CHAPTER 1

PRIMER

WHAT IS COVERED IN CHAPTER 1

Brief explanation of the Monte Carlo method.
Summary of MCNP features.
Introduction to geometry.
Description of MCNP data input illustrated by a sample problem.
How to run MCNP.
Tips on problem setup.

Chapter 1 will enable the novice to start using MCNP, assuming very little knowledge of the Monte Carlo method and no experience with MCNP. The primer begins with a short discussion of the Monte Carlo method. Five features of MCNP are introduced: (1) nuclear data and reactions, (2) source specifications, (3) tallies and output, (4) estimation of errors, and (5) variance reduction. The third section explains MCNP geometry setup, including the concept of cells and surfaces. A general description of an input deck is followed by a sample problem and a detailed description of the input cards used in the sample problem. Section V tells how to run MCNP, VI lists tips for setting up correct problems and running them efficiently, and VII is the references for Chapter 1. The word “card” is used throughout this document to describe a single line of input up to 80 characters.

I. MCNP AND THE MONTE CARLO METHOD

MCNP is a general-purpose, continuous-energy, generalized-geometry, time-dependent, coupled neutron/photon/electron Monte Carlo transport code. It can be used in several transport modes: neutron only, photon only, electron only, combined neutron/photon transport where the photons are produced by neutron interactions, neutron/photon/electron, photon/electron, or electron/photon. The neutron energy regime is from 10^{-11} MeV to 20 MeV, and the photon and electron energy regimes are from 1 keV to 1000 MeV. The capability to calculate keff eigenvalues for fissile systems is also a standard feature.

The user creates an input file that is subsequently read by MCNP. This file contains information about the problem in areas such as:

the geometry specification,
the description of materials and selection of cross-section evaluations,the location and characteristics of the neutron, photon, or electron source,
the type of answers or tallies desired, and
any variance reduction techniques used to improve efficiency.

Each area will be discussed in the primer by use of a sample problem. Remember five “rules’’ when running a Monte Carlo calculation. They will be more meaningful as you read this manual and gain experience with MCNP, but no matter how sophisticated a user you may become, never forget the following five points:

1. Define and sample the geometry and source well;
2. You cannot recover lost information;
3. Question the stability and reliability of results;
4. Be conservative and cautious with variance reduction biasing; and
5. The number of histories run is not indicative of the quality of the answer.

The following sections compare Monte Carlo and deterministic methods and provide a simple description of the Monte Carlo method.

A. Monte Carlo Method vs Deterministic Method

Monte Carlo methods are very different from deterministic transport methods. Deterministic methods, the most common of which is the discrete ordinates method, solve the transport equation for the average particle behavior. By contrast, Monte Carlo does not solve an explicit equation, but rather obtains answers by simulating individual particles and recording some aspects (tallies) of their average behavior. The average behavior of particles in the physical system is then inferred (using the central limit theorem) from the average behavior of the simulated particles. Not only are Monte Carlo and deterministic methods very different ways of solving a problem, even what constitutes a solution is different. Deterministic methods typically give fairly complete information (for example, flux) throughout the phase space of the problem. Monte Carlo supplies information only about specific tallies requested by the user.

When Monte Carlo and discrete ordinates methods are compared, it is often said that Monte Carlo solves the integral transport equation, whereas discrete ordinates solves the integro-differential transport equation. Two things are misleading about this statement. First, the integral and integro-differential transport equations are two different forms of the same equation; if one is solved, the other is solved. Second, Monte Carlo “solves” a transport problem by simulating particle histories rather than by solving an equation. No transport equation need ever be written to solve a transport problem by Monte Carlo. Nonetheless, one can derive an equation that describes the probability density of particles in phase space; this equation turns out to be the same as the integral transport equation.

Without deriving the integral transport equation, it is instructive to investigate why the discrete ordinates method is associated with the integro-differential equation and Monte Carlo with the integral equation. The discrete ordinates method visualizes the phase space to be divided into many small boxes, and the particles move from one box to another. In the limit as the boxes get

progressively smaller, particles moving from box to box take a differential amount of time to move a differential distance in space. In the limit this approaches the integro-differential transport equation, which has derivatives in space and time. By contrast, Monte Carlo transports particles between events (for example, collisions) that are separated in space and time. Neither differential space nor time are inherent parameters of Monte Carlo transport. The integral equation does not have time or space derivatives.

Monte Carlo is well suited to solving complicated three-dimensional, time-dependent problems. Because the Monte Carlo method does not use phase space boxes, there are no averaging approximations required in space, energy, and time. This is especially important in allowing detailed representation of all aspects of physical data.

B. The Monte Carlo Method

Monte Carlo can be used to duplicate theoretically a statistical process (such as the interaction of nuclear particles with materials) and is particularly useful for complex problems that cannot be modeled by computer codes that use deterministic methods. The individual probabilistic events that comprise a process are simulated sequentially. The probability distributions governing these events are statistically sampled to describe the total phenomenon. In general, the simulation is performed on a digital computer because the number of trials necessary to adequately describe the phenomenon is usually quite large. The statistical sampling process is based on the selection of random numbers—analogous to throwing dice in a gambling casino—hence the name “Monte Carlo.” In particle transport, the Monte Carlo technique is pre-eminently realistic (a theoretical experiment). It consists of actually following each of many particles from a source throughout its life to its death in some terminal category (absorption, escape, etc.). Probability distributions are randomly sampled using transport data to determine the outcome at each step of its life.

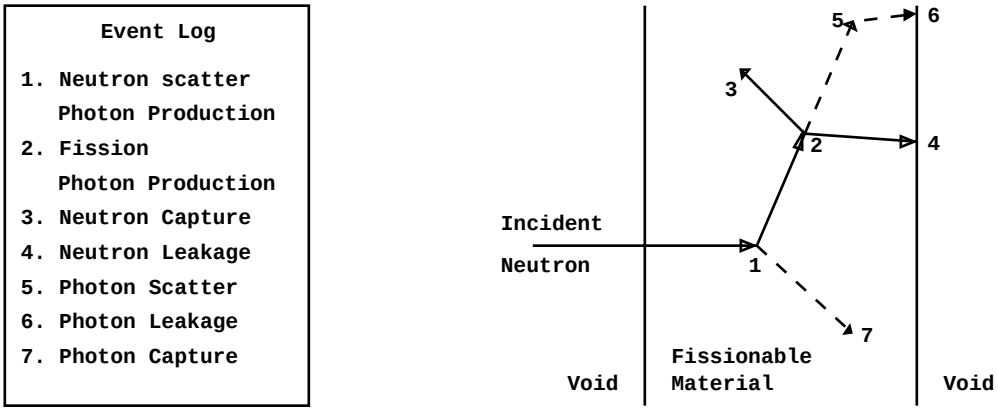

Figure 1-1.

Figure 1.1 represents the random history of a neutron incident on a slab of material that can undergo fission. Numbers between 0 and 1 are selected randomly to determine what (if any) and where interaction takes place, based on the rules (physics) and probabilities (transport data) governing the processes and materials involved. In this particular example, a neutron collision occurs at event 1. The neutron is scattered in the direction shown, which is selected randomly from the physical scattering distribution. A photon is also produced and is temporarily stored, or banked, for later analysis. At event 2, fission occurs, resulting in the termination of the incoming neutron and the birth of two outgoing neutrons and one photon. One neutron and the photon are banked for later analysis. The first fission neutron is captured at event 3 and terminated. The banked neutron is now retrieved and, by random sampling, leaks out of the slab at event 4. The fission-produced photon has a collision at event 5 and leaks out at event 6. The remaining photon generated at event 1 is now followed with a capture at event 7. Note that MCNP retrieves banked particles such that the last particle stored in the bank is the first particle taken out.

This neutron history is now complete. As more and more such histories are followed, the neutron and photon distributions become better known. The quantities of interest (whatever the user requests) are tallied, along with estimates of the statistical precision (uncertainty) of the results.

II. INTRODUCTION TO MCNP FEATURES

Various features, concepts, and capabilities of MCNP are summarized in this section. More detail concerning each topic is available in later chapters or appendices.

A. Nuclear Data and Reactions

MCNP uses continuous-energy nuclear and atomic data libraries. The primary sources of nuclear data are evaluations from the Evaluated Nuclear Data File (ENDF)[1] system, the Evaluated Nuclear Data Library (ENDL)[2] and the Activation Library (ACTL)[3] compilations from Livermore, and evaluations from the Applied Nuclear Science (T–2) Group[4,5,6] at Los Alamos. Evaluated data are processed into a format appropriate for MCNP by codes such as NJOY.[7] The processed nuclear data libraries retain as much detail from the original evaluations as is feasible to faithfully reproduce the evaluator's intent.

Nuclear data tables exist for neutron interactions, neutron-induced photons, photon interactions, neutron dosimetry or activation, and thermal particle scattering $S(\alpha,\beta)$. Photon and electron data are atomic rather than nuclear in nature. Each data table available to MCNP is listed on a directory file, XSDIR. Users may select specific data tables through unique identifiers for each table, called ZAIDs. These identifiers generally contain the atomic number Z, mass number A, and library specifier ID.

Over 500 neutron interaction tables are available for approximately 100 different isotopes and elements. Multiple tables for a single isotope are provided primarily because data have been derived from different evaluations, but also because of different temperature regimes and different processing tolerances. More neutron interaction tables are constantly being added as new and revised evaluations become available. Neutron–induced photon production data are given as part of the neutron interaction tables when such data are included in the evaluations.

Photon interaction tables exist for all elements from Z = 1 through Z = 94. The data in the photon interaction tables allow MCNP to account for coherent and incoherent scattering, photoelectric absorption with the possibility of fluorescent emission, and pair production. Scattering angular distributions are modified by atomic form factors and incoherent scattering functions.

