

ADDER v.1.1.0 Release Notes

This document retains the release notes for all the ADDER releases. The most recent version is ADDER v1.1.0, released in April 2025.

The Advanced Dimensional Depletion for Engineering of Reactors (ADDER) Software

The Advanced Dimensional Depletion for Engineering of Reactors (ADDER) software performs lifecycle neutronics analysis of a nuclear reactor. This includes performing neutronics calculations with external neutronics solvers, inventory transmutation with an external or internal depletion solver, modifying the neutronics model consistent with fuel management strategies used in a reactor design, and providing a simple interface for critical position/rotation searching.

The ADDER software is written for Python 3.8 or greater [1]. The neutronics interface is currently implemented for MCNP5 version 1.60 [2] and MCNP6 version 6.2 [3]. The current depletion solver is ORIGEN2 (version 2.2) [4] or an internal solver using the Chebyshev Rational Approximation Method (CRAM) [5].

Software Installation

The generalized process for installing ADDER is documented in Section 1.1 of the ADDER User's Guide [6] and any modifications to support the local environments.

Software Verification

The ADDER test suite is fully automated using the widely used testing framework, pytest [7]. This test suite is executed after ADDER installation by running the `pytest` command in the ‘tests/’ directory of the ADDER distribution. This test suite will automatically execute the tests, compare results, and report the number of passing and failing tests. All tests shall pass.

When using this test suite, the ORIGEN2, MCNP, MPIRUN executables to be used for testing shall be accessible in the environment as the name ‘origin2.EXE’, ‘mcnp.EXE’, and ‘mpirun’. This may be accomplished through environment variable modifications and symbolic links as needed. Further, the MPI libraries required for ‘mpirun’ shall also be accessible via the execution environment. This requirement only applies for automated testing and not for typical ADDER usage. For completeness, this test suite should be repeated for every MPI, MCNP, and ORIGEN2 combination to be used with ADDER.

Software Installation and Execution

General installation, user input, and execution instructions are thoroughly documented in the ADDER User Guide [6].

Contact Information

For technical questions, bug reports, and other inquiries regarding ADDER, please contact the development team at adder@anl.gov

Version Release Notes

[ADDER v1.1.0 \(April 2025, current\)](#)

ADDER v1.1.0 is the third release of ADDER and introduces several new capabilities, in addition to resolving the known bugs and limitations that have been identified since the previous release. Users should refer to the ADDER v1.1.0 User Guide [6] for additional information and background about the changes listed in these release notes.

New and Improved Features

The following features were implemented into ADDER v1.1.0 and are detailed in the most recent version of the ADDER v1.1.0 User Guide [6]:

1. Optimization of parallel depletion algorithm. Parallel resources are now more effectively utilized to improve computing time of depletion steps. An optimal depletion chunksize is now calculated by ADDER to maximize the use of available parallel resources, if not explicitly set by users in the ADDER input file.
2. Addition of calculation of power density (in W/cm^3), fission density (in fissions/ cm^3), and burnup (in MWd) for every depleting region at every depletion step during the irradiation history, as defined in the ADDER input file.
3. New tally feature that allows users to define custom tallies in the neutronics MCNP (currently only MCNP5 v1.60 and MCNP6.2 are supported) that are carried throughout the ADDER calculation and are handled through fuel management operations. This can be used to associate a tally to a reactor component (defined by either a material or a universe) and “follow” that tally when the component is moved to a different location inside the core, changes orientation (e.g., is rotated/flipped), or is moved to storage.
4. `[[[geometry_search]]]` operations on control groups can now be performed with respect to different references. The last position of the control group can be used as a reference as well as the original position from the initial neutronics (e.g., MCNP) input file (default).
5. `[[[transform]]]` operations can now be reset to ensure that the displacement and rotation of the component is relative to its original position in the initial neutronics (e.g., input file) rather than its last position (that remains the default).
6. `[[[transform]]]` operations can no longer be performed on storage and supply universes. Users are limited to transform universes that are in-core. This change is intended to limit unintentional errors in which users could apply transform operations on out-of-core universes believing that the change was actually applied to an in-core universe with a similar name.

Documentation Changes

The ADDER v1.1.0 User Guide [6] has been significantly revamped. The following additional Sections were added:

1. Added *Section 1.5 ADDER Flowchart*, which includes a graphic depiction of ADDER’s execution and order of operations.
2. Added *Section 2 Depletion Methodology* and its subsections. This section provides the theoretical and methodological background for the ADDER depletion algorithm. This

includes information about how the cross-section data is updated during the ADDER execution, if requested by the user, as well as the methodology used to build the depletion matrix and solve the matrix exponential if the internal CRAM depletion solver is selected. Theoretical background for the scaling constant used for flux normalization is also provided. Finally, the methodology for the newly-implemented calculation of the power density, fission density, and burnup is discussed.

