




User's Manual

# FEMFFUSION
# A finite element code to model nuclear reactors

A. Vidal-Ferràndiz, A. Carreño, Y. Fontenla, D. Ginestar and G. Verdú

May 2, 2023

Version 1.0

## Contents


Introduction 3

1 Introduction 3

Program installation 3

2 Program installation 3

Program running 4

3 Program running 4

Stationary state computations 4

4 Stationary state computations 4

Benchmarks geometries 5

5 Benchmarks geometries 5

Input files 5

6 Input files 5

Main file or '**.prm**' file . . . . . 6

Output files 9

7 Output files 9

Benchmark examples 9

8 Examples 10

9 Related projects 14

Bibliography 15

# 1 Introduction

FEMFFUSION is a neutronic code written in C++ programming language that solves the multi-group transport equation using the neutron diffusion approximation and also the Simplified Spherical Harmonic approximation also known as the $SP_N$ equations [1].

The code uses for the spatial discretization of the continuous Galerkin finite element method to be able to solve different types of geometries and problems of different dimensions (1D, 2D and 3D) [2]. The code is built on top of the open source library DEAL.II which provides, in addition to support, advances in the finite element method [3]. This library includes many self-developed techniques for solving partial differential equations, linear algebra problems and computer science strategies, but it is also compatible and interfaces with other leading open source libraries in these fields. In our case, in addition to DEAL.II, we rely on the structures of PETSC, which is a tool with very sophisticated techniques for the parallel resolution of linear algebra problems [4]. On the other hand, we use techniques developed by the SLEPC library [5] together with other methodologies for solving eigenvalue problems. For the integration of time-dependent problems, we include the SUNDIALS library [6].

Finally, the code allows the use of matrix-free techniques, which do not need to assemble or save the matrices involved in the different linear systems to be solved, in order to keep computational memory demands at reasonable values and to increase the computational speed. [7].

# 2 Program installation

FEMFFUSION works in a GNU/Linux system like most numerical algebra application codes. For Debian-based distributions, an installation script is provided. With other distributions, installation can be done using the analogous commands. First, some numerical tools and libraries must be installed. This can be done by typing in the terminal:

```
sudo apt-get install make g++ git cmake
sudo apt-get install petsc-dev slepc-dev libdeal.ii-dev
sudo apt-get install gmsh paraview
```

After these steps, create or access the place where you want to have the FEMFFUSION code. Once there, clone the code repository and download, compile and test the auxiliary libraries. These last steps, which require an Internet connection, can be done by entering them in the terminal:

```
git clone https://Zonni@bitbucket.org/Zonni/femffusion.git
cd femffusion
cmake .
make release
make -j4
```

The last instruction will take quite a long time, as it has to compile the PETSC, SLEPC and DEAL.II libraries (about 30 minutes).

After s successful compilation of the code, it is recommended to run the FEMFFUSION application examples to check that the code works correctly. These examples are described in the Section "Examples". We can run them with the command:

```
./run_examples.sh
```

# 3 Program running

Once FEMFFUSION is installed, we can run the different problems. To do this, we only need to know the name of the main file (or `.prm` file) that contains all the parameters needed to solve the problem and enter the following command line in the terminal:

```
./femffusion.exe -f folder/filename.prm
```

where `folder` is the path of '`.prm`' file location, the two possible directories with these files names are `test` or `Examples`. Following the filename, other commands for code execution can be entered, either from FEMFFUSION (Table 1) or from the libraries. In this case, the command `-f` or `--file` indicates that the file name follows. Other useful command line options include:

- -**v** or --**verbose**. Displays more details of the code execution in the on-screen output.
- -**s** or --**silent**. Displays nothing on the screen.
- -**t** or --**test**. Runs a series of test files.
- -**p** or --**performance**. Evaluates the performance of matrix-free products and eigenvalue solvers in some test files.

# 4 Stationary state computations

FEMFFUSION uses the library DEAL.II to compute different problems associated with approximations of the neutron transport, such as the multigroup neutron diffusion equation and the $SP_N$ equations. These equations for the steady state have the following form

$$\mathscr{L}\phi = \frac{1}{k_{\text{eff}}}\mathscr{M}\phi\,, \tag{1}$$

where $\mathscr{L}$ is the transport differential operator, $\mathscr{M}$ the operator associated with the sources of neutrons, the vector $\phi$ denotes the neutron flux in steady state and the constant $k_{\text{eff}}$ is the effective multiplication factor of the reactor, that provides an idea of the criticality of the system.

