MCNP for Engineers

A walkthrough on how to use it, get results,
and
What it all means to a fulfilling life
Part I - Super simple I promise!

Prof. R. A. Borrelli

University of Idaho · Idaho Falls Center for Higher Education

Nuclear Engineering and Industrial Management Department

rborrelli@uidaho.edu

@TheDoctorRAB

2023.08.10

1 Preface

1.1 Who is this walkthrough for?

Advanced undergraduate students and graduate students in any nuclear engineering curriculum. Students should know -
- Basic nuclear physics; e.g., cross sections
- Interactions of neutrons and photons with matter
- Shielding, dose rates
- Four/six factor formula; e.g., what k_{EFF} is,
- How a nuclear reactor works
- Solving for buckling
- Neutron diffusion

Just about every nuclear engineering department has this course. Typically, the Lamarsh or Duderstadt textbooks are used mostly. Sometimes the Shultis textbook is used. It's usually one of the first classes taken prior to the higher-level nuclear engineering courses. However, this course provides all you need to know to run and understand MCNP. I happen to teach this course. I use Lamarsh with Shultis as a reference. I happened to use Lamarsh because that was the textbook when I first took this kind of course. There really is no argument for one over the other.

1.2 What is MCNP?

Your best friend. Your greatest nemesis. MCNP is a contradiction. It will make you suffer, but it will open doors and present new opportunities.

MCNP is a computational tool - that means you're not coding *per se*. You set up the input file that the code will read and then execute. MCNP tracks neutrons and photons for specified geometries and produces a wealth of resulting data. That seems simple. It's not. With a little guidance, effective modeling with MCNP is achievable.

1.3 Why is MCNP so important?

It's not that necessarily MCNP itself is so important. Neutronics modeling is. We can't design any reactor without knowing where the neutrons are going and what they're going to do when they get there. MCNP happens to be the first neutronics computational tool. (They literally used punch cards.) All other tools are benchmarked against MCNP.

Not everyone is going to be a neutronics expert. For those that want to be, mastering MCNP makes it far easier to learn other neutronics codes, like Serpent. Any facility with MCNP provides a fundamental basis for a career in nuclear engineering. Frankly, if you're a graduate student, you're looking for internships. You may not really care what you're going to do; you just want a good position for your CV and a chance to network for future career development. Absolutely nothing wrong with that. On more than one occasion, I have had a researcher come up to me and ask 'Do you know any of the students that know MCNP? I have some money for an intern this summer, but I need someone to step in and get right going.' I intend for this walkthrough to give you the skills to step right in and get going.

Learning MCNP will also lend to transferable skills. Whether it is good coding practices, geometric modeling, or just developing engineering judgement. This will lead to success in higher endeavors.

2 Motivation - Do we *really* need another 'How to use MCNP'?

2.1 Maybe?

Don't read this if you don't want to. I'm not losing sleep over it. Due to the virus sweeping the nation in 2020, over the summer, I decided I needed to take the time to prepare my fall course with the contingency for shifting to online delivery. I've taught the course since 2015, so I know the material, have all the slides and assignments, etc., already prepared. This is part of that effort. Students should be able to follow this on their own and learn how to use MCNP.

2.2 An embarrassment of riches

There are tons and tons and tons and tons of MCNP resources floating around out there. I have compiled them as I find them in my online educational resource repository for nuclear engineering. However, all this decentralization just spreads the materials out so much that there isn't really an orderly way to work through them as a learning process. There are a lot of good problems out there that I use in class.

Probably the closest resource to what I'm trying to do here is the famous MCNP primer from Prof. Shultis at Kansas State University. A particularly good feature of the primer is that it cross references to the MCNP manual. I certainly used it when I first learned MCNP. However, I do not know if that manual is shipped with MCNP anymore, and the most recent revision of the primer that I have found is 2011. To be fair, not too much has changed in the ensuing time, but just about all of the resources in addition to the primer that I have found are somewhat dated. The time seems right to develop something new. Another difficulty I have is that most of these resources are gigantic information dumps. It was overwhelming for me when I was first learning MCNP. I like writing, I'm fairly good at it, and given the context of 2020, the time felt right to give this a shot.

3 Experience - And who do you think you are?

Well, I'm an actual nuclear engineering professor. I've taught the Lamarsh class with an MCNP learning module every year since 2015. Students have attained solid success with publications, internships, and gainful employment from the course. I have many publications applying MCNP in a variety of topics. My expertise in MCNP has led to several funded projects. Ask about me.