Cross sections for nearly 2000 dosimetry or activation reactions involving over 400 target nuclei in ground and excited states are part of the MCNP data package. These cross sections can be used as energy-dependent response functions in MCNP to determine reaction rates but cannot be used as transport cross sections.

Thermal data tables are appropriate for use with the $S(\alpha,\beta)$ scattering treatment in MCNP. The data include chemical (molecular) binding and crystalline effects that become important as the neutron's energy becomes sufficiently low. Data at various temperatures are available for light and heavy water, beryllium metal, beryllium oxide, benzene, graphite, polyethylene, and zirconium and hydrogen in zirconium hydride.

B. Source Specification

MCNP's generalized user-input source capability allows the user to specify a wide variety of source conditions without having to make a code modification. Independent probability distributions may be specified for the source variables of energy, time, position, and direction, and for other parameters such as starting cell(s) or surface(s). Information about the geometrical extent of the source can also be given. In addition, source variables may depend on other source variables (for example, energy as a function of angle) thus extending the built-in source capabilities of the code. The user can bias all input distributions.

In addition to input probability distributions for source variables, certain built-in functions are available. These include various analytic functions for fission and fusion energy spectra such as Watt, Maxwellian, and Gaussian spectra; Gaussian for time; and isotropic, cosine, and monodirectional for direction. Biasing may also be accomplished by special built–in functions.

A surface source allows particles crossing a surface in one problem to be used as the source for a subsequent problem. The decoupling of a calculation into several parts allows detailed design or analysis of certain geometrical regions without having to rerun the entire problem from the

beginning each time. The surface source has a fission volume source option that starts particles from fission sites where they were written in a previous run.

MCNP provides the user three methods to define an initial criticality source to estimate k_{eff}, the ratio of neutrons produced in successive generations in fissile systems.

C. Tallies and Output

The user can instruct MCNP to make various tallies related to particle current, particle flux, and energy deposition. MCNP tallies are normalized to be per starting particle except for a few special cases with criticality sources. Currents can be tallied as a function of direction across any set of surfaces, surface segments, or sum of surfaces in the problem. Charge can be tallied for electrons and positrons. Fluxes across any set of surfaces, surface segments, sum of surfaces, and in cells, cell segments, or sum of cells are also available. Similarly, the fluxes at designated detectors (points or rings) are standard tallies. Heating and fission tallies give the energy deposition in specified cells. A pulse height tally provides the energy distribution of pulses created in a detector by radiation. In addition, particles may be flagged when they cross specified surfaces or enter designated cells, and the contributions of these flagged particles to the tallies are listed separately. Tallies such as the number of fissions, the number of absorptions, the total helium production, or any product of the flux times the approximately 100 standard ENDF reactions plus several nonstandard ones may be calculated with any of the MCNP tallies. In fact, any quantity of the form

$$C = \int \phi(E) f(E) dE$$

can be tallied, where $\phi(\mathbf{E})$ is the energy-dependent fluence, and **f(E)** is any product or summation of the quantities in the cross-section libraries or a response function provided by the user. The tallies may also be reduced by line-of-sight attenuation. Tallies may be made for segments of cells and surfaces without having to build the desired segments into the actual problem geometry. All tallies are functions of time and energy as specified by the user and are normalized to be per starting particle.

In addition to the tally information, the output file contains tables of standard summary information to give the user a better idea of how the problem ran. This information can give insight into the physics of the problem and the adequacy of the Monte Carlo simulation. If errors occur during the running of a problem, detailed diagnostic prints for debugging are given. Printed with each tally is also its statistical relative error corresponding to one standard deviation. Following the tally is a detailed analysis to aid in determining confidence in the results. Ten pass/no pass checks are made for the user-selectable tally fluctuation chart (TFC) bin of each tally. The quality of the confidence interval still cannot be guaranteed because portions of the problem phase space possibly still have not been sampled. Tally fluctuation charts, described in the following section, are also

automatically printed to show how a tally mean, error, variance of the variance, and slope of the largest history scores fluctuate as a function of the number of histories run.

Tally results can be displayed graphically, either while the code is running or in a separate postprocessing mode.

D. Estimation of Monte Carlo Errors

MCNP tallies are normalized to be per starting particle and are printed in the output accompanied by a second number *R*, which is the estimated relative error defined to be one estimated standard deviation of the mean $S\bar{x}$ divided by the estimated mean $\bar{x}$. In MCNP, the quantities required for this error estimate—the tally and its second moment—are computed after each complete Monte Carlo history, which accounts for the fact that the various contributions to a tally from the same history are correlated. For a well-behaved tally, *R* will be proportional to $1/\sqrt{N}$ where *N* is the number of histories. Thus, to halve *R*, we must increase the total number of histories fourfold. For a poorly behaved tally, *R* may increase as the number of histories increases.

The estimated relative error can be used to form confidence intervals about the estimated mean, allowing one to make a statement about what the true result is. The Central Limit Theorem states that as *N* approaches infinity there is a 68% chance that the true result will be in the range $\bar{x}(1 \pm R)$ and a 95% chance in the range $\bar{x}(1 \pm 2R)$. *It is extremely important to note that these confidence statements refer only to the* <u>*precision*</u> *of the Monte Carlo calculation itself and not to the* <u>*accuracy*</u> *of the result compared to the true physical value.* A statement regarding accuracy requires a detailed analysis of the uncertainties in the physical data, modeling, sampling techniques, and approximations, etc., used in a calculation.

The guidelines for interpreting the quality of the confidence interval for various values of *R* are listed in Table 1.1.

TABLE 1.1:
Guidelines for Interpreting the Relative Error *R**

Range of R	Quality of the Tally
0.5 to 1.0	Not meaningful
0.2 to 0.5	Factor of a few
0.1 to 0.2	Questionable
< 0.10	Generally reliable
< 0.05	Generally reliable for point detectors

* $R = S_{\bar{x}}/\bar{x}$ and represents the estimated relative error at the 1σ level. These interpretations of *R* assume that all portions of the problem phase space are being sampled well by the Monte Carlo process.

For all tallies except next-event estimators, hereafter referred to as point detector tallies, the quantity *R* should be less than 0.10 to produce generally reliable confidence intervals. Point detector results tend to have larger third and fourth moments of the individual tally distributions, so a smaller value of *R*, < 0.05, is required to produce generally reliable confidence intervals. The estimated uncertainty in the Monte Carlo result must be presented with the tally so that all are aware of the estimated precision of the results.

Keep in mind the footnote to Table 1.1. For example, if an important but highly unlikely particle path in phase space has not been sampled in a problem, the Monte Carlo results will not have the correct expected values and the confidence interval statements may not be correct. The user can guard against this situation by setting up the problem so as not to exclude any regions of phase space and by trying to sample all regions of the problem adequately.

Despite one's best effort, an important path may not be sampled often enough, causing confidence interval statements to be incorrect. To try to inform the user about this behavior, MCNP calculates a figure of merit *(FOM)* for one tally bin of each tally as a function of the number of histories and prints the results in the tally fluctuation charts at the end of the output. The *FOM* is defined as

$$FOM \equiv 1/(R^2 T)$$

where *T* is the computer time in minutes. The more efficient a Monte Carlo calculation is, the larger the *FOM* will be because less computer time is required to reach a given value of *R*.

The *FOM* should be approximately constant as *N* increases because R^2 is proportional to $1/N$ and *T* is proportional to *N*. *Always examine the tally fluctuation charts to be sure that the tally appears well behaved, as evidenced by a fairly constant FOM*. A sharp decrease in the *FOM* indicates that a seldom-sampled particle path has significantly affected the tally result and relative error estimate. In this case, the confidence intervals may not be correct for the fraction of the time that statistical theory would indicate. Examine the problem to determine what path is causing the large scores and try to redefine the problem to sample that path much more frequently.

After each tally, an analysis is done and additional useful information is printed about the TFC tally bin result. The nonzero scoring efficiency, the zero and nonzero score components of the relative error, the number and magnitude of negative history scores, if any, and the effect on the result if the largest observed history score in the TFC were to occur again on the very next history are given. A table just before the TFCs summarizes the results of these checks for all tallies in the problem. Ten statistical checks are made and summarized in table 160 after each tally, with a pass yes/no criterion. The empirical history score probability density function (PDF) for the TFC bin of each tally is calculated and displayed in printed plots.

The TFCs at the end of the problem include the variance of the variance (an estimate of the error of the relative error), and the slope (the estimated exponent of the PDF large score behavior) as a function of the number of particles started.

All this information provides the user with statistical information to aid in forming valid confidence intervals for Monte Carlo results. There is no GUARANTEE, however. The possibility always exists that some as yet unsampled portion of the problem may change the confidence interval if more histories were calculated. Chapter 2 contains more information about estimation of Monte Carlo precision.

E. Variance Reduction

As noted in the previous section, R (the estimated relative error) is proportional to $1/\sqrt{N}$, where N is the number of histories. For a given MCNP run, the computer time T consumed is proportional to N. Thus $R = C/\sqrt{T}$, where C is a positive constant. There are two ways to reduce R: (1) increase T and/or (2) decrease C. Computer budgets often limit the utility of the first approach. For example, if it has taken 2 hours to obtain R=0.10, then 200 hours will be required to obtain R=0.01. For this reason MCNP has special variance reduction techniques for decreasing C. (Variance is the square of the standard deviation.) The constant C depends on the tally choice and/or the sampling choices.

