAC)/RTG)+BETA*GAMMAC
20    GAMMAP=-ALPHAC*RTG+GAMMA*GAMMAC
21    RTA=SQRT(1.0+A**2+2.0*A*GAMMAC)
22    ALPHA=(ALPHA+A*ALPHAP)/RTA
23    BETA=(BETA+A*BETAP)/RTA
24    GAMMA=(GAMMA+A*GAMMAP)/RTA
25    VEL=(VEL*RTA)/(A+1.0)
26    RETURN
      END
      SUBROUTINE ISOANG(ALPHA,BETA,GAMMA,GAMMAC,VEL,IX)
10    GAMMA=GAMMAC
11    CALL RANDU(IX,IY,YFL)
      IX=IY
      R1=YFL
12    CALL RANDU(IX,IY,YFL)
      IX=IY
      R2=YFL
13    ETA=(2.0*R1-1.0)**2+(2.0*R2-1.0)**2
      IF(ETA)20,20,14
14    IF(ETA-1.0)15,15,11
15    ROOT=SQRT((1.0-GAMMA**2)/ETA)
16    ALPHA=(2.0*R1-1.0)*ROOT
17    BETA=(2.0*R2-1.0)*ROOT
18    VEL=VEL
19    RETURN
20    GO TO 11
      END
      SUBROUTINE INTAL(TIME,TD,TCH,KINS,NINS)
      DIMENSION NINS(100)
10    ITIME=(TIME-TD)/TCH
      ITIME=ITIME+1
11    IF(ITIME-1)12,14,14
12    KINS=1
13    RETURN
14    IF(100-ITIME)15,17,17
15    KINS=2
16    RETURN
17    NINS(ITIME)=NINS(ITIME)+1
18    KINS=5
19    RETURN
      END
      SUBROUTINE LEVEL(VEL,SBL,VBOUND,PL,J)
      DIMENSION SBL(20,20),VBOUND(20),PL(20),SL(20)
      IF(20-J)10,10,20
10    DO 15 L=1,20
      SL(L)=SBL(L,20)
15    CONTINUE
      GO TO 25
20    DO 25 L=1,20
      SL(L)=SBL(L,J)+(VEL-VBOUND(J))*(SBL(L,J+1)-SBL(L,J))/(
     1VBOUND(J+1)-VBOUND(J))
25    CONTINUE
      SUM=0.0
      DO 30 L=1,20
      SUM=SUM+SL(L)
30    CONTINUE
31    SUMI=1.0/SUM
      DO 35 L=1,20
      PL(L)=SUMI*SL(L)
35    CONTINUE
      RETURN
      END
      SUBROUTINE INSCAT(ALPHA,BETA,GAMMA,VEL,A,CIN,P,PL,VL,VST
     1,KIA,IX)
      DIMENSION PL(20),VL(20),P(22)
10    GO TO (11,14),KIA
11    CALL RANDU(IX,IY,YFL)
      IX=IY
      GAMMAC=2.0*YFL-1.0
12    CALL ISOANG(ALPHA,BETA,GAMMA,GAMMAC,VEL,IX)
13    GO TO 20
14    CALL ANGLI(VEL,A,GAMMAC)
15    GO TO 12
20    IF(VEL-VST)21,30,30
21    CALL RANDU(IX,IY,YFL)
      IX=IY
22    L=1
23    SUM=0.0
24    SUM=SUM+PL(L)
25    IF(YFL-SUM)28,28,26
26    L=L+1
27    GO TO 24
28    IF(VEL**2-VL(L)**2)35,29,29
29    VEL=SQRT(VEL**2-VL(L)**2)
      RETURN
30    CALL INSPEC(VEL,CIN,P,IX)
31    RETURN
35    GO TO 11
      END
```

```
SUBROUTINE ANGLI(VEL,A,GAMMAC)
GAMMAC=1.0
VEL=VEL
A=A
RETURN
END
```

```
SUBROUTINE INSPEC(VEL,CIN,P,IX)
DIMENSION P(22)
EMAX=CIN*VEL
VMAX=SQRT(EMAX/0.5227)
CALL RANDU(IX,IY,YFL)
IX=IY
K=20.0*YFL+1.0
REM=YFL-0.05*FLOAT(K-1)
W=P(K)+(REM/0.05)*(P(K+1)-P(K))
VEL=W*VMAX
RETURN
END
```

```
      SUBROUTINE CAPTAL(TIME,TD,TCH,KCAP,KAPT)
      DIMENSION KAPT(100)
10    ITIME=(TIME-TD)/TCH
      ITIME=ITIME+1
11    IF(ITIME-1)12,14,14
12    KCAP=1
13    RETURN
14    IF(100-ITIME)15,17,17
15    KCAP=2
16    RETURN
17    KAPT(ITIME)=KAPT(ITIME)+1
18    KCAP=5
19    RETURN
      END
      SUBROUTINE FISTAL(TIME,TD,TCH,KFIS,NFIS)
      DIMENSION NFIS(100)
10    ITIME=(TIME-TD)/TCH
      ITIME=ITIME+1
11    IF(ITIME-1)12,14,14
12    KFIS=1
13    RETURN
14    IF(100-ITIME)15,17,17
15    KFIS=2
16    RETURN
17    NFIS(ITIME)=NFIS(ITIME)+1
18    KFIS=5
19    RETURN
      END
      SUBROUTINE FISSN(X,Y,Z,VEL,TIME,FP,FNU,DELNU,NF,KT,IX)
      DIMENSION FP(22)
      FISNO=FNU+DELNU*(VEL**2)
      IF(FISNO-3.0)20,30,40
20    CALL RANDU(IX,IY,YFL)
      IX=IY
      R1=YFL+2.0
      IF(R1-FISNO)30,30,25
25    I=2
      GO TO 50
30    I=3
      GO TO 50
40    IF(FISNO-4.0)41,49,49
41    CALL RANDU(IX,IY,YFL)
      IX=IY
      R2=YFL+3.0
      IF(R2-FISNO)49,49,45
45    I=3
      GO TO 50
49    I=4
50    DO 60 N=1,I
 51   CALL RANDU(IX,IY,YFL)
      IX=IY
      K=20.0*YFL+1.0
      REM=YFL-0.05*FLOAT(K-1)
      PARA=FP(K)+(REM/0.05)*(FP(K+1)-FP(K))
 53   THETA=TIME
      XS=X
      YS=Y
      ZS=Z
      WRITE(KT)XS,YS,ZS,PARA,THETA
60    NF=NF+1
      RETURN
      END
```