I am by no means the leading expert in MCNP. I'm probably not a leading anything unless I'm leading you to happy hour. I can name probably 12 - 15 people I know personally that are better at neutronics modeling than me, and probably 3 to 5 of them are students that took my course. However, actually *teaching* MCNP formally is a different paradigm, altogether. I am confident that I'm on a short list of being able to teach *how to use MCNP effectively*. I am purposely very specific in claiming what I can actually do here.

4 Style - What's wrong with you?

Plenty, but now's not that time for that.

As you may have gathered so far, my tone is intended to be conversational and not overly ponderous. So, not like a journal paper. I'm writing in the way I have been lecturing on MCNP. I'm trying to ease you through the learning process and not just dump everything on to the page. I'm not trying to talk down to anyone. I know the difficulties I had with learning MCNP, so I'm writing in a way that I would have wanted to be spoken to. Even that's a bad sentence, but this isn't meant to be a formally reviewed and published document.

5 Teaching MCNP for *engineers*

There is a reason why I chose the title that I did. I'm interested in getting students and other researchers the guidance they need to run MCNP. If I were going to prepare a course, then, sure, I would spend a significant time on Monte Carlo theory. And I'll have a brief section in here about that. I'm not arguing that a scientific approach isn't valuable in comparison to an engineering approach. I have found most MCNP resources include a ton of material on the theory behind MCNP. I'm assuming you know that already. If you feel like you need some background on the theory behind MCNP, I direct you to Fundamentals of Monte Carlo particle transport by Dr. Forrest B. Brown at Los Alamos National Laboratory. If there's only one expert in MCNP and Monte Carlo theory, he is probably that person.

6 *Walkthrough*

I also chose the term '*walkthrough*' deliberately. Here's where I think I bring some value versus other resources. When I think about a walkthrough, I'm thinking about video games - You're playing a cool game but for the last hour you're stuck in a room looking for a lever or loose tile or you're asking questions to the townspeople and you know you have to do it in a certain order but you're going in circles. The game is starting to get tedious and you need a hint. The walkthrough provides a detailed account of the game - right down to where to look or, not only what weapon to select, but what move to employ. That's what I'm going for here - 'Fatal error? Look up. Unlatch the window.' MCNP is slightly more complex than a video game, but if you can build an input file from scratch and get it to run, then you're well on your way. Building in more complexity isn't as overwhelming.

7 Installation

Detailed installation instructions are provided with your MCNP disks. I am not going to repeat them here, but you should follow them to the letter. Run the tests after to make sure everything is all set. Remember, your license only allows you to install MCNP in one place due to export control restrictions. Mine is on my office desktop. I do not recommend installing it on a laptop. If it gets stolen, you could get into some trouble. Your license is granted through your institution. If you change positions and go somewhere else, then you need to apply for another license and you have to uninstall the distribution under the prior license and destroy the disks.

8 Operating system

I strongly recommend installing in linux. Over the years, I've experienced either first hand or with students, inexplicable wonkiness with MCNP in a Windows environment. Don't have a linux system? No problem! You can install a 'virtual linux machine' on your windows computer. It takes some time, but it's not hard. I have MCNP installed on the virtual machine in my office. I am using the 'Oracle VM VirtualBox Manager'. That is basically the holder for your linux environment. I use the CentOS8 linux system. There's others to choose from. If you have the computing power, you can have the box make two screens or more if you have a multiple monitors. I actually use the linux box about 99% of the time for MCNP, data processing, lecture slides, presentations, papers, all of it.

9 Visualization

The linux distribution for MCNP comes with a plotter equipped to view your models. There's nothing extra to do in a standard linux environment. MCNP used to ship VisEd. I started on VisEd, so I am more used to it than the plotter. Both aren't super great. VisEd only runs in windows. You can request a VisEd download separately after you obtain your MCNP license. I would recommend it. Try both. It's kind of a pain to copy your MCNP file from linux to windows, but it's not really that bad.

10 Manual

The MCNP manual is very comprehensive. That's also kind of the problem. When I was learning MCNP on my own, the manual was just overwhemling. It is important to have even if for showing what each card (essentially, a card is a command) default is, or what output you want to write. Additionally, it contains several tables for the cross section libraries that you can designate. The 'standard' manual which I think everyone refers to when they're talking about the manual is the one with the *LA-UR-03-1987* designation. Mine was last revised in 2008. I don't know if this is shipped or not with the most current MCNP distribution, but I would recommend to find it. I'm not going to be referring to the manual. The Primer actually does a good job with that. I'm intending this walkthrough to be standalone, where the user can then access more comprehensive resources and not feel deluged with information.