1. Tally Choice

As an example of the tally choice, note that the fluence in a cell can be estimated either by a collision estimate or a track length estimate. The collision estimate is obtained by tallying $1/\Sigma_t$ (Σ_t=macroscopic total cross section) at each collision in the cell and the track length estimate is obtained by tallying the distance the particle moves while inside the cell. Note that as Σ_t gets very small, very few particles collide but give enormous tallies when they do, a high variance situation (see page 2–109). In contrast, the track length estimate gets a tally from every particle that enters the cell. For this reason MCNP has track length tallies as standard tallies, whereas the collision tally is not standard in MCNP, except for estimating k_{eff}.

2. Nonanalog Monte Carlo

Explaining how sampling affects C requires understanding of the nonanalog Monte Carlo model.

The simplest Monte Carlo model for particle transport problems is the analog model that uses the natural probabilities that various events occur (for example, collision, fission, capture, etc.). Particles are followed from event to event by a computer, and the next event is always sampled (using the random number generator) from a number of possible next events according to the natural event probabilities. This is called the analog Monte Carlo model because it is directly analogous to the naturally occurring transport.

The analog Monte Carlo model works well when a significant fraction of the particles contribute to the tally estimate and can be compared to detecting a significant fraction of the particles in the physical situation. There are many cases for which the fraction of particles detected is very small, less than 10^{-6}. For these problems analog Monte Carlo fails because few, if any, of the particles tally, and the statistical uncertainty in the answer is unacceptable.

Although the analog Monte Carlo model is the simplest conceptual probability model, there are other probability models for particle transport. They estimate the same average value as the analog Monte Carlo model, while often making the variance (uncertainty) of the estimate much smaller than the variance for the analog estimate. Practically, this means that problems that would be impossible to solve in days of computer time can be solved in minutes of computer time.

A nonanalog Monte Carlo model attempts to follow “interesting” particles more often than “uninteresting” ones. An “interesting” particle is one that contributes a large amount to the quantity (or quantities) that needs to be estimated. There are many nonanalog techniques, and they all are meant to increase the odds that a particle scores (contributes). To ensure that the average score is the same in the nonanalog model as in the analog model, the score is modified to remove the effect of biasing (changing) the natural odds. Thus, if a particle is artificially made q times as likely to execute a given random walk, then the particle’s score is weighted by (multiplied by) $1/q$. The average score is thus preserved because the average score is the sum, over all random walks, of the probability of a random walk multiplied by the score resulting from that random walk.

A nonanalog Monte Carlo technique will have the same expected tallies as an analog technique if the expected weight executing any given random walk is preserved. For example, a particle can be split into two identical pieces and the tallies of each piece are weighted by 1/2 of what the tallies would have been without the split. Such nonanalog, or variance reduction, techniques can often decrease the relative error by sampling naturally rare events with an unnaturally high frequency and weighting the tallies appropriately.

3. Variance Reduction Tools in MCNP

There are four classes of variance reduction techniques[8] that range from the trivial to the esoteric.

Truncation Methods are the simplest of variance reduction methods. They speed up calculations by truncating parts of phase space that do not contribute significantly to the solution. The simplest example is geometry truncation in which unimportant parts of the geometry are simply not modeled. Specific truncation methods available in MCNP are energy cutoff and time cutoff.

Population Control Methods use particle splitting and Russian roulette to control the number of samples taken in various regions of phase space. In important regions many samples of low weight are tracked, while in unimportant regions few samples of high weight are tracked. A weight adjustment is made to ensure that the problem solution remains unbiased. Specific population

control methods available in MCNP are geometry splitting and Russian roulette, energy splitting/ roulette, weight cutoff, and weight windows.

Modified Sampling Methods alter the statistical sampling of a problem to increase the number of tallies per particle. For any Monte Carlo event it is possible to sample from any arbitrary distribution rather than the physical probability as long as the particle weights are then adjusted to compensate. Thus, with modified sampling methods, sampling is done from distributions that send particles in desired directions or into other desired regions of phase space such as time or energy, or change the location or type of collisions. Modified sampling methods in MCNP include the exponential transform, implicit capture, forced collisions, source biasing, and neutron-induced photon production biasing.

Partially-Deterministic Methods are the most complicated class of variance reduction methods. They circumvent the normal random walk process by using deterministic-like techniques, such as next event estimators, or by controlling the random number sequence. In MCNP these methods include point detectors, DXTRAN, and correlated sampling.

Variance reduction techniques, used correctly, can greatly help the user produce a more efficient calculation. Used poorly, they can result in a wrong answer with good statistics and few clues that anything is amiss. Some variance reduction methods have general application and are not easily misused. Others are more specialized and attempts to use them carry high risk. The use of weight windows tends to be more powerful than the use of importances but typically requires more input data and more insight into the problem. The exponential transform for thick shields is not recommended for the inexperienced user; rather, use many cells with increasing importances (or decreasing weight windows) through the shield. Forced collisions are used to increase the frequency of random walk collisions within optically thin cells but should be used only by an experienced user. The point detector estimator should be used with caution, as should DXTRAN.

For many problems, variance reduction is not just a way to speed up the problem but is absolutely necessary to get any answer at all. Deep penetration problems and pipe detector problems, for example, will run too slowly by factors of trillions without adequate variance reduction. Consequently, users have to become skilled in using the variance reduction techniques in MCNP. Most of the following techniques cannot be used with the pulse height tally.

The following summarizes briefly the main MCNP variance reduction techniques. Detailed discussion is in Chapter 2, page 2–127.

1. ***Energy cutoff:*** Particles whose energy is out of the range of interest are terminated so that computation time is not spent following them.
2. ***Time cutoff***: Like the energy cutoff but based on time.

3. ***Geometry splitting with Russian roulette:*** Particles transported from a region of higher importance to a region of lower importance (where they will probably contribute little to the desired problem result) undergo Russian roulette; that is, some of those particles will be killed a certain fraction of the time, but survivors will be counted more by increasing their weight the remaining fraction of the time. In this way, unimportant particles are followed less often, yet the problem solution remains undistorted. On the other hand, if a particle is transported to a region of higher importance (where it will likely contribute to the desired problem result), it may be split into two or more particles (or tracks), each with less weight and therefore counting less. In this way, important particles are followed more often, yet the solution is undistorted because on average total weight is conserved.
4. ***Energy splitting/Russian roulette:*** Particles can be split or rouletted upon entering various user–supplied energy ranges. Thus important energy ranges can be sampled more frequently by splitting the weight among several particles and less important energy ranges can be sampled less frequently by rouletting particles.
5. ***Weight cutoff/Russian roulette:*** If a particle weight becomes so low that the particle becomes insignificant, it undergoes Russian roulette. Most particles are killed, and some particles survive with increased weight. The solution is unbiased because total weight is conserved, but computer time is not wasted on insignificant particles.
6. ***Weight window:*** As a function of energy, geometrical location, or both, low–weighted particles are eliminated by Russian roulette and high–weighted particles are split. This technique helps keep the weight dispersion within reasonable bounds throughout the problem. An importance generator is available that estimates the optimal limits for a weight window.
7. ***Exponential transformation:*** To transport particles long distances, the distance between collisions in a preferred direction is artificially increased and the weight is correspondingly artifically decreased. Because large weight fluctuations often result, it is <u>highly recommended</u> that the weight window be used with the exponential transform.
8. ***Implicit capture:*** When a particle collides, there is a probability that it is captured by the nucleus. In analog capture, the particle is killed with that probability. In implicit capture, also known as survival biasing, the particle is never killed by capture; instead, its weight is reduced by the capture probability at each collision. Important particles are permitted to survive by not being lost to capture. On the other hand, if particles are no longer considered useful after undergoing a few collisions, analog capture efficiently gets rid of them.
9. ***Forced collisions:*** A particle can be forced to undergo a collision each time it enters a designated cell that is almost transparent to it. The particle and its weight are appropriately split into a collided and uncollided part. Forced collisions are often used to generate contributions to point detectors, ring detectors, or DXTRAN spheres.
10. ***Source variable biasing:*** Source particles with phase space variables of more importance are emitted with a higher frequency but with a compensating lower weight

than are less important source particles. This technique can be used with pulse height tallies.

11. ***Point and ring detectors:*** When the user wishes to tally a flux–related quantity at a point in space, the probability of transporting a particle precisely to that point is vanishingly small. Therefore, pseudoparticles are directed to the point instead. Every time a particle history is born in the source or undergoes a collision, the user may require that a pseudoparticle be tallied at a specified point in space. In this way, many pseudoparticles of low weight reach the detector, which is the point of interest, even though no particle histories could ever reach the detector. For problems with rotational symmetry, the point may be represented by a ring to enhance the efficiency of the calculation.

12. ***DXTRAN***: DXTRAN, which stands for deterministic transport, improves sampling in the vicinity of detectors or other tallies. It involves deterministically transporting particles on collision to some arbitrary, user–defined sphere in the neighborhood of a tally and then calculating contributions to the tally from these particles. Contributions to the detectors or to the DXTRAN spheres can be controlled as a function of geometric cell or as a function of the relative magnitude of the contribution to the detector or DXTRAN sphere.

    The DXTRAN method is a way of obtaining large numbers of particles on user–specified "DXTRAN spheres." DXTRAN makes it possible to obtain many particles in a small region of interest that would otherwise be difficult to sample. Upon sampling a collision or source density function, DXTRAN estimates the correct weight fraction that should scatter toward, and arrive without collision at, the surface of the sphere. The DXTRAN method then puts this correct weight on the sphere. The source or collision event is sampled in the usual manner, except that the particle is killed if it tries to enter the sphere because all particles entering the sphere have already been accounted for deterministically.

13. ***Correlated sampling:*** The sequence of random numbers in the Monte Carlo process is chosen so that statistical fluctuations in the problem solution will not mask small variations in that solution resulting from slight changes in the problem specification. The ith history will always start at the same point in the random number sequence no matter what the previous $i–1$ particles did in their random walks.