3. The content of the Input File Format section that was in the ADDER v.1.0.1 (previous release) User Guide has been subdivided into two different sections, that have also been expanded:
 - a. *Section 3 Fuel Management Methodology*, which includes the definition of materials, universes, components, and control groups, as relevant to the ADDER implementation.
 - b. *Section 4 Input File Format*, which is now more focused on the syntax and sections of the ADDER input file. The structure of this section has been changed to more clearly separate each operation to have an independent subsection. Additionally, several subsections were added:
 - i. *Section 4.5 Tallies*, details the newly implemented tally feature and its input syntax
 - ii. *Section 4.8 Examples of Fuel Management Operations*, provides a few simple examples of various fuel management operations and includes graphic representations of them.
4. Modified Section 5.2 *Output File Format* to include information about:
 - a. Data in the HDF5 output file as part of the new tally feature implementation.
 - b. Data in the HDF5 output pertaining to control groups. It should be noted that the data was always included in the HDF5 output data; however, the User Guide did not provide any details about it.
5. Modified Section 7.1.2 *MCNP User Tallies* to provide additional guidance and details about the tally feature implementation and its implications in preparing the initial MCNP input file required for performing ADDER calculations.
6. An additional subsection (*Section 7.1.4 Equivalence of ADDER-calculated Power Density to MCNP f4 and f7 Tallies*) discusses the equivalence between the calculation of the power density performed within ADDER and the MCNP tally structures normally used to calculate the same quantity.
7. An additional subsection (*Section 7.1.5 Transforms, Geometry Sweeps, and Critical Geometry Searches*) discusses the methodology behind geometry search operations, including flowchart representations of the iteration scheme and convergence criteria.

Bug Fixes

ADDER v1.1.0 solves the following known bugs with ADDER v1.0.1:

1. ADDER's implementation of the string method "strip" was incorrect and has since been fixed. This error could lead to incorrect processing of the following input:
 - a. FM cards defined in MCNP inputs which included parentheses
 - b. MSR feed materials defined in ADDER inputs which included parentheses or any of the characters in the following string: "materials_"

2. Incorrect rotation angle when using the `[[[transform]]]` operation to rotate a component by a given angle using either one or more of the *yaw*, *pitch*, *roll*, and *matrix* parameters. When these operations were performed to rotate a component, the component's transform cards in the ADDER-generated MCNP input files were inconsistent with the angles or rotation matrix provided in the ADDER input file, due to a different notation in the indexes of MCNP's and ADDER's rotation matrices. For example, if (x,y,z) is the original coordinate system and (x',y',z') is the transformed coordinate system, the matrix element in row 1 column 2 refers to the cosine of the angle between axes y and x' in MCNP, whereas it is the cosine of the angle between axes x and y' in ADDER. This discrepancy was not considered and algebraic operations between rotation matrices were inconsistent. Now, ADDER allows the use of both notations when using the *matrix* parameter and correctly makes sure that the internal calculations are performed using a consistent notation, irrespective of which notation and parameters (*yaw*, *pitch*, *roll*, *matrix*) were used to define the `[[[transform]]]` operation.
3. ADDER now correctly processes capitalized MCNP cards' jump features (i.e., the uppercase "J" letter) correctly. Before the fix, ADDER could also process lowercase jump feature (i.e., the lowercase "j" letter).
4. ADDER now correctly processes cards in the MCNP input file that are continued over multiple lines by use of one or multiple ampersands (&), which were not correctly recognized in the previous ADDER release.
5. ADDER is now able to process "keyword-equal-value" pairs with spaces in between (e.g., "vol = 1" attribute for a cell card).
6. ADDER now correctly accounts for MCNP input files that have the same material with a different density type (i.e., atom or mass) defined for different cells. For example, if two cells in the MCNP input file are assigned the same material, it is now possible to define a mass density value for one and an atom density value for the other without incurring in an ADDER error.
7. ADDER now correctly reports the value of the k_{eff} standard deviation in the HDF5 output file at each relevant step rather than writing the k_{eff} value again in the standard deviation field.
8. ADDER can now process MCNP input files with tally multiplier cards (FM cards) that include an attenuator. The attenuator is denoted by a -1 following the constant multiplier, and it precedes the material ID. Before the fix, ADDER would not recognize the -1 value as a valid input.
9. `[[[geometry_search]]]` operations following `[[[transform]]]` operations now correctly use the last position of the control group as identified by the `[[[transform]]]` operation, rather than the second-to-last.
10. Transformations were associated with the *location* that contained universes rather than the constituents (e.g., cells) of the universe itself. As such, when a component was moved out of a location (e.g., via a `[[[shuffle]]]` operation) after being transformed, that location retained the same transform. This required users to "de-transform" universe components before a different universe could replace them. Now, the displacement and rotation associated with `[[[transform]]]` operations are associated to universes themselves. As such, when a transformed universe is shuffled into a different location, it retains its

transformed orientation/displacement, whereas the universe that replaces it inherits the original orientation/displacement as per the base neutronics (e.g., MCNP) input file. Note that this updated behavior is not backwards compatible.