After applying Galerkin finite element method to the differential equations on a specific spatial grid, an algebraic eigenvalue problem is obtained (more details in reference [8]). The obtained matrices have a block structure and they are sparse. They are constructed in the format of PETSc in order to be able to work with the algorithms already implemented in this

library together with the methods included in the library SLEPc. In addition, these structures would allow for parallelization of the code by means of PETSc. Another possibility is the use of the aforementioned *matrix-free* methodology, in which only the diagonal blocks are stored in memory and the rest are used by means of the matrix-vector product.

For the solution of the eigenvalue problem, FEMFFUSION has implemented numerous types of solvers in addition to those provided by the SLEPc library. In particular, the user can use the power method [1], the Krylov-Schur method [9] (from SLEPc), the *Generalized Davidson* method [10] (from SLEPc), the Block Inverse-Free Preconditioned Arnoldi (BIFPAM) method [11], the Modified Generalized Block Newton (MGBNM) [11] and a hybrid method based on the previous two (hybrid) [11]. Each one of these methods in turn has different types of initialization and preconditioners that optimize computational resources.

Once the problem has been solved, FEMFFUSION has a second layer that links the solution of the eigenvalue problem with the post-processing tools of the results.

## 5 Benchmarks geometries

The code is prepared for different types of geometries: rectangular geometries, hexagonal geometries, pin-level geometries. For each of these types, a number of recommendations are made regarding the representation of the reactor.

- **Rectangular reactors**. The code is implemented in such a way that, through a series of parameters, with the aids of DEAL.II library the discrete mesh of the problem is built. These values are: the maximum number of cells in each dimension, the size of each cell in all dimensions, and the values of the rectangular parallelepiped (formed by the maximum set of cells in each dimension) that is or is not part of the reactor geometry (bounding box).

- **Hexagonal reactors**. The DEAL.II library only allows the use of quadrangular finite elements, therefore, each hexagonal element is divided into three elements of this type for its representation. For the creation of these meshes, the authors make use of the GMSH library [12, 13].

- **Reactors at pin level**. The construction of the geometry is done using the DEAL.II library, indicating on the one hand the geometry at pin level and also the geometry of each one of the pins that contain the fuel.

## 6 Input files

For the execution of a problem FEMFFUSION always needs the main file '`.prm`' and at least one other file type that has the cross-sections (XS) of the materials and their distribution in the geometry. However, depending on the type of problem, other input files may be required. A summary of the input files for FEMFFUSION is the following one:

- **file.prm**. This is the main file and will contain all the parameters that we can provide to the code as input variables for problem-solving and post-processing. These variables will be detailed later in this Section.
- **file.xsec or file.xlm**. The first file is for problems with two energy groups, and the second one for more than two energy groups. In these files you will find all the information related to the materials of the problem (distribution and cross sections).
- **file.msh**. The code can also read meshes that come from the GMSH library [12, 13]. This file type is mainly used for hexagonal meshes.
- **file.bar**. If the stationary problem is the initial state of a transient where a bar movement takes place, we can enter the position of the bars with this file.
- **file.tri and file.tri.vec**. This is a file generated by FEMFFUSION (the first time the example is run) for the creation of the mesh in the case of spatial discretization of the pins.
- **file.xlm**. Contains data concerning the reactor geometry. This file is used for pin level calculations.

The name of the files is included in the main file, '`.prm`', as an input variable.

The input files '`.xsec`' and '`.xlm`' have completely different formats from each other. An '`.xsec`' input file provides geometric information plane-by-plane of the nuclear reactor with values representing each materials (i.e. material_id=1, 2, 3, 4, 5, 6), provides characteristics materials represented by their cross-section for a maximum of 2 energy groups, $\beta$ and $\lambda$ characteristics of precursors, and neutron speeds. An '`.xlm`' input file collects the vector and matrix values that represent the characteristics of the reactor using the cross-section of materials for each energy-group. C5G7 benchmark tasks uses '`.xlm`' input file with seven groups-energy. More information can be found in the FEMFFUSION validation report: `https://femffusion.webs.upv.es/`

### Main file or '.prm' file

The file '`.prm`' contains the input parameters for the code. To add new values we write in the file an instruction

```
set Name = Option
```

where `Name` is one of the variables listed in Table 1 and `Option` is one of the options contained in that variable. In the case of vectors, these are entered as consecutive numbers separated by spaces.