III. MCNP GEOMETRY

We will present here only basic information about geometry setup, surface specification, and cell and surface card input. Areas of further interest would be the complement operator, use of parentheses, and repeated structure and lattice definitions, found in Chapter 2. Chapter 4 contains geometry examples and is recommended as a next step. Chapter 3 has detailed information about the format and entries on cell and surface cards and discusses macrobodies.

The geometry of MCNP treats an arbitrary three-dimensional configuration of user-defined materials in geometric cells bounded by first- and second-degree surfaces and fourth-degree elliptical tori. The cells are defined by the intersections, unions, and complements of the regions bounded by the surfaces. Surfaces are defined by supplying coefficients to the analytic surface equations or, for certain types of surfaces, known points on the surfaces.

MCNP has a more general geometry than is available in most combinatorial geometry codes. Rather than combining several predefined geometrical bodies, as in a combinatorial geometry scheme, MCNP gives the user the added flexibility of defining geometrical regions from all the first and second degree surfaces of analytical geometry and elliptical tori and then of combining them with Boolean operators. The code does extensive internal checking to find input errors. In addition, the geometry-plotting capability in MCNP helps the user check for geometry errors.

MCNP treats geometric cells in a Cartesian coordinate system. The surface equations recognized by MCNP are listed in Table 3.1 on page 3–14. The particular Cartesian coordinate system used is arbitrary and user defined, but the right–handed system shown in Figure 1.2 is often chosen.

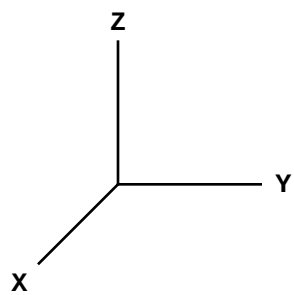

Figure 1-2.

Using the bounding surfaces specified on cell cards, MCNP tracks particles through the geometry, calculates the intersection of a track's trajectory with each bounding surface, and finds the minimum positive distance to an intersection. If the distance to the next collision is greater than this minimum distance and there are no DXTRAN spheres along the track, the particle leaves the current cell. At the appropriate surface intersection, MCNP finds the correct cell that the particle will enter by checking the sense of the intersection point for each surface listed for the cell. When a complete match is found, MCNP has found the correct cell on the other side and the transport continues.

A. Cells

When cells are defined, an important concept is that of the *sense* of all points in a cell with respect to a bounding surface. Suppose that $s = f(x, y, z)) = 0$ is the equation of a surface in the

problem. For any set of points (x,y,z), if $s = 0$ the points are on the surface. However, for points not on the surface, if s is negative, the points are said to have a negative sense with respect to that surface and, conversely, a positive sense if s is positive. For example, a point at $x = 3$ has a positive sense with respect to the plane $x - 2 = 0$. That is, the equation $x - D = 3 - 2 = s = 1$ is positive for $x = 3$ (where D = constant).

Cells are defined on cells cards. Each cell is described by a cell number, material number, and material density followed by a list of operators and signed surfaces that bound the cell. If the sense is positive, the sign can be omitted. The material number and material density can be replaced by a single zero to indicate a void cell. The cell number must begin in columns 1–5. The remaining entries follow, separated by blanks. A more complete description of the cell card format can be found on page 1–23. Each surface divides all space into two regions, one with positive sense with respect to the surface and the other with negative sense. The geometry description defines the cell to be the intersection, union, and/or complement of the listed regions.

The subdivision of the physical space into cells is not necessarily governed only by the different material regions, but may be affected by problems of sampling and variance reduction techniques (such as splitting and Russian roulette), the need to specify an unambiguous geometry, and the tally requirements. The tally segmentation feature may eliminate most of the tally requirements.

Be cautious about making any one cell very complicated. With the union operator and disjointed regions, a very large geometry can be set up with just one cell. The problem is that for each track flight between collisions in a cell, the intersection of the track with <u>each</u> bounding surface of the cell is calculated, a calculation that can be costly if a cell has many surfaces. As an example, consider Figure 1.3a. It is just a lot of parallel cylinders and is easy to set up. However, the cell containing all the little cylinders is bounded by fourteen surfaces (counting a top and bottom). A much more efficient geometry is seen in Figure 1.3b, where the large cell has been broken up into a number of smaller cells.

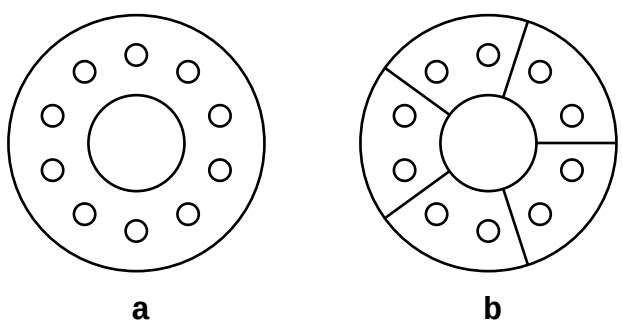

Figure 1-3.

1. Cells Defined by Intersections of Regions of Space

The intersection operator in MCNP is implicit; it is simply the blank space between two surface numbers on the cell card.

If a cell is specified using only intersections,all points in the cell must have the same sense with respect to a given bounding surface. This means that, for each bounding surface of a cell, all points in the cell must remain on only one side of any particular surface. Thus, there can be no concave corners in a cell specified only by intersections. Figure 1.4, a cell formed by the intersection of five surfaces (ignore surface 6 for the time being), illustrates the problem of concave corners by allowing a particle (or point) to be on two sides of a surface in one cell. Surfaces 3 and 4 form a concave corner in the cell such that points p_1 and p_2 are on the same side of surface 4 (that is, have the same sense with respect to 4) but point p_3 is on the other side of surface 4 (opposite sense). Points p_2 and p_3 have the same sense with respect to surface 3, but p_1 has the opposite sense. One way to remedy this dilemma (and there are others) is to add surface 6 between the 3/4 corner and surface 1 to divide the original cell into two cells.

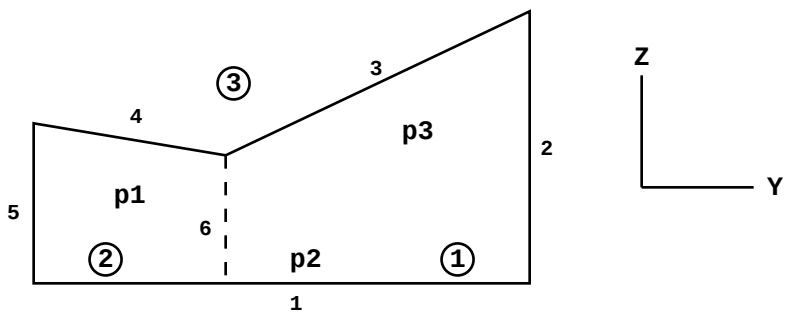

Figure 1-4.

With surface 6 added to Figure 1.4, the cell to the right of surface 6 is number~1 (cells indicated by circled numbers); to the left number 2; and the outside cell number 3. The cell cards (in two dimensions, all cells void) are

1	0	1	–2	–3	6
2	0	1	–6	–4	5

Cell 1 is a void and is formed by the intersection of the region above (positive sense) surface 1 with the region to the left (negative sense) of surface 2 intersected with the region below (negative sense) surface 3 and finally intersected with the region to the right (positive sense) of surface 6. Cell 2 is described similarly.

Cell 3 cannot be specified with the intersection operator. The following section about the union operator is needed to describe cell 3.

2. Cells Defined by Unions of Regions of Space

The union operator, signified by a colon on the cell cards, allows concave corners in cells and also cells that are completely disjoint. The intersection and union operators are binary Boolean operators, so their use follows Boolean algebra methodology; unions and intersections can be used in combination in any cell description.

Spaces on either side of the union operator are irrelevant, but remember that a space without the colon signifies an intersection. In the hierarchy of operations, intersections are performed first and then unions. There is no left to right ordering. Parentheses can be used to clarify operations and in some cases are required to force a certain order of operations. Innermost parentheses are cleared first. Spaces are optional on either side of a parenthesis. A parenthesis is equivalent to a space and signifies an intersection.

For example, let A and B be two regions of space. The region containing points that belong to both A and B is called the intersection of A and B. The region containing points that belong to A alone or to B alone or to both A and B is called the union of A and B. The lined area in Figure 1.5a represents the union of A and B (or A : B), and the lined area in Figure 1.5b represents the intersection of A and B (or A B). The only way regions of space can be added is with the union operator. An intersection of two spaces always results in a region no larger than either of the two spaces. Conversely, the union of two spaces always results in a region no smaller than either of the two spaces.

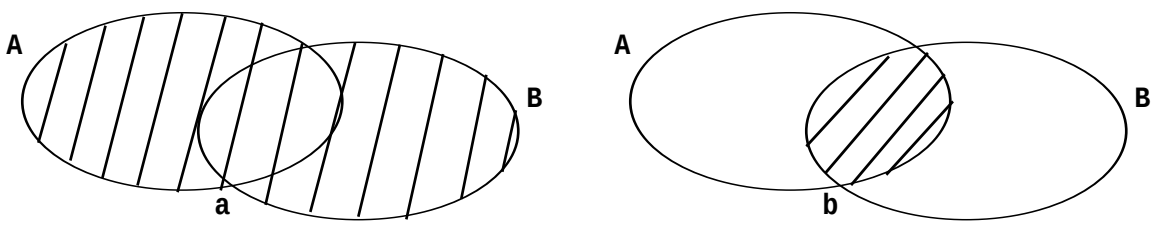

Figure 1-5.

A simple example will further illustrate the concept of Figure 1.5 and the union operator to solidify the concept of adding and intersecting regions of space to define a cell. See also the second example in Chapter 4. In Figure 1.6 we have two infinite planes that meet to form two cells. Cell 1 is easy to define; it is everything in the universe to the right of surface 1 (that is, a positive sense) that is also in common with (or intersected with) everything in the universe below surface 2 (that is, a negative sense). Therefore, the surface relation of cell 1 is 1 –2.

