McStas neutron ray-trace tutorial

Peter Willendrup and Kim Lefmann, Risø National Laboratory

March 29, 2006

Postal adress:
Risø National Laboratory
Frederiksborgvej 399
DK-4000 Roskilde, Denmark

email:
peter.willendrup@risoe.dk,kim.lefmann@risoe.dk

Abstract

This document is a tutorial about McStas and neutron scattering for beginners.

The text below is also included as a chapter in the McStas manual.

1 Introduction

This paper has been written to help out novel users of McStas and neutron scattering instruments. McStas is a software package for simulating neutron scattering experiments using a ray-tracing technique. This paper aims at helping the user to gain insight into basic neutron scattering as well as neutron raytracing using the McStas software package [1],[2],[4].

1.1 Prerequisites

Needed knowledge and equipment work through the tutorial is

- Undergraduate knowledge of mathematics and physics
- A computer with McStas installed (refer to the McStas homepage [4] for details) or a bootable McKnoppix cdrom (no installation required)
- This paper

1.2 Goals and tasks

The goals and tasks of this paper are

- To teach you about the most basic neutron scattering
- To let you understand some of the typical components in a neutron scattering instrument
- To teach you basic usage of the McStas neutron simulation package
- To let you create your first McStas instruments, a two axis diffractometer and a triple axis spectrometer

- To teach you how to modify your instrument for a specific task
- To help you learn to debug instruments
- To help you aquire and analyze data from McStas simulations

2 Basic neutron scattering

You may recall the Bragg law from your high school physics

$$n\lambda = 2d\sin(\theta),$$

giving the scattering condition for a wave of wavelength λ against a series of lattice planes with lattice spacing d, rotated the angle θ off the lattice plane normal. n is an integer giving the spectral order of the scattered wave. In neutron science one often refers to the *scattering vector*, $\vec{\kappa}$ of a given reflection, where

$$\kappa = |\vec{\kappa}| = \frac{2\pi}{d}.$$

This gives us the scattering vector formulation of the Bragg law

$$\kappa = 2k\sin(\theta),$$

where $k = \frac{2\pi}{n\lambda}$. The Bragg law / scattering condition is illustrated in Figure 1. Most of the neutron

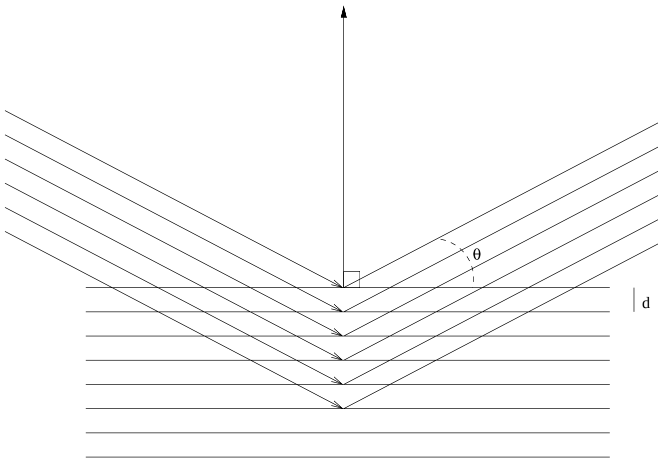

Figure 1: Illustration of the Bragg Law.

processes we will study in this paper are elastic, meaning that the wavelength of the neutron is unaltered by the process.

3 Basic understanding of instrument components

In the McStas formulation of a neutron scattering instrument, all objects apart from the neutron ray are referred to as components. This includes for instance

- **Source** The exit of a neutron production facility, where neutron rays of certain velocities are emitted into some portion of space

- **Monochromator** (Idealized) crystals used to select a neutrons of a single wavelength λ_0 [1] to probe the sample with (monochromator) or to analyze (analyzer)
- **Sample** An object altering the neutron physical properties in some sense, examples used here are
  - Vanadium. Scatters incoming neutron rays incoherently
  - Powder2. Can be thought of as a large number of crystals, each scattering neutron rays according to the Bragg law, thereby producing two concentric Debye Sherrer cones. This sample also has the posibility of adding inchoherent, eleasticly scattered neutron rays.
- **Monitors** Objects *monitoring* or registering neutron ray characteristics. In the exercises below are used different types of detectors or monitors:
  - Monitor. Single monitor, detecting the number of flying through a plane. (User defined opening size)
  - PSD_monitor. Square monitor, detecting the number of neutron rays passing through a plane, divided into pixels. square regions of a plane. (User defined resolution and opening siz)
  - PSD_monitor_4PI. As PSD_monitor but shaped like a sphere.
  - L_monitor. Wavelength monitor, measuring the different wavelengths of the passing neutron rays. (L is for λ)
  - Monitor_nD. General monitor for detecting all sorts of physical properties of the neutron ray. In our cases used with options
    - 'single' - as PSD_monitor but only one small square
    - 'banana' - as PSD_monitor but shaped like a curved, horizontal band
- **Collimators** Devices controling the direction and divergence of the neutron ray.
  - Collimator_linear A series of parallel absorbing neutron plates that limits the beam divergence. Typical values are 0.1° to 2°.