**Table 1:** Input Parameters for the file '.prm'. The symbol * indicates the default value.

| | Name | *Description* | Command | Options |
|---|---|---|---|---|
| Basic parameter | Dimension | *Problem dimension* | | 1, 2, 3* |
| | Transport_Appr | *Type of neutron transport approximation* | -transport | Diffusion*<br>SPN |
| | N_SPN | *Number 'N' of the equations $SP_N$* | | 1,3*,5 |
| | FE_Degree | *Degree of the polynomials in the finite element method* | -fe_degree | 1,2,3*,4,5 |
| | Energy_Groups | *Number of energy groups* | | 1,2*,...,7 |
| Geometric parameter | Geometry_Type | *Type of geometry of the reactor* | | Rectangular*<br>Hexagonal<br>Unstructured<br>Composed |
| | N_Refinements | *Number of global refinements* | -n_refinements | 0*,...,10 |
| | N_Refs_Radial | *Number of radial refinements* | | 0*,...,10 |
| | N_Refs_Axial | *Number of axial refinements* | | 0*,...,10 |
| | Mesh_Filename | *File where the mesh is described '.msh'* | | |
| | Geometry_Filename | *File where the mesh is described '.xml'* | | |
| | Mesh_Size | *Number of cells in each dimension* | | Vector |
| | Cell_Pitch_x | *Size of the cells in x direction* | | Vector |
| | Cell_Pitch_y | *Size of the cells in y direction* | | Vector |
| | Cell_Pitch_z | *Size of the cells in z direction* | | Vector |
| | Geometry_Points | *First and last cell of each row in a reactor radial plane* | | Vector |
| | Geometry_Matrix | *Matrix with the geometry of a radial plane of the reactor*<br>*0: No material*<br>*1: Fuel*<br>*2: Reflector* | | 0<br>1<br>2 |
| Geometric Parameter | Boundary_Conditions | *Boundary conditions: x1,x2; y1,y2; z1,z2*<br>*0: Zero Flux*<br>*1: Simetry*<br>*2: Albedo*<br>*3: Vacuum* | | 0<br>1<br>2<br>3 |

| | Name | *Description* | Command | Options |
|---|---|---|---|---|
| | Albedo_Factors | *Albedo factors for each group* | | Vector |
| | Triang_Filename | *File where the triangularization is created* | | |
| | Refinement_Model | *Refinement model for the case of compound geometry* | | Local<br>Uniform |
| **Archivos de salida** | Output_Filename | *File where the output is saved* | -out_file | |
| | Output_Flag | *Create a '.vtk' with the output* | -out_flag | true, false* |
| | Print_Grid_Flag | *Plot the mesh in a '.eps' file* | -print_grid | true, false* |
| | Out_Refinements | *Number of refinements per cell in the '.vtk' output* | -n_out_ref | [0,1*,...,10] |
| | Solver_Type | *Method for the eigenvalues computation* | -solver_type | power_it<br>bifpam<br>slepc_2g<br>ks<br>gd<br>newton<br>hybrid |
| | N_Eigenvalues | *Number of eigenvalues to compute* | -n_eigenvalues | 1*,...,50 |
| | EPS_Tolerance | *Relative tolerance for the eigenvalues problem* | -tol_eps | 1e-7*[a] |
| | KSP_Tolerance | *Relative tolerance to solve the linear systems* | -tol_ksp | 1e-12* |
| | Static_KSP_Tolerance | *Use a static tolerance for the linear systems* | -static_ksp_tol | true, false* |
| | Adjoint | *Solve the adjoint problem* | -adjoint | true, false* |
| | Matrix_Free_Type | *Kind of matrix-free methodology* | -matrixfree_type | full_allocated<br>non_diagonal<br>full_matrixfree |
| | P_Init | *Use a multilevel initialization* | -p_init | true*, false |
| **Calculation Method** | XSEC_Type | *Type of cross sections to be read*<br>*XS2G: 2 energy groups by XS*<br>*XSEC: 1 or 2 energy groups by XS**<br>*XLM: More than 2 energy groups by XS (pin level)* | | XS2G<br>XSEC*<br>XLM |
| | XSECS_Filename | *File with the cross sections* | | |