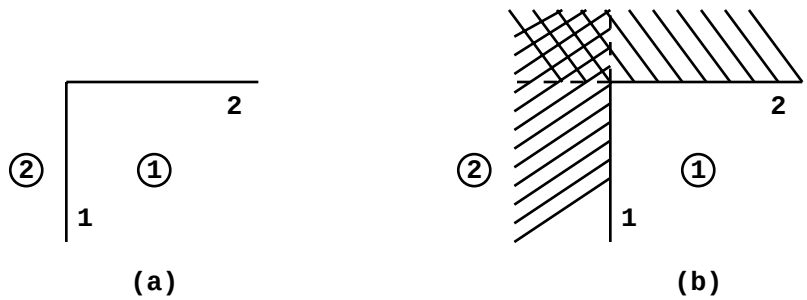

Figure 1-6.

Cell 2 is everything in the universe to the left (negative sense) of surface 1 plus everything in the universe above (positive sense) surface 2, or –1 : 2, illustrated in Figure 1.6b by all the shaded regions of space. If cell 2 were specified as –1 2, that would represent the region of space common to –1 and 2, which is only the cross-hatched region in the figure and is obviously an improper specification for cell 2.

Returning to Figure 1.4 on page 1–16, if cell 1 is inside the solid black line and cell 2 is the entire region outside the solid line, then the MCNP cell cards in two dimensions are (assuming both cells are voids)

```
1   0   1   –2   (–3 : –4)   5
2   0   –5 : –1 : 2 : 3   4
```

Cell 1 is defined as the region above surface 1 intersected with the region to the left of surface 2, intersected with the union of regions below surfaces 3 and 4, and finally intersected with the region to the right of surface 5. Cell 2 contains four concave corners (all but between surfaces 3 and 4), and its specification is just the converse (or complement) of cell 1. Cell 2 is the space defined by the region to the left of surface 5 plus the region below 1 plus the region to the right of 2 plus the space defined by the intersections of the regions above surfaces 3 and 4.

A simple consistency check can be noted with the twocell cards above. All intersections for cell 1 become unions for cell 2 and vice versa. The senses are also reversed.

Note that in this example, all corners less than 180 degrees in a cell are handled by intersections and all corners greater than 180 degrees are handled by unions.

To illustrate some of the concepts about parentheses, assume an intersection is thought of mathematically as multiplication and a union is thought of mathematically as addition. Parentheses are removed first, with multiplication being performed before addition. The cell cards for the example cards above from Figure 1.4 may be written in the form

1	$a \cdot b \cdot (c + d) \cdot e$
2	$e + a + b + c \cdot d$

Note that parentheses are required for the first cell but not for the second, although the second could have been written as $e + a + b + (c \cdot d)$, $(e + a + b) + (c \cdot d)$, $(e) + (a) + (b) + (c \cdot d)$, etc.

Several more examples using the union operator are given in Chapter 4. Study them to get a better understanding of this powerful operator that can greatly simplify geometry setups.

B. Surface Type Specification

The first- and second-degree surfaces plus the fourth-degree elliptical and degenerate tori of analytical geometry are all available in MCNP. The surfaces are designated by mnemonics such as C/Z for a cylinder parallel to the z-axis. A cylinder at an arbitrary orientation is designated by the general quadratic GQ mnemonic. A paraboloid parallel to a coordinate axis is designated by the special quadratic SQ mnemonic. The 29 mnemonics representing various types of surfaces are listed in Table 3.1 on page 3–14.

C. Surface Parameter Specification

There are two ways to specify surface parameters in MCNP: (1) by supplying the appropriate coefficients needed to satisfy the surface equation, and (2) by specifying known geometrical points on a surface that is rotationally symmetric about a coordinate axis.

1. Coefficients for the Surface Equations

The first way to define a surface is to use one of the surface-type mnemonics from Table 3.1 on page 3–14 and to calculate the appropriate coefficients needed to satisfy the surface equation. For example, a sphere of radius 3.62-cm with the center located at the point (4,1,–3) is specified by

```
S   4   1   –3   3.62
```

An ellipsoid whose axes are not parallel to the coordinate axes is defined by the GQ mnemonic plus up to 10 coefficients of the general quadratic equation. Calculating the coefficients can be (and frequently is) nontrivial, but the task is greatly simplified by defining an auxiliary coordinate system whose axes coincide with the axes of the ellipsoid. The ellipsoid is easily defined in terms of the auxiliary coordinate system, and the relationship between the auxiliary coordinate system and the main coordinate system is specified on a TRn card, described on page 3–30.

The use of the SQ (special quadratic) and GQ (general quadratic) surfaces is determined by the orientation of the axes. One should always use the simplest possible surface in describing

geometries; for example, using a GQ surface instead of an S to specify a sphere will require more computational effort for MCNP.

2. Points that Define a Surface

The second way to define a surface is to supply known points on the surface. This method is convenient if you are setting up a geometry from something like a blueprint where you know the coordinates of intersections of surfaces or points on the surfaces. When three or more surfaces intersect at a point, this second method also produces a more nearly perfect point of intersection if the common point is used in the surface specification. It is frequently difficult to get complicated surfaces to meet at one point if the surfaces are specified by the equation coefficients. Failure to achieve such a meeting can result in the unwanted loss of particles.

There are, however, restrictions that must be observed when specifying surfaces by points that do not exist when specifying surfaces by coefficients. Surfaces described by points must be either skew planes or surfaces rotationally symmetric about the x, y, or z axes. They must be unique, real, and continuous. For example, points specified on both sheets of a hyperboloid are not allowed because the surface is not continuous. However, it is valid to specify points that are all on one sheet of the hyperboloid. (See the X,Y,Z, and P input cards description on page 3–16 for additional explanation.)

IV. MCNP INPUT FOR SAMPLE PROBLEM

The main input file for the user is the INP (the default name) file that contains the input information to describe the problem. We will present here only the subset of cards required to run the simple fixed source demonstration problem. All input cards are discussed in Chapter 3 and summarized in Table 3.8 starting on page 3–148.

MCNP does extensive input checking but is not foolproof. A geometry should be checked by looking at several different views with the geometry plotting option. You should also surround the entire geometry with a sphere and flood the geometry with particles from a source with an inward cosine distribution on the spherical surface, using a VOID card to remove all materials specified in the problem. If there are any incorrectly specified places in your geometry, this procedure will usually find them. Make sure the importance of the cell just inside the source sphere is not zero. Then run a short job and study the output to see if you are calculating what you think you are calculating.

The basic constants used in MCNP are printed in optional print table 98 in the output file. The units used are:

1. lengths in centimeters,
2. energies in MeV,
3. times in shakes (10^{-8} sec),

4. temperatures in MeV (*kT*),
5. atomic densities in units of atoms/barn-cm,
6. mass densities in g/cm^3,
7. cross sections in barns (10^{-24} cm^2),
8. heating numbers in MeV/collision, and
9. atomic weight ratio based on a neutron mass of 1.008664967. In these units, Avogadro's number is 0.59703109 x 10^{-24}.

A simple sample problem illustrated in Figure 1.7 is referred to throughout the remainder of this chapter. We wish to start 14-MeV neutrons at a point isotropic source in the center of a small sphere of oxygen that is embedded in a cube of carbon. A small sphere of iron is also embedded in the carbon. The carbon is a cube 10 cm on each side; the spheres have a 0.5-cm radius and are centered between the front and back faces of the cube. We wish to calculate the total and energy-dependent flux in increments of 1 MeV from 1 to 14 MeV, where bin 1 will be the tally from 0 to 1 MeV

1. on the surface of the iron sphere and
2. averaged in the iron sphere volume.

This geometry has four cells, indicated by circled numbers, and eight surfaces—six planes and two spheres. Surface numbers are written next to the appropriate surfaces. Surface 5 comes out from the page in the +x direction and surface 6 goes back into the page in the –x direction.

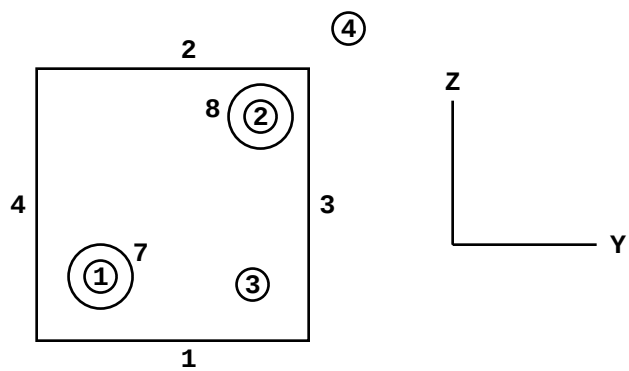

Figure 1-7.

With knowledge of the cell card format, the sense of a surface, and the union and intersection operators, we can set up the cell cards for the geometry of our example problem. To simplify this step, assume the cells are void, for now. Cells 1 and 2 are described by the following cards:

```
1   0   –7
2   0   –8
```

where the negative signs denote the regions inside (negative sense) surfaces 7 and 8. Cell 3 is everything in the universe above surface 1 intersected with everything below surface 2 intersected with everything to the left of surface 3 and so forth for the remaining three surfaces. The region in

common to all six surfaces is the cube, but we need to exclude the two spheres by intersecting everything outside surface 7 and outside surface 8. The card for cell 3 is

```
3   0    1 –2 –3 4 –5 6 7 8
```

Cell 4 requires the use of the union operator and is similar to the idea illustrated in Figure 1.6. Cell 4 is the outside world, has zero importance, and is defined as everything in the universe below surface 1 plus everything above surface 2 plus everything to the right of surface 3 and so forth. The cell card for cell 4 is

```
4   0   –1 : 2 : 3 : –4 : 5 : –6
```

A. INP File

An input file has the following form:

```
Message Block
Blank Line Delimiter      } Optional
One Line Problem Title Card
Cell Cards
      .
      .
Blank Line Delimiter
Surface Cards
      .
      .
Blank Line Delimiter
Data Cards
      .
      .
Blank Line Terminator (optional)
```

All input lines are limited to 80 columns. Alphabetic characters can be upper, lower, or mixed case. A $ (dollar sign) terminates data entry. Anything that follows the $ is interpreted as a comment. Blank lines are used as delimiters and as an optional terminator. Data entries are separated by one or more blanks.