More information on the McStas components is available by using the `mcdoc` program (You may need to set the BROWSER system variable to your webbrowser of choice):

- `mcdoc -s` , Shows a html list of all the components
- `mcdoc Monitor.comp` , Shows the documentation for a given component
- `mcdoc -M` , brings up the McStas manual in PDF format
- `mcdoc -c` , brings up the McStas component manual in PDF format

4 Basic McStas

In short, the core of the McStas system is a precompiler. From a user-provided instrument description, components are assembled into a single piece of `ansi-c` code. Using a compiler, e.g. `gcc`, the c code is compiled into an executable program which can be run on your computer. Optionally, the program takes input arguments to tune the setup of your instrument/simulation. This section will take you through a simple example instrument to teach you the basic instrument language of McStas. (Instrument filename is vanadium_example.instr, can be loaded using the `Neutron Site/Tutorial` menu item of the `mcgui`, see below) Please study *carefully* the instructive comments, marked by `/* ... */` characters

```
/* The line below defines the 'name' of our instrument */
/* Here, we have a single input parameter, ROT         */
DEFINE INSTRUMENT vanadium_example(ROT=0)
```

[1] In reality, the monochromator selects a normal distribution of wavelegths around λ_0

```
/* The DECLARE section allows us to declare variables  */
/* in c syntax. Here, coll_div (collimator divergence) */
/* is set to 60 arc minutes...                         */
DECLARE
%{
  double coll_div = 60;
%}

/* Here comes the TRACE section, where the actual      */
/* instrument is defined....                           */
TRACE

/* The Arm() class component defines reference points  */
/* and directions in 3D space. Every component instance*/
/* must have a unique name. Here, arm is used. This    */
/* Arm() component is set to define the origin of our  */
/* global coordinate system (AT (0,0,0) ABSOLUTE)      */
COMPONENT arm = Arm() AT (0,0,0) ABSOLUTE

/* Next, we need some neutrons. Let's place a neutron  */
/* source. Refer to documentation of Source_flat to    */
/* understand the different input parameters.          */
/* The source component is placed RELATIVE to the arm  */
/* component, meaning that modifying the position or   */
/* orientation of the arm will also affect the source  */
/* component (and other components after that one...   */
COMPONENT source = Source_simple(radius = 0.015, dist = 1,
  xw=0.024, yh=0.015, E0=5, dE=0.2)
 AT (0,0,0) RELATIVE arm

/* Here we have a collimator - placed to improve beam  */
/* divergence. The component is placed at a distance   */
/* RELATIVE to a previous component...                 */
COMPONENT collimator = Collimator_linear(len = 0.2,
  divergence = coll_div, xmin = -0.02, xmax = 0.02,
  ymin = -0.03, ymax = 0.03)
  AT (0, 0, 0.4) RELATIVE arm

/* We also need something to 'shoot at' - here a sample*/
/* made from vanadium - an isotrope scatterer. Options */
/* are available to restrict the solid angle in which  */
/* neutrons are emitted (no need to simulate anything  */
/* that we know for sure will not gain us more insight)*/
/* Other options for smart targeting are available -   */
/* refer to component documentation for info.          */
COMPONENT target = V_sample(radius_i = 0.008, radius_o = 0.012,
  h = 0.015, focus_r = 0, pack = 1,
  target_x = 0, target_y = 0, target_z = 1)
  AT (0,0,1) RELATIVE arm
```

```
/* Here, a secondary arm - or reference point, placed  */
/* on the sample position. The ROT parameter above     */
/* defines rotation of this arm (and components        */
/* relative to the arm)                                */
COMPONENT arm2 = Arm()
  AT (0,0,0) RELATIVE target
  ROTATED (0,ROT,0) relative arm

/* For data output, let us place a detector. This      */
/* detector is not very realistic, since it is sphere  */
/* shaped and has a 10 m radius, but has the advantage */
/* that EVERYTHING emitted from the sample will be     */
/* picked up. Notice that this component changes       */
/* orientation with the ROT input parameter of the     */
/* instrument.                                         */
COMPONENT PSD_4pi = PSD_monitor_4PI(radius=10, nx=101, ny=51,
  filename="vanadium.psd")
  AT (0,0,0) RELATIVE target ROTATED (ROT,0,0) RELATIVE arm2
END
```

