OpenNodal User Manual

Nicholas F. Herring

William C. Dawn

September 2, 2022

Revision Log

Revision	Date	Affected Pages	Revision Description
0	September 2, 2022	All	Initial Release

Acronyms

Contents

1. Input Format

1.1 Main Input File

OpenNodal's input is separated into blocks describing aspects of the problem. Each block may be given a number of cards describing specific details about the problem parameters. OpenNodal currently features the following three blocks:

Block Name	Available Cards
[CASE_DETAILS]	title, nsplit, k_eps, phi_eps, max_its, nodal_method, and wielandt
[CORE]	dim, size, apitch, sym, assm_map, bc, refl_mat, and buckling
[MATERIAL]	xs_file and xs_map

1.1.1 [CASE_DETAILS] Block

The [CASE_DETAILS] block describes details for the problem case, including run specifications such as convergence criteria and solution method. The following table describes the available input cards:

Card Name	Description	Options	Required?
title	Problem Title.	String of size ≤ 100	No. Default: blank
nsplit	Spatial Refinement for nodes. Splits each assembly in the specified geometry into nsplit×nsplit nodes.	$\mathbb{Z} \geq 1$	No. Default: 1
k_eps	Eigenvalue convergence criteria.	$\mathbb{R} > 0$	No. Default: 1.0E-6
phi_eps	Flux convergence criteria.	$\mathbb{R} > 0$	No. Default: 1.0E-5
max_its	Maximum number of outer iterations.	$\mathbb{Z} \geq 1$	No. Default: 100
nodal_method	Nodal method to be employed. Either Finite Difference or polynomial NEM	fd, poly	No. Default: fd
wielandt	Wielandt shift $\delta\lambda$ value. A value of 0 will leave it turned off.	$\mathbb{R} > 0$	No. Default: off (0)

1.1.2 [CORE] Block

The [CORE] block describes the geometry of the core itself, including auxiliary geometric information such as axial buckling, symmetry, boundary conditions, etc. The following table describes the available input cards:

Card Name	Description	Options	Required?
`dim`	Dimensionality of problem. 1D, 2D, 3D. Currently ONLY 2D is supported.	`1D, 2D, 3D`	No. Default: `2D`
`size`	Size of the core across. Cores are assumed to be square.	$\mathbb{Z} \geq 1$	No. Default: `1`
`apitch`	Assembly pitch in cm. All assemblies are assigned assumed square and given equal widths. Non-square assemblies may be constructed using multiple assemblies.	$\mathbb{R} > 0$	No. Default: `1.0E0`
`sym`	Problem symmetry. For half (`half`) symmetry it is assumed the left side is reflective, and for quarter (`qtr`) symmetry it is assumed that the left and top sides are reflective.	`full, half, qtr`	No. Default: `full`
`assm_map`	Assembly map matrix. Matrix starts on the next line. Will have number of rows and columns equal to `size` unless symmetry is used. For `half` symmetry the number of columns will be $\lceil$`size`$/2\rceil$. For `qtr` symmetry the number of columns AND rows will be $\lceil$`size`$/2\rceil$. Any gaps will be filled with `refl_mat`, so if gaps are left for ragged core specification, then `refl_mat` MUST be specified in this case.	Map of $\mathbb{Z}$ assembly IDs	Yes.
`bc`	Boundary conditions to be applied. These are not applied to the reflective sides for a problem with non-`full` symmetry. For `albedo`, $\mathbb{R}$ albedo values must also be given on the same line for each energy group. For `reflector`, the `refl_mat` will be added as a single assembly buffer as well as self-scatter diffusion length boundary, so `refl_mat` MUST be specified in this case.	`vacuum, reflective, reflector, zero, albedo`	No. Default: `vacuum`
`refl_mat`	Reflector material index to be used in either `reflector` boundary conditions and/or to fill in gaps for a ragged core.	$\mathbb{Z} \geq 1$	No. Default: no reflector material
`buckling`	Axial geometric buckling. If `height` is given, then it should be immediately followed by a positive real height on the same line and the axial buckling will be computed from that height.	$\mathbb{R}$, `height` $\mathbb{R}$	No. Default: `0.0E0`

1.1.3 [MATERIAL] Block

The `[MATERIAL]` block describes cross sections for each assembly. The following table describes the available input cards:

Card Name	Description	Options	Required?
`xs_file`	Cross section filename. Can include the path to the file for a cross sections file not in the same directory.	String of size $\leq$ 200	Yes.
`xs_map`	The mapping for cross section identifiers to assembly IDs as well as specifications about macro or micro cross sections. Currently ONLY macro cross sections are supported. The card should be followed with the number of assembly IDs to map onto on the same line. This number should at least equal the maximum assembly ID in the `assm_map`. On each following line the assembly cross section mapping should be given as `ID macro/micro XSid` counting from assembly `ID` from 1 up to the number of IDs to map.	$\mathbb{Z} \geq 1$ 1 `macro <XSid>` 2 `macro <XSid>` $\vdots$	Yes.