11 Text editor

MCNP is a FORTRAN code. *FORTRAN*. That's an old language. O. L. D. That means you have to be really careful in your selection of a text editor. FORTRAN will go sideways fast if there's any whitespace or '$\wedge M$' in there. It is also very column specific - where to start and end a line. This is because when they first invented the language, they were programming on punch cards. Fortunately, there are plenty of choices in linux that can be used to build your input files. I use `vim`. I started on `vi` at an internship in undergrad at the Naval Undersea Warfare Center because the particular group I was working in used

that. I actually learned FORTRAN there. The other popular editor is `emacs`. In linux, there are many choices. Use one of these. Make sure to turn on line numbers!

12 Scientific computing best practices

12.1 Python

As a nuclear engineering graduate student, faculty, or other professional, I know you all have a copy of Effective Computation in Physics: Field Guide to Research with Python. This book, and I feel like using the term 'book' to describe this treatise to a successful research career just falls short, should be your guide to best computing practices. Again, installing MCNP in linux allows for more ready postprocessing with the accessibility to python. You can install python in your windows environment, but I just find installing packages to be a lot easier in linux.

12.2 Postprocessing

For more complex neutronics and/or photon modeling, effective postprocessing is a necessity. Fortunately, with python, you can find many scripts to edit for your own purposes. I have several on my github. Please feel free to look around my repositories. Additionally, I've found it easier to prepare graphs in python. I have several python graph scripts as well. MCNP comes with plotting capabilities. I never got around to learning how to use them yet. That doesn't mean they aren't useful. I just haven't had the time. You can also explore the PyNE toolkit.

12.3 Reproducibility

Good computing means your creation can be used by other people. If you make an MCNP file, someone else should be able to run it and produce the same results that you obtained. This provides confidence that you produced research that is scientifically sound. Then, whomever has your MCNP file can modify, enhance, or build upon the model to tackle more problems. We must have this principle in mind when developing MCNP files. Much of that just involves comprehensive commenting. As I walkthrough MCNP, I'll make note of this.

13 A brief discussion on Monte Carlo theory

The theory behind MCNP is fairly complicated. It would take a semester course to really get into it in depth. I don't really teach much of it at all, just enough so a user knows how MCNP works. Dr. Brown's 400+ slides contain an exhaustive amount of information on Monte Carlo theory, and I have used them for my own lectures in the Lamarsh course. I'm not trying to be glib - it's important to know the theory. However, this is a walkthrough for engineers on how to use MCNP. The theory can be discovered elsewhere.

In short - Monte Carlo in general is a mathematical approach that simulates particle transport using lots of random number generators and statistical distributions. I'm sure the savvy graduate student will quickly understand that the science of random numbers is of critical importance. True randomness can't ever be achieved, but you need your simulation to be as high fidelity as possible. Dr. Brown's slides contain a very in-depth discussion about random number generation in MCNP. It's very good. Both the slides and the generation of randomness. I'm not a benchmark expert, but I think this is one of the reasons everything is benchmarked with MCNP. The user can actually change the kernels for random number generation, but I have never done that.

13.1 MCNP solves the transport equation

Monte Carlo methods solve integral problems, and it is very applicable to multi-dimensional integration. This is great for us because we're concerned with where the neutrons are born, where they're going, and what happens when they get there. Lots and lots and lots of times over.

For completeness, the Boltzmann transport equation is -

$$\Psi(\underline{r},\underline{v}) = \int [\int \Psi(\underline{r}',\underline{v}')C(\underline{v}' \rightarrow \underline{v})d\underline{v}' + Q(\underline{r}',\underline{v})]T(\underline{r}' \rightarrow \underline{r})d\underline{r}' \tag{1}$$

Where -

$\Psi(\underline{r},\underline{v})$ - particle collision density
$Q(\underline{r}',\underline{v})$ - source density; fixed, fission, eigenvalue
$C(\underline{v}' \rightarrow \underline{v})$ - collision density; velocity change at fixed position
$T(\underline{r}' \rightarrow \underline{r})$ - transport density; position change at a fixed velocity

Monte Carlo methods will then employ a Markov scheme applying superposition to solve the integral equation. The trick is that the use of Markov makes the solution 'memoryless'; the current state of the particle is only dependent on the prior state. Then, *histories* are simulated for a particle, or a sequence of states. Each occurence in the sequence is then *tallied*. These terms are frequently used in MCNP so I wanted to define them here.