| | Name | *Description* | Command | Options |
|---|---|---|---|---|
| | `Bar_Filename` | *File where the bars movement is defined* | | |
| | `RodCuspingMethod` | *Rod cusping treatment method*<br>*1: Volume Homogenization**<br>*2: Move the whole plane* | | 1*<br>2 |
| Temp. Parameter | | | | |
| Time Data | `Delta_Time` | *Deltatime of the iteration* | 0.01* | |
| | `Time_End` | *Final time of the computation* | | 1* |
| | `Transient` | *Activate transition or dynamics of the system* | `-transient` | true, false* |

a Read as $1 \cdot 10^{-7}$

# 7 Output files

The objective of the code is to know the effective multiplication factor, $k_{\text{eff}}$ (or the following modes) and the neutron flux or neutron power distribution. For the use of the results and their subsequent visualization with other tools, FEMFFUSION creates the following output files:

- **file.out**. It contains the main data of the problem (number of cells, degrees of freedom, ...), the computed eigenvalues, the average neutron flux per cell, and in the case of three-dimensional reactors the axial power profile.
- **file.vtk**. It contains the values of fluxes and neutronic power at the problem nodes for visualization in programs such as PARAVIEW or VISIT.
- **file.eps**. Drawing of the mesh that we have used in the resolution of the problem. This file is recommended only in two-dimensional cases
- **file.log**. Contains information related with the solution of the eigenvalues problem: solver used, preconditioner, number of iterations, number of matrix-vector products, CPU time, ...

# 8 Examples

There are as examples some benchmark problems related with nuclear reactors:

1. **1D:** Example of a one-dimensional 2 cm reactor solved with vacuum boundary conditions. The solution for the $SP_1$, $SP_3$ and $SP_5$ approximations can be seen in Figure 1 along with a reference.

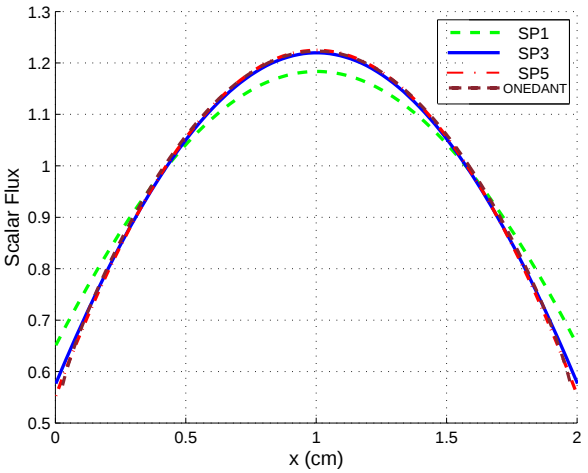
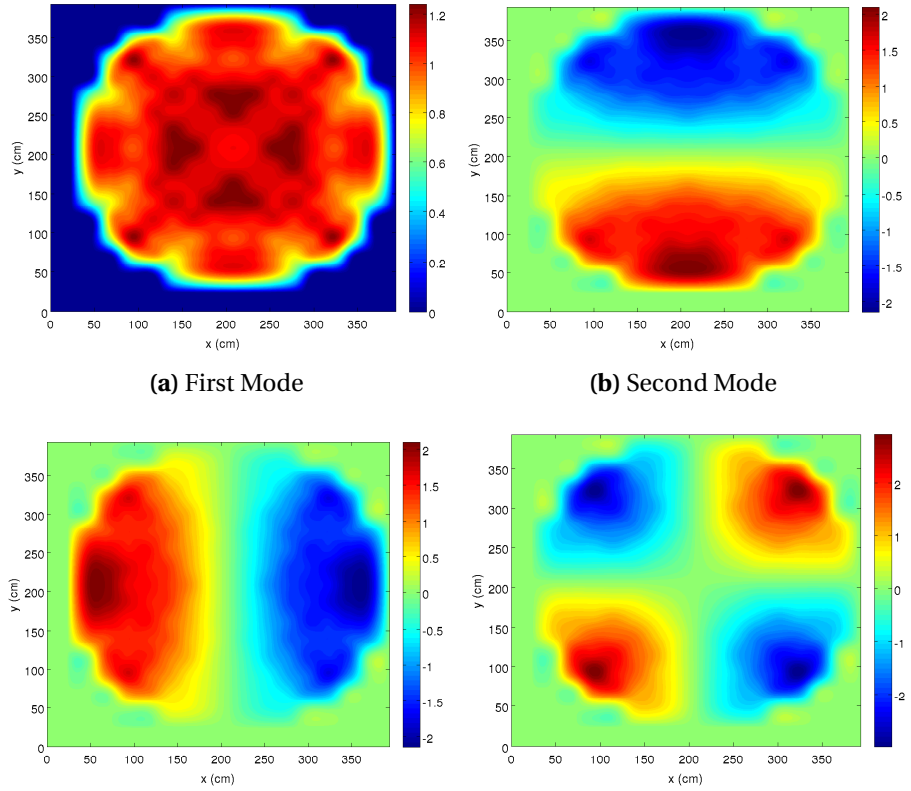