1.2 Cross Section Input File

The OpenNodal cross section file contains all cross section data for an OpenNodal calculation. Currently, only macroscopic cross sections are supported, so only the description for them will be given. The following is the order of the data as it appears in the cross section file (note that blank lines between blocks are ignored):

```
Line 1: OpenNodal_ASSY-XS_V1 <num_mats> <G>
Block 1: Each entry in this block contains cross sections for a single material.
Each block contains G+5 lines. There are num_mats blocks.
Entry line 1: id <XSid>
Entry line 2: diffusion_coefficient_1 diffusion_coefficient_2... diffusion_coefficient_G
Entry line 2: fission_spectrum_1 fission_spectrum_2... fission_spectrum_G
Entry line 3: nuSigma_f_1 nuSigma_f_2 nuSigma_f_3... nuSigma_f_G
Entry line 4: nu_bar_1 nu_bar_2... nu_bar_G
Entry line 5: Sigma_a_1 Sigma_a_2... Sigma_a_G
Entry line 6: sig_scat_{1->1} sig_scat_{2->1}... sig_scat_{G->1}
Entry line 7: sig_scat_{1->2} sig_scat_{2->2}... sig_scat_{G->2}
        :
Entry line G+5: sig_scat_{1->G} sig_scat_{2->G}... sig_scat_{G->G}
        :
```

Where:

- `num_mats` = Total number of cross section materials (presumably homogenized assemblies). There should be this many blocks.
- `G` = Total number of energy groups.
- `XSid` = ID of the material. Used in identifying the material and region mapping. Not limited to integers, can be a general string.
- `diffusion_coefficient_g` = Diffusion coefficient in energy group g (D).

- `fission_spectrum_g` = Fraction of neutrons born in fission that appear in energy group g (χ).
- `nuSigma_f_g` = Fission production cross section in group g ($\nu\Sigma_f$ NOT Σ_f).
- `nu_bar_g` = Average number of neutrons released by fission caused by a neutron in energy group g (ν).
- `Sigma_a_g` = Absorption cross section in energy group g (Σ_a).
- `sig_scat_{g'->g}` = Scattering cross section from group g' to g ($\Sigma_{s,g'\to g}$).

2. Output Format

If the input filename is of the form of one of the following `<file>.in, <file>.inp, <file>.input` then the output filenames will take on the form be `<file>_<type>.<ext>`. If the input file has any other extension, then the entire filename will simply be considered `<file>` and similarly use the convetion `<file>_<type>.<ext>` for each of the output files.

2.1 Output Summary File

OpenNodal generates a summary output file for the run results. The naming convention for this summary file is `<file>_results.out`. This file contains the final eigenvalue on the first line, followed by the final flux core averaged ratio of ϕ_2/ϕ_1 iff the problem was a 2 group problem. The summary file will then have `G` blocks giving the assembly averaged flux for each energy group mapped in a manner matching `assm_map`.

2.2 Log File

OpenNodal generates a log file echoing all output which is also printed to the screen, including any warnings or errors. The naming convention for this summary file is `<file>.log`.

2.3 Flux Output Files

OpenNodal generates `G` flux output files as well as a fission power output file. The naming convention for these files is `<file>_flux_g<g>.csv` and `<file>_power.csv`. The data is listed in the following format:

```
x_centroid_node_row_1 y_centroid_node_col_1 flux_{g,1,1}
x_centroid_node_row_1 y_centroid_node_col_2 flux_{g,1,2}
x_centroid_node_row_1 y_centroid_node_col_3 flux_{g,1,3}
        :
x_centroid_node_row_1 y_centroid_node_col_Ny flux_{g,1,Ny}

x_centroid_node_row_2 y_centroid_node_col_1 flux_{g,2,1}
x_centroid_node_row_2 y_centroid_node_col_2 flux_{g,2,2}
        :
```

So that it is a map for the node averaged flux of each group onto the centroid of the node to which it belongs. Notice that each change in rows is separated by a blank line. The power output is similar, except that the

flux is replaced by node averaged power. Note that since OpenNodal solves an eigenvalue problem, values should not be considered absolute but relative to each other.

2.4 Flux Output Plots

If the user has `gnuplot` installed, then OpenNodal will generate plots of the groupwise flux, as well as the fission power. The naming convention for these files is `<file>_flux_g<g>.png` and `<file>_power.png`. The files themselves are 2D heat-maps for the magnitude of the relative flux/power in each node across the whole core. Example for the `2d_ss_iaea-BSS11.inp` case with `nsplit 64`:

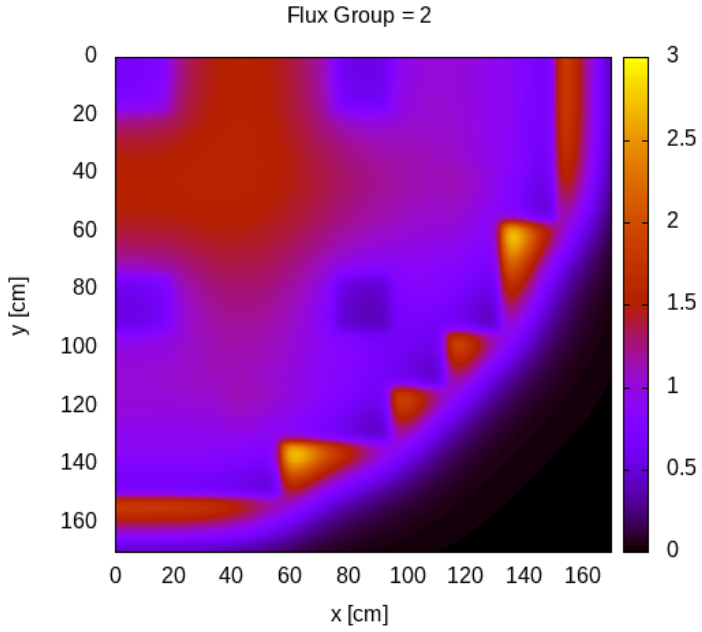

Figure 2.1: Thermal Flux for the BSS-11 Problem

Bibliography