MCNP then simulates a single particle history from birth to death by tallying the random walk for the particle. Collisions are modeled by physics equations and cross sections. The free flight of the particle is modeled using the computational geometry of the model.

There are a few assumptions that are inherent in Monte Carlo solutions.

(1) The medium is static and homogeneous. Although use of homogeneous here just means it doesn't change.
(2) The solution is time independent.
(3) Markov principles are appled.
(4) Relativistic effects are neglected.
(5) Particle 'fly' in straight lines.

Right off the bat, you can see that for reactor concepts like a molten salt reactor, MCNP isn't the best choice, althought it will produce reasonably meaningful results. However, MCNP can solve a wide variety of problems. Even if using newer Monte Carlo computational tools, I would still recommend learning MCNP first.

13.2 Using Monte Carlo to compute π

Computation of π by Monte Carlo methods is a classic and fun example to illustrate the theory.

We know π has a special relationship with a circle.

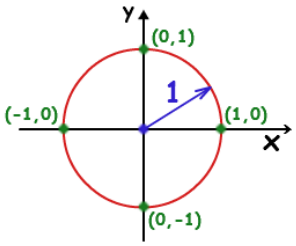

Figure 1: A unit circle.

The equation of the circle is -

$$x^2 + y^2 = 1 \tag{2}$$

And the area of this circle is -

$$A = \pi r^2 = \pi(1^2) = \pi \tag{3}$$

The procedure is then -

(1) Sample a random number for $x \in (0, 1)$.
(2) Sample a random number for $y \in (0, 1)$.
(3) Input x, y into Equation 2.
(4) Compute Equation 2.
(5) If the result is greater than 1, then discard.
(6) If the result is less than 1, then record. (I call it a 'hit'.)
(7) Repeat for N times.
(8) Tally the total number of hits.
(9) Divide the total hits by N.

Looking a Figure 1, you'll see we're recording the hits to the first quadrant of the circle. Essentially, we've evaluated the integral -

$$G = \int_0^1 \sqrt{1 - x^2} dx = \frac{1}{2}(x\sqrt{1 - x^2} - sin^{-1}x)|_{0,1} = \frac{\pi}{4} \tag{4}$$

To get π, multiply the result by 4.

Each 'particle' that is generated; i.e., (x, y) pair, is independent on the prior particle. Here, the 'interaction' is only whether the particle falls within the boundary. With a neutron, this could be a scattering or absorption event, which would be determined by cross sections and probability density functions. The concept is essentially the same on a fundamental level.

It should be clear that the choice of N is important. Increasing N will result in solution convergence. I have a python routine where you can experiment with the selection of N and observe the convergence. In fact, that's typically the first option in MCNP when the statistics aren't working out - increase the number of particles to generate more histories.

That's the extent of theory that is really necessary to run MCNP. Of course, as higher level problems pop up, you'll need to learn more, but this discussion should give you enough of a base that you'll be able to understand the theory in more depth if needed. And that's really engineering in a nutshell - learn what you need to solve the problem and whatever else comes after.

14 Getting to know the MCNP input file

Jezebel is the most basic MCNP problem. Everyone knows it. It is a bare Pu sphere with a Ni reflector. We'll start with a model of just a bare Pu sphere. We want to find the critical size. Download the input file and open it in your text editor. I'll be referring to line numbers, so turn that on.

The next page has a copy of the input file.

14.1 MCNP input file for a bare Pu sphere

```
Jezebel bare Pu sphere
c      ---------------------------------------------------------------
c      R.A.Borrelli
c      ---------------------------------------------------------------
c      rev.22.August.2014
c      ---------------------------------------------------------------
c
c      ------------------------- MODEL NOTES ----------------------------
c
c      The origin is the center of the sphere.
c
c      ----------------------- END MODEL NOTES -------------------------
c
c      ---------------------------------------------------------------
c
c      ------------------------- START CARDS ----------------------------
c
c      -----------
c      cell cards
c      -----------
c
c      ---
100    1 -19.86       -10       imp:n=1       $plutonium sphere
c      ---
c
c      ---
150    0               11       imp:n=0       $void
c      ---
c
c      -----------
c      end cell cards
c      -----------

c      -----------
c      surface cards
c      -----------
c
c      ---
c      geometry
c      ---
10     SPH  6.38493     $plutonium sphere
c      ---
c      end geometry
c      ---
c
c      -----------
c      end surface cards
c      -----------

c      -----------
c      physics cards
c      -----------
c
c      ---
MODE   n
c      ---
c
c      ---
c      end physics cards
c      ---
c
c      ---
c      source
c      ---
c
c      ---
KCODE  1000 1.0 30 130
KSRC   0 0 0
c      ---
c
c      ---
c      end source
c      ---
c
c      ---
c      materials
c      ---
c
c      ---
c      plutonium
M1     94239.66c 1.0
c      ---
c
c      ---
c      end materials
c      ---
c
c      ----------- END CARDS -----------
c
c      ---------------------------------------------------------------
```