Comment cards can be used anywhere in the INP file after the problem title card and before the optional blank terminator card. Comment lines must have a C somewhere in columns 1-5 followed by at least one blank and can be a total of 80 columns long.

Cell, surface, and data cards must all begin within the first five columns. Entries are separated by one or more blanks. Numbers can be integer or floating point. MCNP makes the appropriate conversion. A data entry item, e.g., IMP:N or 1.1e2, must be completed on one line.

Blanks filling the first five columns indicate a continuation of the data from the last named card. An & (ampersand) ending a line indicates data will continue on the following card, where data on the continuation card can be in columns 1-80.

The optional message block, discussed in detail on page 3–1, is used to change file names and specify running options such as a continuation run. On most systems these options and files may alternatively be specified with an execution line message (see page 1–32). Message block entries supersede execution line entries. The blank line delimiter signals the end of the message block.

The first card in the file after the optional message block is the required problem title card. If there is no message block, this must be the first card in the INP file. It is limited to one 80-column line and is used as a title in various places in the MCNP output. It can contain any information you desire but usually contains information describing the particular problem.

MCNP makes extensive checks of the input file for user errors. A FATAL error occurs if a basic constraint of the input specification is violated, and MCNP will terminate before running any particles. The first fatal error is real; subsequent error messages may or may not be real because of the nature of the first fatal message.

B. Cell Cards

The cell number is the first entry and must begin in the first five columns.

The next entry is the cell material number, which is arbitrarily assigned by the user. The material is described on a material card (Mn) that has the same material number (see page 1–29). If the cell is a void, a zero is entered for the material number. The cell and material numbers cannot exceed 5 digits.

Next is the cell material density. A positive entry is interpreted as atom density in units of 10^{24} atoms/cm^3. A negative entry is interpreted as mass density in units of g/cm^3. No density is entered for a void cell.

A complete specification of the geometry of the cell follows. This specification includes a list of the signed surfaces bounding the cell where the sign denotes the sense of the regions defined by the surfaces. The regions are combined with the Boolean intersection and union operators. A space indicates an intersection and a colon indicates a union.

Optionally, after the geometry description, cell parameters can be entered. The form is keyword=value. The following line illustrates the cell card format:

```
1    1    –0.0014   –7    IMP:N=1
```

Cell 1 contains material 1 with density 0.0014 g/cm^3, is bounded by only one surface (7), and has an importance of 1. If cell 1 were a void, the cell card would be

```
1    0    –7    IMP:N=1
```

The complete cell card input for this problem (with 2 comment cards) is

```
c  cell cards for sample problem
1    1   –0.0014    –7
2    2   –7.86       –8
3    3   –1.60    1  –2  –3  4  –5  6  7  8
4    0              –1:2:3:–4:5:–6
c  end of cell cards for sample problem
blank line delimiter
```

The blank line terminates the cell card section of the INP file. We strongly suggest that the cells be numbered sequentially starting with one. A complete explanation of the cell card input is found in Chapter 3, page 3–9.

C. Surface Cards

The surface number is the first entry. It must begin in columns 1-5 and not exceed 5 digits. The next entry is an alphabetic mnemonic indicating the surface type. Following the surface mnemonic are the numerical coefficients of the equation of the surface in the proper order. This simplified description enables us to proceed with the example problem. For a full description of the surface card see page 3–12.

Our problem uses planes normal to the x, y, and z axes and two general spheres. The respective mnemonics are PX, PY, PZ, and S. Table 1.2 shows the equations that determine the sense of the surface for the cell cards and the entries required for the surface cards. A complete list of available surface equations is contained in Table 3.1 on page 3–14.

TABLE 1.2:
Surface Equations

Mnemonic	Equation	Card Entries
PX	x - D = 0	D
PY	y - D = 0	D
PZ	x - D = 0	D
S	$(x-\bar{x})^2 + (x-\bar{y})^2 + (z-\bar{z})^2 - R^2 = 0$	$\bar{x}\bar{y}\bar{z}R$

For the planes, D is the point where the plane intersects the axis. If we place the origin in the center of the 10-cm cube shown in Figure 1.7, the planes will be at x = –5, x = 5, etc. The two spheres are not centered at the origin or on an axis, so we must give the x,y,z of their center as well as their radii. The complete surface card input for this problem is shown below. A blank line terminates the surface card portion of the input.

```
C   Beginning of surfaces for cube
     1   PZ      –5
     2   PZ       5
     3   PY       5
     4   PY      –5
     5   PX       5
     6   PX      –5
C   End of cube surfaces
     7    S  0    -4    -2.5     .5  $ oxygen sphere
     8    S  0     4     4       .5  $ iron sphere
     blank line delimiter
```

D. Data Cards

The remaining data input for MCNP follows the second blank card delimiter, or third blank card if there is a message block. The card name is the first entry and must begin in the first five columns. The required entries follow, separated by one or more blanks.

Several of the data cards require a particle designator to distinguish between input data for neutrons, data for photons, and data for electrons. The particle designator consists of the symbol : (colon) and the letter N or P or E immediately following the name of the card. For example, to enter neutron importances, use an IMP:N card; enter photon importances on an IMP:P card; enter electron importances on an IMP:E card. No data card can be used more than once with the same mnemonic, that is, M1 and M2 are acceptable, but two M1 cards are not allowed. Defaults have been set for cards in some categories. A summary starting on page 3–147 shows which cards are required, which are optional, and whether defaults exist and if so, what they are. The sample problem will use cards in the following categories:

	MCNP card name
1. mode,	MODE
2. cell and surface parameters,	IMP:N
3. source specification,	SDEF
4. tally specification,	Fn, En
5. material specification, and	Mn
6. problem cutoffs.	NPS

A complete description of the data cards is found on page 3–22 in Chapter 3.

1. MODE Card

MCNP can be run in several different modes:

Mode	N	— neutron transport only (default)
	N P	— neutron and neutron-induced photon transport
	P	— photon transport only
	E	— electron transport only
	P E	— photon and electron transport
	N P E	— neutron, neutron-induced photon and electron transport

The MODE card consists of the mnemonic MODE followed by the choices shown above. If the MODE card is omitted, mode N is assumed.

Mode N P does not account for photo-neutrons but only neutron-induced photons. Photon-production cross sections do not exist for all nuclides. If they are not available for a Mode N P problem, MCNP will print out warning messages. To find out whether a particular table for a nuclide has photon-production cross sections available, check the Appendix G cross-section list.

Mode P or mode N P problems generate bremsstrahlung photons with a computationally expensive thick-target bremsstrahlung approximation. This approximation can be turned off with the PHYS:E card.

The sample problem is a neutron-only problem, so the MODE card can be omitted because MODE N is the default.

2. Cell and Surface Parameter Cards

Most of these cards define values of cell parameters. Entries correspond in order to the cell or surface cards that appear earlier in the INP file. A listing of all available cell and surface parameter cards is found on page 3–32. A few examples are neutron and photon importance cards (IMP:N,IMP:P), weight window cards (WWE:N, WWE:P, WWNi:N, WWNi:P), etc. Some method of specifying relative cell importances is required; the majority of the other cell parameter cards are for optional variance reduction techniques. The number of entries on a cell or surface parameter card must equal the number of cells or surfaces in the problem or MCNP prints out a WARNING or FATAL error message. In the case of a WARNING, MCNP assumes zeros.

The IMP:N card is used to specify relative cell importances in the sample problem. There are four cells in the problem, so the IMP:N card will have four entries. The IMP:N card is used (a) for terminating the particle's history if the importance is zero and (b) for geometry splitting and

Russian roulette to help particles move more easily to important regions of the geometry. An IMP:N card for the sample problem is

```
IMP:N    1  1  1  0
```

Cell parameters also can be defined on cell cards using the keyword=value format. If a cell parameter is specified on **any** cell card, it must be specified **only** on cell cards and **not at all** in the data card section.

3. Source Specification Cards

A source definition card SDEF is one of four available methods of defining starting particles. Chapter 3 has a complete discussion of source specification. The SDEF card defines the basic source parameters, some of which are

POS = x y z	default is 0 0 0;
CEL = starting cell number	
ERG = starting energy	default is 14 MeV;
WGT = starting weight	default is 1;
TME = time	default is 0;
PAR = source particle type	1 for N, N P, N P E; 2 for P, P E; 3 for E.

MCNP will determine the starting cell number for a point isotropic source, so the CEL entry is not always required. The default starting direction for source particles is isotropic.

For the example problem, a fully specified source card is

```
SDEF    POS = 0 –4 –2.5    CEL = 1    ERG = 14    WGT = 1    TME = 0    PAR = 1
```

Neutron particles will start at the center of the oxygen sphere (0 –4 –2.5), in cell 1, with an energy of 14 MeV, and with weight 1 at time 0. All these source parameters except the starting position are the default values, so the most concise source card is

```
SDEF    POS = 0 –4 –2.5
```

If all the default conditions applied to the problem, only the mnemonic SDEF would be required.