Enlightened by the above example, you are probably now ready to learn a few more important details and tips about McStas.

- **Neutron representation:** A neutron 'history' or package is an entity representing a large number of neutrons. It has the following physical properties:
  - Spatial coordinates, $\vec{x}$ or x, y, z
  - Velocity components, $\vec{v}$ or v_x, v_y, v_z
  - Spin components, $\vec{s}$ or s_x, s_y, s_z
  - Time, t
  - Neutron weight factor, p

  initial *neutron weight* p_0.
- **Neutron histories/Intensities:** McStas simulates neutron histories rather than direct neutron counts, i.e. when a Monte Carlo choice is made in a given component (e.g. a random number is generated to decide a new direction of the neutron ray), the neutron *weight factor* is adjusted accordingly. As you may have guessed already, the weight factor is actually a probability of observing a neutron of the given behaviour. The transition to direct neutron intensites is made by adjusting the initial neutron weight of the source component, giving the absolute flux of neutrons emitted in one second. This means that the intensity of the neutron beam at a given position is the initial neutron weight multiplied by the product of all the Monte Carlo weight factors occuring from the source to the given position. When observing McStas output, I is the intensity, not N.
- **3D space:** The 3D space in which the instrument is defined, usually has a single component which is placed ABSOLUTEly in space, e.g. at (0,0,0). All other components can be placed RELATIVE to this component.
- **Changing coordinate system:** Each component has its own local coordinate system. As the neutron travels from one component to the other, the local component coordinate system changes. The definition is that z is the direction toward the next component, and that the y direction is vertical.
- **Component order matters!** It is important to understand that McStas is component order dependent. The basic idea is to follow the neutron as it travels from one component to the next

in the instument description. This means that if you place one component *geometrically* before another component, but *orderly* after the other component, neutrons may never reach your 'first' component. This means that some designs can be difficult to achieve, though generally a solution can be found.

- **Use Arm()'s!** The Arm() component is very good for defining changed orientation of the instrument, e.g. for axis turning points etc. Placing many Arm()'s will improve future flexibility of your instrument.
- **Use PSD_monitor()'s!** The PSD_monitor() component is a **p**osition **s**ensitive **d**etector. This component can be used to image the shape of your beam as it travels through the instrument. This is very useful for debugging purposes. Other monitors, for instance wavelength monitors can also be useful.

In the McStas manual, available by clicking here if you are using an internet browser to view this document, description of usage of the different McStas tools is printed. The main McStas programs are

- *mcstas* - Core application
- *mcgui* - Main graphical user interface
- *mcdisplay* - Ray trace / debugging application
- *mcplot* - Data / display application
- *mcdoc* - Documentation application

Here are a few hints on using the tools:

- To start mcgui, execute `mcgui` in a terminal window (`mcgui.pl` on Windows)
- To handle instrument files (opening, editing, compiling), use `File` menu of `mcgui`
- To simulate and plot data, use the `Simulation` menu of `mcgui`
- To use the distributed example McStas instruments, use the `Neutron Site` menu of `mcgui`
- For further help on usage, use the items of the `mcgui` menu of `Help` menu or read the chapter *Running McStas* of the McStas manual [2].

5 Exercises

Throughout the rest of this paper, you will have to do the work! Through a series of small exersices, you will set up and use two simple neutron scattering instruments: a two axis diffractometer and a triple axis spectrometer. To get an idea of what your final instrument might look like, see the sample instrument protrayed in Figure 2.