14.2 General MCNP input file format

14.2.1 Title card

The very first line of the file is the title card. MCNP will print the title to the output file. This may be a relic of an old age because you can name the output file when you run MCNP on the command line. I suppose back when you had physical cards, this wasn't the case.

14.2.2 Comments

The MCNP input file is separated into 'cards'. I will talk about the format of the MCNP in general and then each card. First of all, note that `c` designates a commented line; i.e., the code won't read it. I know everyone probably knows this, but I'm being tedious on purpose to walk you through the code. Obviously, anything not commented out is read by the code. In addition to the `c` for commented lines, there is the use of `$` as a comment where a line is a card. In line 23, the cell labeled `100` ends with `$plutonium sphere`. Obviously, this means that the cell is the sphere of plutionium for which we are modeling for critical size.

Even for this simple model, you'll see my input file (also called as deck - as in deck of cards) is extensively commented. This is just good coding practice. Anyone not even remotely familiar with MCNP can see in the file what goes where. This is an older file, so I have my name and the date. I made this file before I had github. If you're using github, it's probably not necessary to include the date. I'd also put my github/twitter handle instead of my full name because it's easier to find me that way. A more recent deck for my used fuel cask is more extensively commented.

I have model notes section to describe what the geometry is, etc. I would strongly recommend to note the origin in the geometry, especially when the geometry becomes increasingly complicated. Here, the origin is the center of the Pu sphere. MCNP doesn't actually care where the origin is because you actually define the geometry relative to the origin. Noting the origin in the comments is helpful in case you forget, or if someone else is going to use the file.

Next, I have a `START CARDS` comment. This is just so I know where the actual MCNP starts. It helps when the model notes section is extensive because I can just search the keyword in `vim` and go right to the start.

The dollar sign '$' is used as an inline comment. Don't use it in column 1. On line 23, there is a comment indicating that this line describes the plutonium sphere.
MCNP contains cell, surface, physics, source, materials, and print cards. These are all indicated in the file comments, except the print cards. We'll do that later when we add the reflector.

14.2.3 Blank lines

There are only TWO blank lines in the file. Line 34 and line 50. That's it. You will get the dreaded FATAL ERROR if there are more or if they are in the wrong position. One is at the end of the cell cards, and the other at the end of the surface cards. At lines 51,52 where I indicated the physics cards, to the end of the file is collectively called the data cards. I divided it up so I can just search by keyword.

14.2.4 FORTRAN format

Excluding the comments, there are some 'commands' that start in column 1. Each of these can be considered a card, specific to the section it is in. On line 82, a material card is designated `M1`. It is for the plutonium. I include a comment that labels the material. What would `100` mean? You can see it is in

the cell card section, so it is a card designating cell 100.

I use capitals for the card designations. It's not necessary. When the file gets really large though, it's just easier to read.

After each card, there is more information that defines the card. Note that the card information always starts in column 8. This is because of FORTRAN. Similarly, FORTRAN is limited to 80 columns per line.

14.3 Detailed card descriptions

14.3.1 Cell cards

Cell cards come first in the file. But what exactly is cell? A cell is an object you want to model or part of the geometry environment. A room could be a cell, but you might want to model the dose rate of material in the room. A cell can be a collection of cells. A fuel rod is a cell. However, the rod is composed of pins, which are also cells.

Cells are defined by an assemblage of surfaces that you define in the surface card section.

For my first cell card, I designate the cell that can't be further decomposed into other cells. There's no required way to list the cell cards, but I recommend to pick a style and stick with it.

I label cells starting with `100`. This is just a style convention. Cell, surface, and material cards can have the same label, but you can see that this could be confusing for complicated geometries.