**Figure 1:** Scalar flux for the 1D homogeneous slab.

**2D_BIBLIS:** This is a classical neutron diffusion benchmark with checkerboard features [14]. Solved with SP1, SP3 and SP5 approximations. The solution for the first modes can be seen in Figure 2.

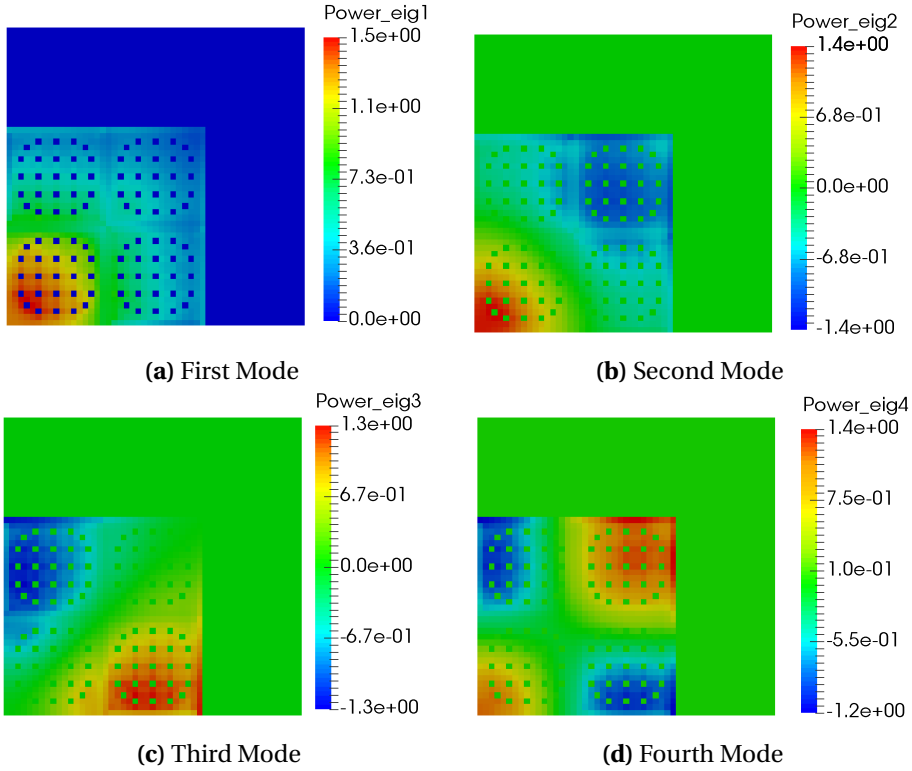



**(a)** First Mode **(b)** Second Mode

**(c)** Third Mode **(d)** Fourth Mode

**Figure 2:** Neutronic power associated with the first modes for BIBLIS 2D reactor.

**2D_C5G7:** NEA's 2D-C5G5 benchmark for deterministic pin-level transport calculations without spatial homogenization. The full definition of the benchmark problem can be found in reference [15].