5.1 Exercise: Source and PSD

1. Start by inserting a template instrument (Insert menu). Set up an instrument, consisting only of an Arm (keep the example 'a1' Arm instance), a Source_Maxwell_3, a PSD_monitor and an L_monitor. Read the documentation for each of the components to find the needed input parameters. For the source we will help you out, try

   ```
   COMPONENT source = Source_Maxwell_3(
       size = 0.1, l_low = 0.1, l_high = 10, dist = 10, xw = 0.01,
       yh = 0.01, T1 = 300, T2=300, T3=300, I1=1e14, I2=0, I3=0)
   ```

   Read the Source_Maxwell_3 docs using mcdoc to understand the suggested parameters. The file names for monitors, specified as character strings, should be given *surrounded* by double quotes ("file.dat").

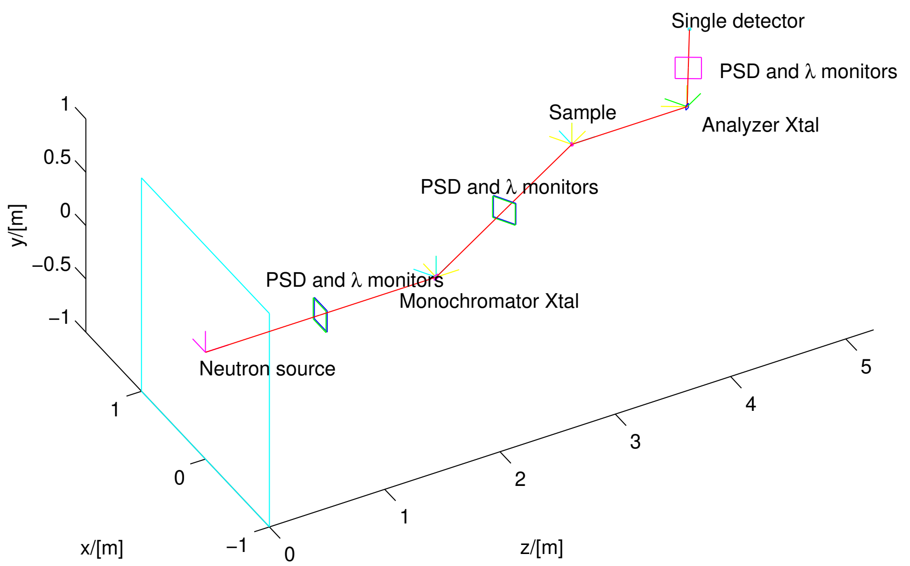

Figure 2: Illustration of a triple axis diffractometer

2. Run a simulation and plot the results, looking at an image of the source plus the wavelength distribution of the source.
3. Narrow down the interval of wavelengths emitted from the source to e.g. l_low=0.999 and l_high=1.001. Rerun your simulation to check the effect. Reset the wavelength interval to [0.1 10] Å.
4. Estimate the solid angle covered by your PSD. Try if you can understand I, the neutron intensity in the PSD, in the two previous runs. Try running the simulation with half or double the number of neutron rays. Try also to vary the source focus area. Explain what you observe.

5.2 Exercise: Insert a monochromator

1. Keeping your current components, insert a Monochromator_flat component (use mcdoc or the component manual to get the needed parameters) and a new set of PSD and L_monitor after the monochromator. You should now add two new input parameters of your instrument, e.g. OMM (Omega Monochromator) and TTM (Two Theta Monochromator) for rotation of the monochromator and the remaining part of the instrument so that the orientation of monochromator is as portrayed in Figure 3. These two instrument input parameters should be added to the DEFINE line of the instrument (start of file). Remember to add an Arm()'s at the rotation point.
2. Given $\lambda = 4$Å, and knowing that for the monochromator $\kappa = 1.8734$ Å^{-1} (Pyrolytic Graphite), use Bragg's law to determine the correct Bragg angle (i.e. OMM/TTM) for the monochromator for the $n = 1$ reflection.
3. Do a scan of OMM a couple of angles around this value to verify the finding, keeping TTM fixed. This will be achieved by setting the OMM value to 'min,max' (replace by relevant numerical values)

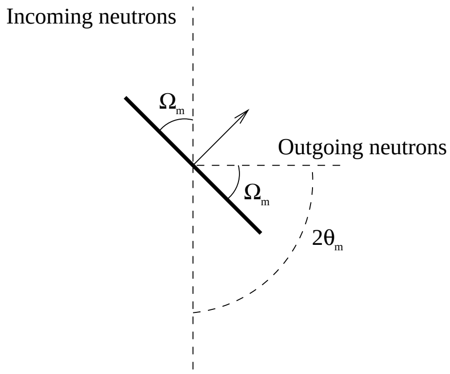

Figure 3: Illustration of the monochromator orientation

in the Run Simulation window, and selecting the Scan check button in that same window. Enter the number of steps to compute (e.g. 10). Check the position of the peak on the PSD and the wavelength on the L_monitor.