4. Tally Specification Cards

The tally cards are used to specify what you want to learn from the Monte Carlo calculation, perhaps current across a surface, flux at a point, etc. You request this information with one or more tally cards. Tally specification cards are not required, but if none is supplied, no tallies will be

printed when the problem is run and a warning message is issued. Many of the tally specification cards describe tally "bins." A few examples are energy (En), time (Tn), and cosine (Cn) cards.

MCNP provides six standard neutron, six standard photon, and four standard electron tallies, all normalized to be per starting particle. Some tallies in criticality calculations are normalized differently. Chapter 2, page 2–76, discusses tallies more completely, and Chapter 3, page 3–73, lists all the tally cards and fully describes each one.

	Tally Mnemonic		Description
F1:N	or F1:P	or F1:E	Surface current
F2:N	or F2:P	or F2:E	Surface flux
F4:N	or F4:P	or F4:E	Track length estimate of cell flux
F5a:N	or F5a:P		Flux at a point (point detector)
F6:N	or F6:N,P		Track length estimate of energy deposition
	or F6:P		
F7:N			Track length estimate of fission energy deposition
	F8:P	or F8:E	Energy distribution of pulses created
	or F8:P,E		in a detector

The tallies are identified by tally type and particle type. Tallies are given the numbers 1, 2, 4, 5, 6, 7, 8, or increments of 10 thereof, and are given the particle designator :N or :P or :E (or :N,P only in the case of tally type 6 or P,E only for tally type 8). Thus you may have as many of any basic tally as you need, each with different energy bins or flagging or anything else. F4:N, F14:N, F104:N, and F234:N are all legitimate neutron cell flux tallies; they could all be for the same cell(s) but with different energy or multiplier bins, for example. Similarly F5:P, F15:P, and F305:P are all photon point detector tallies. Having both an F1:N card and an F1:P card in the same INP file is not allowed. The tally number may not exceed three digits.

For our sample problem we will use Fn cards (Tally type) and En cards (Tally energy).

a. Tally (Fn) Cards: The sample problem has a surface flux tally and a track length cell flux tally. Thus, the tally cards for the sample problem shown in Figure 1.7 are

```
F2:N    8    $   flux across surface 8
F4:N    2    $   track length in cell 2
```

Printed out with each tally bin is the relative error of the tally corresponding to one estimated standard deviation. Read page 1–6 for an explanation of the relative error. Results are not reliable until they become stable as a function of the number of histories run. Much information is provided for one bin of each tally in the tally fluctuation charts at the end of the output file to help determine tally stability. The user is strongly encouraged to look at this information carefully.

b. Tally Energy (En) Card: We wish to calculate flux in increments of 1 MeV from 14 to 1 MeV. Another tally specification card in the sample input deck establishes these energy bins.

The entries on the En card are the upper bounds in MeV of the energy bins for tally n. The entries must be given in order of increasing magnitude. If a particle has an energy greater than the last entry, it will not be tallied, and a warning is issued. MCNP automatically provides the total over all specified energy bins unless inhibited by putting the symbol NT as the last entry on the selected En card.

The following cards will create energy bins for the sample problem:

```
E2     1  2  3  4  5  6  7  8  9  10  11  12  13  14
E4     1  12I  14
```

If no En card exists for tally n, a single bin over all energy will be used. To change this default, an E0 (zero) card can be used to set up a default energy bin structure for all tallies. A specific En card will override the default structure for tally n. We could replace the E2 and E4 cards with one E0 card for the sample problem, thus setting up identical bins for both tallies.

5. Materials Specification

The cards in this section specify both the isotopic composition of the materials and the cross-section evaluations to be used in the cells. For a comprehensive discussion of materials specification, see page 3–108.

a. Material (Mm) Card: The following card is used to specify a material for all cells containing material m, where m cannot exceed 5 digits:

Mm $ZAID_1$ $fraction_1$ $ZAID_2$ $fraction_2$ ...

The m on a material card corresponds to the material number on the cell card (see page 1–23). The consecutive pairs of entries on the material card consist of the identification number (ZAID) of the constituent element or nuclide followed by the atomic fraction (or weight fraction if entered as a negative number) of that element or nuclide, until all the elements and nuclides needed to define the material have been listed.

i. Nuclide Identification Number (ZAID). This number is used to identify the element or nuclide desired. The form of the number is ZZZAAA.nnX, where

- ZZZ is the atomic number of the element or nuclide,
- AAA is the mass number of the nuclide, ignored for photons and electrons,
- nn is the cross-section evaluation identifier; if blank or zero, a default cross-section evaluation will be used, and

X is the class of data: C is continuous energy; D is discrete reaction; T is thermal; Y is dosimetry; P is photon; E is electron; and M is multigroup.

For naturally occurring elements, AAA=000. Thus ZAID=74182 represents the isotope $^{182}_{74}W$, and ZAID=74000 represents the element tungsten.

ii. Nuclide Fraction. The nuclide fractions may be normalized to 1 or left unnormalized. For example, if the material is H_2O, the fractions can be entered as .667 and .333, or as 2 and 1 for H and O respectively. If the fractions are entered with negative signs, they are weight fractions; otherwise they are atomic fractions. Weight fractions and atomic fractions cannot be mixed on the same Mm card.

The material cards for the sample problem are

```
M1      8016    1    $ oxygen 16
M2      26000   1    $ natural iron
M3      6000    1    $ carbon
```

b. VOID Card: The VOID card removes all materials and cross sections in a problem and sets all nonzero importances to unity. It is very effective for finding errors in the geometry description because many particles can be run in a short time. Flooding the geometry with many particles increases the chance of particles going to most parts of the geometry—in particular, to an incorrectly specified part of the geometry—and getting lost. The history of a lost particle often helps locate the geometry error. The other actions of and uses for the VOID card are discussed on page 3–113.

The sample input deck could have a VOID card while testing the geometry for errors. When you are satisfied that the geometry is error-free, remove the VOID card.

6. Problem Cutoffs

Problem cutoff cards are used to specify parameters for some of the ways to terminate execution of MCNP. The full list of available cards and a complete discussion of problem cutoffs is found on page 3–124. For our problem we will use only the history cutoff (NPS) card. The mnemonic NPS is followed by a single entry that specifies the number of histories to transport. MCNP will terminate after NPS histories unless it has terminated earlier for some other reason.

7. Sample Problem Summary

The entire input deck for the sample problem follows. Recall that the input can be upper, lower, or mixed case.

```
Sample Problem Input Deck
c      cell cards for sample problem
1      1 -0.0014   -7
2      2 -7.86     -8
3      3 -1.60      1 -2 -3 4 -5 6 7 8
4      0           -1:2:3:-4:5:-6
c      end of cell cards for sample problem
C      Beginning of surfaces for cube
1      PZ  -5
2      PZ   5
3      PY   5
4      PY  -5
5      PX   5
6      PX  -5
C      End of cube surfaces
7      S  0  -4  -2.5  .5         $  oxygen sphere
8      S  0   4   4.5             $  iron sphere
blank line delimiter
IMP:N  1  1  1  0
SDEF POS=0 -4 -2.5
F2:N   8                          $  flux across surface 8
F4:N   2                          $  track length in cell 2
E0     1  12I  14
M1     8016   1                   $  oxygen 16
M2     26000  1                   $  natural iron
M3     6000   1                   $  carbon
NPS    100000
blank line delimiter   (optional)
```

V. HOW TO RUN MCNP

This section assumes a basic knowledge of UNIX. Lines the user will type are shown in **`lower case typewriter style`** type. Press the RETURN key after each input line. MCNP is the executable binary file and XSDIR is the cross-section directory. If XSDIR is not in your current directory, you may need to set the environmental variable:

```
setenv DATAPATH /ab/cd
```

where `/ab/cd` is the directory containing both XSDIR and the data libraries.

A. Execution Line

The MCNP execution line has the following form:

`mcnp` *`Files Options`*

`Files` and *`Options`* are described below. Their order on the execution line is irrelevant. If there are no changes in default file names, nothing need be entered for *`Files`* and *`Options`*.

1. Files

MCNP uses several files for input and output. The file names cannot be longer than eight characters. The files pertinent to the sample problem are shown in Table 1.3. File INP must be present as a local file. MCNP will create OUTP and RUNTPE.

TABLE 1.3:
MCNP Files

Default File Name	Description
INP	Problem input specification
OUTP	BCD output for printing
RUNTPE	Binary start-restart data
XSDIR	Cross-section directory

The default name of any of the files in Table 1.3 can be changed on the MCNP execution line by entering

```
default_file_name=newname
```

For example, if you have an input file called MCIN and want the output file to be MCOUT and the runtpe to be MCRUNTPE, the execution line is

```
mcnp inp=mcin outp=mcout runtpe=mcruntpe
```

Only enough letters of the default name are required to uniquely identify it. For example,

```
mcnp i=mcin o=mcout ru=mcrntpe
```

also works. If a file in your local file space has the same name as a file MCNP needs to create, the file is created with a different unique name by changing the last letter of the name of the new file

to the next letter in the alphabet. For example, if you already have an OUTP, MCNP will create OUTQ.

Sometimes it is useful for all files from one run to have similar names. If your input file is called JOB1, the following line

```
mcnp name=job1
```

will create an OUTP file called JOB1O and a RUNTPE file called JOB1R. If these files already exist, MCNP will NOT overwrite them, but will issue a message that JOB1O already exists and then will terminate.

2. Options

There are two kinds of options: program module execution options and other options. Execution options are discussed next.

MCNP consists of five distinct execution operations, each given a module name. These operations, their corresponding module names, and a one-letter mnemonic for each operation are listed in Table 1.4.