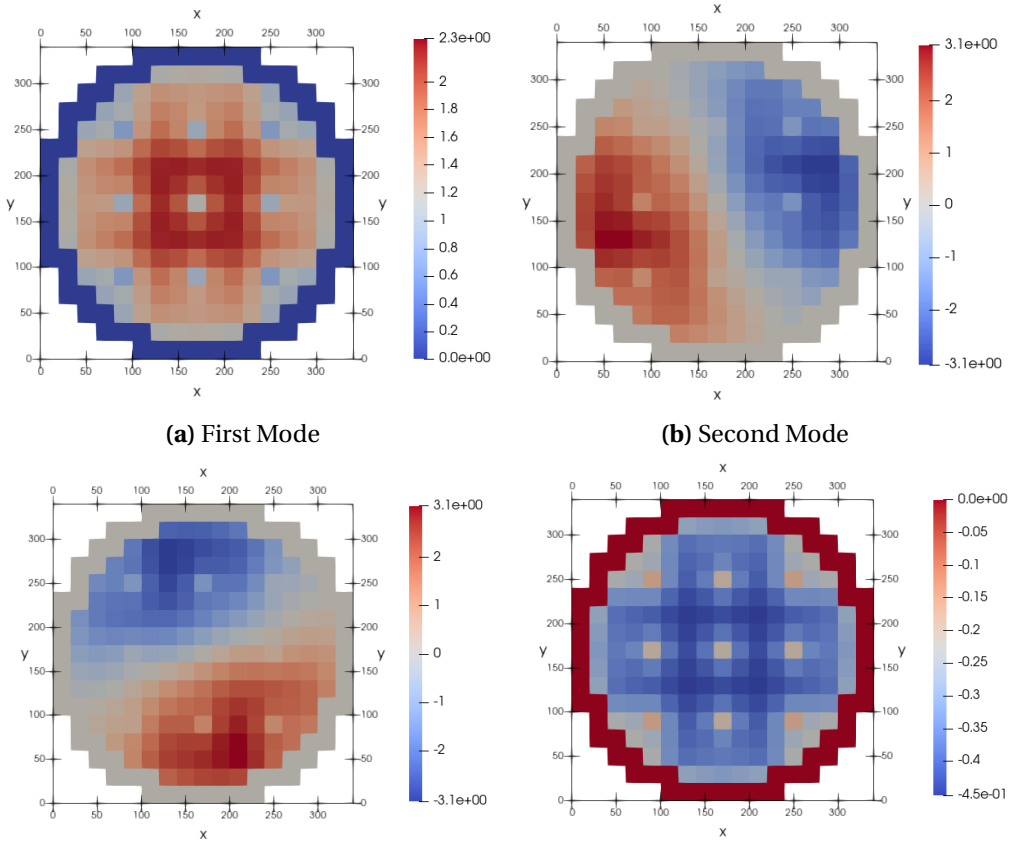



**(a)** First Mode **(b)** Second Mode

**(c)** Third Mode **(d)** Fourth Mode

**Figure 3:** Neutronic power distribution computed with $SP_3$ model for the four modes of the reactor 2D-C5G7.

**3D_IAEA:** A classical 3D benchmark with rectangular geometry. The complete definition can be found in reference [16].




**(a)** First Mode

**(b)** Second Mode

**(c)** Third Mode

**(d)** Fourth Mode

**Figure 4:** Half radial plane of the neutronic power associated with 4 different modes for 3D-IAEA reactor.

**3D_VVER440:** An hexagonal 3D reactor. The complete definition of this problem can be found in reference [17].

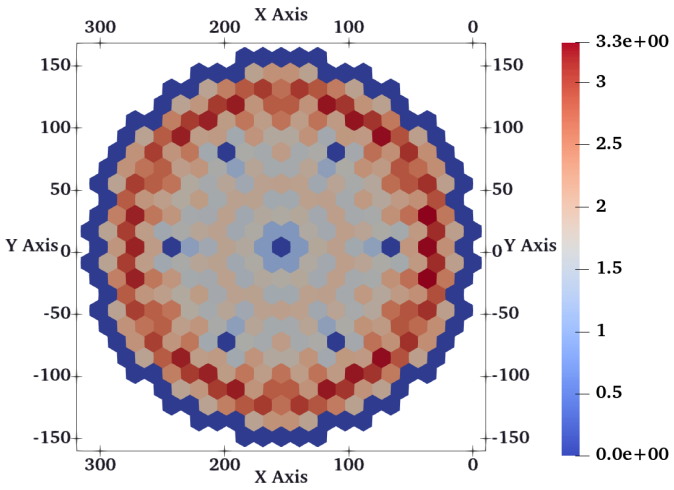



**Figure 5:** Half radial plane of the neutronic power for 3D-VVER440 reactor.

## 9 Related projects

Work on FEMFFUSION is partially supported by the following projects:

- CORTEX project from the Euratom Research and Training Programme 2014-2018 under grant agreement No 754316.
- Grant NEUCONCEPT PDC2021-121667-I00 financed by MCIN/AEI /10.13039/501100011033 and by the European Union Next GenerationEU/PRTR
- Grand PAID-10-19 and SP20180095 by Universitat Politècnica de València.

# Bibliography


[1] W.M. Stacey. *Nuclear reactor physics*. John Wiley & Sons, 2018.

[2] O.C. Zienkiewicz et al. *The finite element method*. Vol. 3. McGraw-hill London, 1977.

[3] W. Bangerth, R. Hartmann, and G. Kanschat. "deal.II – A General-purpose Object-oriented Finite Element Library". In: *ACM Trans. Math. Softw.* 33.4 (2007). ISSN: 0098-3500.

[4] S. Abhyankar et al. "PETSc/TS: A Modern Scalable ODE/DAE Solver Library". In: *arXiv preprint arXiv:1806.01437* (2018).

[5] V. Hernandez, J.E. Roman, and V. Vidal. "SLEPc: A scalable and flexible toolkit for the solution of eigenvalue problems". In: *ACM Transactions on Mathematical Software (TOMS)* 31.3 (2005), pp. 351–362.

[6] A.C. Hindmarsh et al. "SUNDIALS: Suite of nonlinear and differential/algebraic equation solvers". In: *ACM Transactions on Mathematical Software (TOMS)* 31.3 (2005), pp. 363–396.

[7] P. Bastian et al. "Matrix-free multigrid block-preconditioners for higher order Discontinuous Galerkin discretisations". In: *Journal of Computational Physics* (2019).

[8] A. Vidal Ferràndiz. "Development of a finite element method for neutron transport equation approximations". PhD thesis. Universitat Politècnica de València, 2018.

[9] G.W. Stewart. "A Krylov–Schur algorithm for large eigenproblems". In: *SIAM Journal on Matrix Analysis and Applications* 23.3 (2002), pp. 601–614.

[10] R.B. Morgan and D.S. Scott. "Generalizations of Davidson's method for computing eigenvalues of sparse symmetric matrices". In: *SIAM Journal on Scientific and Statistical Computing* 7.3 (1986), pp. 817–825.

[11] A. Carreño et al. "Block hybrid multilevel method to compute the dominant $\lambda$-modes of the neutron diffusion equation". In: *Annals of Nuclear Energy* 121 (2018), pp. 513–524.

[12] Antoni Vidal-Ferràndiz et al. "Moving meshes to solve the time-dependent neutron diffusion equation in hexagonal geometry". In: *Journal of computational and applied mathematics* 291 (2016), pp. 197–208.

[13] C. Geuzaine and J. Remacle. “Gmsh: A 3-D finite element mesh generator with built-in pre-and post-processing facilities”. In: *International journal for numerical methods in engineering* 79.11 (2009), pp. 1309–1331.

[14] EZ Müller and ZJ Weiss. “Benchmarking with the multigroup diffusion high-order response matrix method”. In: *Annals of Nuclear Energy* 18.9 (1991), pp. 535–544.

[15] M. A. Smith, E. E. Lewis, and B. C. Na. *Benchmark on deterministic transport calculations without spatial homogenisation – A 2-D/3-D MOX Fuel Assembly Benchmark (C5G7 MOX Benchmark)*. Tech. rep. NEA/NSC/DOC(2003)16. OECD/NEA, 2003.

[16] American Nuclear Society. *Argonne Code Center: benchmark problem book, ANL-7416(Suppl.2)*. Tech. rep. ANS, 1977. DOI: 10.2172/5037820.

[17] Y.A. Chao and Y.A. Shatilla. “Conformal mapping and hexagonal nodal methods—II: implementation in the ANC-H code”. In: *Nuclear Science and Engineering* 121.2 (1995), pp. 210–225.