4. What should κ be set to to get the $n = 2$ reflection at exactly OMM=45° (TTM=90°)? Adjust κ for the monochromator and verify the calculation by a scan, check the wavelength.
5. Determine the Bragg angle for the $n = 1$ reflection in this setting of κ, and verify by scaning OMM. Set OMM to this value. Perform the simulation and check the wavelength distribution. Comment.
6. Before you go on, change the minimum and maximum wavelengths of the source to a narrow interval around the wavelength you select, e.g. 4Å. (No need to produce neutron rays that will not be scattered at the monochromator...)

5.3 Exercise: Insert a sample

1. Now, insert a V_sample() component (Vanadium, scatters in all directions) after the last PSD_monitor and L_monitor. Insert a Beamstop component after the sample, e.g. AT (0,0,0.5) RELATIVE sample. At the same position as the sample, insert a PSD_monitor_4PI_log component of radius 1.0 m., located AT (0,0,0) RELATIVE to your sample. Read the documentation for details on input parameters. Run a simulation. Notice the number of hits.
2. Restrict the solid angle of scattering from the vanadium sample, focus on the PSD_monitor_4PI_log by using `target_x=1` and e.g. `focus_r=0.2` as input parameter to the sample. Run a simulation. Notice the number of hits. Improvement?

5.4 Exercise: Insert a different sample

1. Next, let us insert something more interesting. Remove the V_sample and PSD_monitor_4PI_log. Insert a Powder2 component with the following parameters:

```
COMPONENT sample = Powder2(radius=0.01,h=0.01, q_1=1, q_2=1.3,
   w_1=0, w_2=0, d_phi0=0.1, pack=0.5, DW=0.9, frac=0.5,
   focus_r=0.03, j_1=8, j_2=4, F2_1=1000, F2_2=1000,
   Vc=3.86*3.86*11.82, sigma_a=0, sigma_inc=2,
   target_x = 0, target_y = 0, target_z = 0)
  AT (0, 0, 0) RELATIVE a4
```

(Adjust placement relative to your component naming etc.). Also insert a banana shaped detector,

```
COMPONENT ND = Monitor_nD(xwidth=1, options="banana, theta limits=[-45 45],auto y")
  AT (0,0,0) RELATIVE a5
```

and take away the previously inserted Beamstop. Run a long simulation (many neutron rays) and have coffee.

5.5 Exercise: Insert an analyzer

1. Insert a single detector by changing the options for the Monitor_nD component to "single". Also, add an angle to rotate the part of the instrument located after the sample, e.g. TT (Two Θ) and decide a more relevant size of the now rectangular detector. Look at your results from the last simulation to determine an approximate scan range for the TT angle. Also, set a small focus_r on the sample component to minimize calculation time on non-detected neutron rays. Scan TT across the powder lines.
2. Between sample and detector, set up an analyser crystal by copying and modifying your monochromator component. (Add new arms and angles OMA, TTA - A is for Analyzer - for adjusment.) Adjust the analyser to Bragg condition for the chosen wavelength. Re-scan TT and notice the difference to the scan performed in the previous task. Try also scanning around -TT and notice the difference to the other scan.

6 Suffix

Well done, you have come to the end of the McStas tutorial. Hopefully, most of the goals of the tutorials have been fulfilled. Otherwise, feel free to contact the authors of this paper or the McStas users mailinglist for further help.

References

[1] K. Lefmann and K. Nielsen: *McStas, a general software package for neutron ray-tracing simulations*, Neutron News, **10** pp. 20-23, 1999

[2] P. Willendrup, E. Farhi K. Lefmann et. al.: *User and Programmers Guide to the Neutron Ray-Tracing Package McStas, Version 1.9*,Risø National Laboratory, Roskilde, Denmark, January 2005

[3] P. Willendrup, E. Farhi K. Lefmann et. al.: *Component Manual for the Neutron Ray-Tracing Package McStas, Version 1.9*,Risø National Laboratory, Roskilde, Denmark, January 2005

[4] McStas homepages:
Official website at Risø, http://mcstas.risoe.dk
Supplementary ILL website, http://www.ill.fr/tas/mcstas