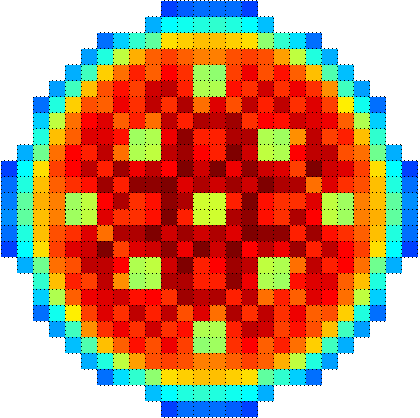
****

**CMS SUITE**

**2012-04-18**

Overview

The CMS system is built on a number of different programs; CASMO, SIMULATE, S3K and many more. All these programs will return a number of ascii and binary files with a huge amount of data. To do your own analysis of the data received and/or creating plots for reports and other documentations, the data from the files will be needed to be handled in a separate program.

CMS Suite is a collection of m-files used to read and manipulate data from the CMS system in MatLab, i.e. CMS Suite addresses the access portion to the left in the figure below.

Share

Basic Tools

The basic concept of CMS Suite is to have a easy reading tool for all different kinds of files and a number of basic geometric handles to be able to do your own analyses.   
The basic tools in CMS Suite are

* CmsRead – basic reading tool to get data into MatLab
* CmsTools – Geometrical tools
* CmsPlot – GUI plot tool

CmsRead

The reading tools are basically working the same way. Firstly a metadata variable the “info” structure is created. This structure contains the geometric data for the core, and a list of the different data that can be read from the file. The data could be scalar data or core wide data, which we will refer to as ‘distributions’.

There are a number of different m-files to read from each kind of file. For Simulate files there are one m-file that can handle \*.sum, \*.out, \*.cms, \*.res, \*.pinfile.

Syntax first read  
coreinfo = ReadCore(filename);

The structure coreinfo will always contain the following data:

* coreinfo.core – contians geometrical data
  + mminj – core conture
  + kmax – axial nodes
  + kan – number of channels
  + knum – channel numbers (matrix if not full symetrie)
* coreinfo.fileinfo – file data
  + path
  + time stamp of simulate run
* coreinfo.Xpo – exposure points on the file
* coreinfo.distlist – available distributions present on the file

Example 1: first read  
coreinfo = ReadCore(‘sim-dep.cms’);

The second read is when data is wanted. The info structure (coreinfo) from the first read will now be the first argument and should always be used. The second read can be done in a number of different ways.

Syntax second read 1  
dist = ReadCore(coreinfo,’distlabel’);

The first kind of read the ’distlabel’ is the keyword found in coreinfo.distlist. There can be a number of other keywords working, these can be found in fields such as

* coreinfo.cards (\*.sum)
* coreinfo.prista (\*.out),
* coreinfo.ScalarNames and coreinfo.DistNames (\*.cms)
* coreinfo.misclist and coreinfo.data.Label (\*.res)

distlabel may also be a cell array with a number of distlabels.

When called with no third input argument, the first state point of the wanted distribution will be returned as a matrix of numbers with size kmax-by-kan where kmax is the numbers of nodes per assembly (kmax=1 if the distribution is 2D) and kan is the number of fuel bundles in the symmetry used. If a scalar is read, the default is to return the data for all state points.

Example 2: second read 1  
dist = ReadCore(coreinfo,’3RPF’);

Returns:

dist = matrix size(kmax-by-kan)

Example 3: second read 1

dists = ReadCore(coreinfo,{‘3EXP’,’3RPF’});

Returns:

dist = struct with fields 3exp and 3rpf, size(kmax-by-kan)

Syntax second read 2  
dist = ReadCore(coreinfo,’distlabel’,st\_pt);

With the third argument st\_pt, the state point(s) wanted of the wanted distribution are read. ReadCore will handle

* exposure points (0,500, 1350 and so on)
* 1-99 (number in order)
* 10000, 20000 (first and last)
* ‘first’, ’last’, ’all’, ’ends’ (will give that specific state point, ‘ends’ means first and last)

The first two st\_pt options to ReadCore will accept arrays. There is no difference how scalar and distributions are treated here, except that distributions will be returned as cell arrays, while scalars will be returned as vectors.

Example 4: second read 2  
dist = ReadCore(coreinfo,’3RPF’,’all’);

Returns:

dist = cell array of all state points

Example 5: second read 2  
dist = ReadCore(coreinfo,’3RPF’,[1 2 3]);

Returns:

dist = cell array of the first three state points

With four inputs ReadCore will return the desired distribution, state point and the chosen assembly.

Syntax second read 3  
dist = ReadCore(coreinfo,’distlabel’,st\_pt,assembly);

The assembly option will handle

* Channel numbers
* i j coordinates
* serial numbers

Example 6: second read 3  
dist = ReadCore(coreinfo,’3RPF’,[1 2 3],150);

Returns:

dist = cell array of the first three state points of assembly in channel 150.

Example 7: second read 3  
dist = ReadCore(coreinfo,’3RPF’,[1 2 3],’50365’);

Returns:

dist = cell array of the first three state points of assembly with serial number ’50365’.

Example 8: second read 3  
dist = ReadCore(coreinfo,’3RPF’,[1 2 3],[1 11]);

Returns:

dist = cell array of the first three state points of assembly on position i:1 j:11.

CmsTools

CmsTools is a collection of different geometrical tools for the data read from CmsRead.

There are a function for core, control rod and detector geometries. One of the most common arguments to these functions are mminj (found in coreinfo.core.mminj) or for the control rod crmminj (same as mminj but for the control rods).

Example 9: CmsTools  
coremap = vec2cor(corevec,coreinfo.core.mminj);