TABLE 1.4:
Execution Options

Mnemonic	Module	Operation
i	IMCN	Process problem input file
p	PLOT	Plot geometry
x	XACT	Process cross sections
r	MCRUN	Particle transport
z	MCPLOT	Plot tally results or cross section data

When *Options* are omitted, the default is **ixr**. The execution of the modules is controlled by entering the proper mnemonic on the execution line. If more than one operation is desired, combine the single characters (in any order) to form a string. Examples of use are as follows: **i** to look for input errors, **ip** to debug a geometry by plotting, **ixz** to plot cross-section data, and **z** to plot tally results from the RUNTPE file.

After a job has been run, the BCD print file OUTP can be examined with an editor on the computer and/or sent to a printer. Numerous messages about the problem execution and statistical quality of the results are displayed at the terminal.

The "other" options add more flexibility when running MCNP and are shown in Table 1.5.

TABLE 1.5:
Other Options

Mnemonic	Operation
C m	Continue a run starting with m^{th} dump. If m is omitted, last dump is used. See page 3–2
CN	Like C, but dumps are written immediately after the fixed part of the RUNTPE, rather than at the end. See page 3–2
DBUG n	Write debug information every n particles. See DBCN card, page 3–130
NOTEK	Indicates that your terminal has no graphics capability. PLOT output is in PLOTM.PS. Equivalent to TERM=0. See
FATAL	Transport particles and calculate volumes even if fatal errors are found.
PRINT	Create the full output file; equivalent to PRINT card. See page 3–134
TASKS n	Invokes multiprocessing on common or distributed memory systems. n=number of processors to be used. –n is allowed only on distributed memory systems to disable load balancing and fault tolerance, increasing system efficiency.

The TASK option must be used to invoke multiprocessing on common or distributed memory computer systems and is followed by the number of tasks or CPUs to be used for particle tracking. The multiprocessing capability must be invoked at the time of compilation to create a compatible executable. Two compilation options exist: common memory systems (UNICOS, etc.) and distributed memory systems (workstation clusters, Cray T3D, etc.) While multiprocessing on common memory systems is invoked and handled by the compiler with compiler directives, on distributed memory systems it is performed by the software communications package Parallel Virtual Machine[9] (PVM). Thus, using this capability on distributed memory systems requires the installation and execution of PVM.[10] On such systems, a negative entry following the TASKS option will maximize efficiency for homogeneous dedicated systems (e.g., workstation with multiple CPUs). For heterogeneous or multiuser systems, a positive entry should be used, in which case load balancing and fault tolerance are enabled.[11] In either case, the absolute value of this entry indicates the number of hosts (or CPUs) available for use during particle tracking. On both common and distributed memory systems, a table is provided in the output file that lists the number of particles tracked by each host.

```
mcnp i=input o=output tasks 8
```

Indicates eight processors are to be used for particle tracking. On a common memory system, eight tasks are initiated (if fewer processors are actually available, multiple tasks are run on each processor.) On a distributed memory system, the master task and one subtask are initiated on the primary host (i.e., machine from which the execution is initiated), and a subtask is initiated on each of the seven secondary hosts.

```
mcnp name=inp tasks -4
```

A negative entry following the TASKS option is allowed only on a distributed memory system and is recommended for homogeneous dedicated systems. As in the previous example, the master task and one subtask are initiated on the primary host, and a subtask is initiated on each of the three secondary hosts. The negative entry disables load balancing and fault tolerance, increasing system efficiency.

B. Interrupts

MCNP allows four interactive interrupts while it is running:

`(ctrl c)<cr>` (default)	MCNP status
`(ctrl c)s`	MCNP status
`(ctrl c)m`	Make interactive plots of tallies
`(ctrl c)q`	Terminate MCNP normally after current history
`(ctrl c)k`	Kill MCNP immediately

The (ctrl c)s interrupt prints the computer time used so far, the number of particles run so far, and the number of collisions. In the IMCN module, it prints the input line being processed. In the XACT module, it prints the cross section being processed.

The (ctrl c)q interrupt has no effect until MCRUN is executed. (Ctrl c)q causes the code to stop after the current particle history, to terminate “gracefully,” and to produce a final print output file and RUNTPE file.

The (ctrl c)k interrupt kills MCNP immediately, without normal termination. If (ctrl c)k fails, enter (ctrl c) three or more times in a row.

C. Running MCNP

To run the example problem, have the input file in your current directory. For illustration, assume the file is called SAMPLE. Type

```
mcnp   n=sample
```

where n uniquely identifies NAME. MCNP will produce an output file SAMPLEO that you can examine at your terminal, send to a printer, or both. To look at the geometry with the PLOT module using an interactive graphics terminal, type in

```
mcnp   ip   n=sample
```

After the plot prompt *plot* > appears, type in

```
px=0   ex=20
```

This plot will show an intersection of the surfaces of the problem by the plane X = 0 with an extent in the x-direction of 20 cm on either side of the origin. If you want to do more with PLOT, see the instructions on page B-1. Otherwise type “end” after the next prompt to terminate the session.

VI. TIPS FOR CORRECT AND EFFICIENT PROBLEMS

This section has a brief checklist of helpful hints that apply to three phases of your calculation: defining and setting up the problem, preparing for the long computer runs that you may require, and making the runs that will give you results. Not everything mentioned in the checklist has been covered in this chapter, but the list can serve as a springboard for further reading in preparation for tackling more difficult problems.

A. Problem Setup

1. Model the geometry and source distribution accurately.
2. Use the best problem cutoffs.
3. Use zero (default) for the neutron energy cutoff (MODE N P).
4. Do not use too many variance reduction techniques.
5. Use the most conservative variance reduction techniques.
6. Do not use cells with many mean free paths.
7. Use simple cells.
8. Use the simplest surfaces.
9. Study warning messages.
10. Always plot the geometry.
11. Use the VOID card when checking geometry.
12. Use separate tallies for the fluctuation chart.
13. Generate the best output (consider PRINT card).
14. RECHECK the INP file (materials, densities, masses, sources, etc.).
15. GARBAGE into code = GARBAGE out of code.

B. Preproduction

1. Run some short jobs.
2. Examine the outputs carefully.
3. Study the summary tables.
4. Study the statistical checks on tally quality and the sources of variance.
5. Compare the figures of merit and variance of the variance.

6. Consider the collisions per source particle.
7. Examine the track populations by cell.
8. Scan the mean free path column.
9. Check detector diagnostic tables.
10. Understand large detector contributions.
11. Strive to eliminate unimportant tracks.
12. Check MODE N P photon production.
13. Do a back-of-the-envelope check of the results.
14. DO NOT USE MCNP AS A BLACK BOX.

C. Production

1. Save RUNTPE for expanded output printing, continue run, tally plotting.
2. Look at figure of merit stability.
3. Make sure answers seem reasonable.
4. Make continue runs if necessary.
5. See if stable errors decrease by $1/\sqrt{N}$ (that is, be careful of the brute force approach).
6. Remember, accuracy is only as good as the nuclear data, modeling, MCNP sampling approximations, etc.

VII. REFERENCES

1. R. Kinsey, "Data Formats and Procedures for the Evaluated Nuclear Data File, ENDF," Brookhaven National Laboratory report BNL-NCS-50496 (ENDF 102) 2nd Edition (ENDF/B-V) (October 1979).
2. R. J. Howerton, D. E. Cullen, R. C. Haight, M. H. MacGregor, S. T. Perkins, and E. F. Plechaty, "The LLL Evaluated Nuclear Data Library (ENDL): Evaluation Techniques, Reaction Index, and Descriptions of Individual Reactions," Lawrence Livermore National Laboratory report UCRL-50400, Vol. 15, Part A (September 1975).
3. M. A. Gardner and R. J. Howerton, "ACTL: Evaluated Neutron Activation Cross–Section Library-Evaluation Techniques and Reaction Index," Lawrence Livermore National Laboratory report UCRL-50400, Vol. 18 (October 1978).
4. E. D. Arthur and P. G. Young, "Evaluated Neutron-Induced Cross Sections for ^{54,56}Fe to 40 MeV," Los Alamos Scientific Laboratory report LA-8626-MS (ENDF-304) (December 1980).
5. D. G. Foster, Jr. and E. D. Arthur, "Average Neutronic Properties of "Prompt" Fission Products," Los Alamos National Laboratory report LA-9168-MS (February 1982).
6. E. D. Arthur, P. G. Young, A. B. Smith, and C. A. Philis, "New Tungsten Isotope Evaluations for Neutron Energies Between 0.1 and 20 MeV," *Trans. Am. Nucl. Soc.* 39, 793 (1981).
7. R. E. MacFarlane, D. W. Muir, and R. M. Boicourt, "The NJOY Nuclear Data Processing System, Volume I: User's Manual," Los Alamos National Laboratory report LA-9303-M, Vol. I (ENDF-324) (May 1982).
   R. E. MacFarlane, D. W. Muir, and R. M. Boicourt, "The NJOY Nuclear Data Processing System, Volume II: The NJOY, RECONR, BROADR, HEATR, and THERMR Modules," Los Alamos National Laboratory report LA-9303-M, Vol. II (ENDF-324) (May 1982).
8. R. A. Forster, R. C. Little, J. F. Briesmeister, and J. S. Hendricks, "MCNP Capabilities For Nuclear Well Logging Calculations," *IEEE Transactions on Nuclear Science*, **37** (3), 1378 (June 1990)
9. A. Geist et al, "PVM 3 User's Guide and Reference Manual," ORNL/TM-12187, Oak Ridge National Laboratory (1993).
10. G. McKinney, "A Practical Guide to Using MCNP with PVM," *Trans. Am. Nucl. Soc.* 71, 397 (1994).
11. G. McKinney, "MCNP4B Multiprocessing Enhancements Using PVM," LANL memo X-6:GWM-95-212 (1995).