In this example, the first cell is the plutonium sphere. After the cell designation (`100`), the next piece of information is a `1`. This refers to material 1, which is plutonium. Next, there is a `-19.86`. This is the material density. Use of the negative sign indicates that the density is given in grams per cubic centimeter. Use of a positive number would indicate units of atom density. A + sign is not used. I like to stick to mass density because it's easy to look up quickly. PNNL has compiled a document of over 350 materials with densities and atom fractions for over 350 materials.

The next piece of information is a `-10`. This is the surface card that comprises the cell. Here, we only have one surface, which is the sphere. For a room, there would be six surfaces. The negative sign indicates the 'sense' of the surface, and this will be discussed separately.

Finally, there is `imp:n=1`. This denotes the cell importance for neutrons. Just about all MCNP problems will have a cell importance of 1. This just means that a neutron entering or born in the cell will be treated as one neutron when MCNP 'tracks' it. Each cell must have an importance.

I've changed the cell importance a grand total of one time in my decades of using MCNP. Leave it as 1.

A second cell is also listed (`150`). What is different about it when compared to cell `100`? Because I commented the file well, it is clear that this is void cell. In MCNP, the last cell always has to be a void. It is literally the nothingness that exists outside the model universe. Who knew MCNP was so zen? The material will always be `0`, and the neutron importance is zero `imp:n=0`. A neutron entering the void just disappears and is not tracked.

How does MCNP know exactly what is the void? This comes back to sense. The void is essentially the complemenatary sense of all the outer cell boundaries to it. This is fairly easy here because the void is just the positive sense to surface 10, which is just the Pu sphere. This will become clearer when we

discuss sense.

As a final note, the spacing of each piece if information is not relevant. I like to have everything line up. For teaching, it's just easier for the class to see the information on the card when it's displayed up on the sceen. The order is important. And the material designation must start in column 8.
Review.
```
100 1 -19.86 -10 imp:n=1 $plutonium sphere
```

(1) cell label - `100`
(2) material definition - `1`
(3) mass density - `-19.86`
(4) surface definitions and sense - `-10`
(5) importance - `imp:n=1`
(6) comment

It doesn't matter how complicated the geometry gets. The cell cards all have the same format!

14.3.2 Surface cards

After the cell cards is a **BLANK LINE**.

Then, the surface cards are defined.

For this problem, the only surface is the Pu sphere. On line 42, similar with the cell cards, it starts with the surface label, `10`. Again, there isn't any numbering convention. I like to use double digit numbers for the label. I also commented that this surface defined the fission source.

Next, `SPH` defines the surface, a sphere. Of course, there are numerous surface definitions. These are in the manual. I'm not going to dump them all out here.

When defining any surface, numerical dimensions are required. For `SPH` only requires a radius - `6.38493`. That's it! Capitals aren't needed. It is just my style convention.
Looking at cell `100`, you can now see that the cell is comprised of surface `10` with negative sense. In this case, the cell is the single surface with the defined sense and importance.
Review.
```
10 SPH 6.38493 $plutonium sphere
```

(1) surface label - `10`
(2) surface definition - `SPH`
(3) surface dimensions - `6.38493`
(4) comment

14.3.3 Physics cards

MCNP allows the user to define all sorts of physics models. I rarely have used any for the problems I've worked on. In this file, I have included `MODE n` to designate neutron transport. This is the default so it is not really needed. I include it for completeness. You can add `MODE n p e` for photons and electrons, respecively. So, I have the card in place already in case I need to include photons or electrons. Since we are only interested in determining the critical mass for the bare Pu sphere, the neutrons are all we care about. It doesn't hurt to be overly deliberate though.

14.3.4 Source cards

MCNP requires a source to be defined. For computations involving k_{EFF}, the `KCODE` and `KSRC`. The `SDEF` card can be used in place of the `KSRC` card for more complicated problems. It actually offers an enormous amount of flexiblity. That is a topic for a higher level walkthrough.

The `KSRC` card contains x y z coordinates designating a fission source point. Essentially, where each neutron is first 'born'. It stands to reason then that you want to make sure you designate the source point within the cell containing the fissile material. This is another reason why it's important to note where the origin is. Here, a source point is designated at the orgin - `0 0 0`. From experience, I know this is sufficient. However, I could have as many as I want. If we had two spheres, source points should be designated in each.

The `KCODE` card tells MCNP to compute k_{EFF} and how to do it.