11. *[[[geometry_search]]]* operations no longer fail or return nonsensical values if not enough active cycles are run in the neutronics solver MCNP. This behavior would happen, for example, for low statistics runs in which ADDER would not set a high enough number of active cycles for the MCNP simulation and valid values of k_{eff} would not be calculated. Now, ADDER always ensures that a minimum number of active cycles is run.
12. ADDER no longer calculates a 0 value for the recoverable energy per fission, Q_{rec} , if the flux is zero in one or more depleting regions.
13. ADDER is now correctly able to define aliases for all universes. Previously, ADDER would not recognize universes with number that was not also the ID for a material in the same neutronics MCNP input file.
14. ADDER now correctly assigns the case and operation IDs (sequentially and without skips) when aliases are used in fuel management operations (e.g., *[[[shuffle]]]*).
15. ADDER now correctly sets the status of universes that are moved from storage to in-core and correctly depletes them.
16. ADDER now exits with an error message if the same name is assigned to multiple components or aliases in the ADDER input, including checks to avoid that the same name corresponds to a name automatically generated by ADDER.
17. ADDER now correctly parses certain isotope ZAIDs that were erroneously identified as metastable state indexes larger than 1.
18. Resolved bug that would cause fluxes and cross-sections to not be correctly calculated for materials in storage and supply universes, which would lead to erroneous depletion calculations.
19. Resolved bug that would cause materials with certain isotopes (e.g., Li-6) to deplete incorrectly due to conflicting reaction identifiers between ORIGEN2 standard reactions and ENDF reactions. For example, the reaction of Li-6 with a neutron that produces an atom of He-4 and H-3 is correctly identified by both the (n,t) and (n,alpha) reaction. The former is consistent with the ENDF convention (lighter particle in parentheses), while the latter is used in ORIGEN2 and its libraries.

ADDER v1.0.1 (May 2022)

ADDER Version 1.0.1 is the second release of ADDER. The following changes have been made since ADDER v1.0.0 was released:

1. The MIT Open Source Software license is now distributed with the ADDER Software.
2. ADDER v1.0.0 may unintentionally deplete materials with incorrect fluxes because of an error in the order that it processed MCNP tallies. ADDER v1.0.1 corrects this error.
3. ADDER v1.0.0 did not provide an error to the user when an element or isotope is present in the initial neutronics model that does not have neutronics cross sections available in the neutronics library file. ADDER v1.0.1 now provides such an error.
4. If an isotope is present in the neutronics model but is not present in the depletion library, then ADDER should mark that isotope as non-depleting. ADDER v1.0.0 only did this for the first isotope encountered in a depleting material and all others were marked as depleting. ADDER v1.0.1 correctly applies the same non-depleting status to all instances of such isotopes in depleting materials.
5. ADDER v1.0.0 crashes when the “all_depleting” value is used as an argument to the “write_depletion_lib” operation. ADDER v1.0.1 has (1) renamed this value to “all_modified” to more accurately reflect that only non-default libraries are written and (2) now no longer crashes when this is used.
6. ADDER v1.0.1 now raises an error when a user provides a material that is both: (1) marked as depleting based on the ADDER input file, and (2) the neutronics inputs file does not have a default neutronics cross section library defined.
7. ADDER v1.0.0 would erroneously not apply the non_depleting_isotopes input to the model. In ADDER v1.0.1, isotopes explicitly marked by the user as “non_depleting_isotopes” are now read correctly and these isotopes not depleted.
8. ADDER v1.0.1 provides a user-understandable warning to the log file when user-supplied MCNP inputs contain zero mass and/or atom fractions on material cards.
9. ADDER v1.0.1 now provides a user-facing option where the user can specify if all isotopes explicitly entered by the user in the initial neutronics model are intended to be included in all time steps of the model regardless of their reactivity importance and the value of the “neutronics_reactivity_threshold” parameter. This “apply_reactivity_threshold_to_initial_inventory” input parameter can be set either globally or on a per-material basis.
10. ADDER v1.0.0 applied incorrect logic for the reactivity threshold implementation in the case of non-fissionable materials. This logic is now corrected in ADDER v1.0.1 by using a method which is only based on the fraction of neutron absorption of each isotope in that material. The method for fissionable materials is unchanged.
11. ADDER v1.0.1 now raises an error whenever an isotope is included multiple times in a material definition, whether they be defined with neutron libraries (e.g., nlibs) or otherwise.
12. ADDER v1.0.0 would incorrectly parse valid real numbers in MCNP input files (e.g., -1+1 to represent -10) and exponential notation used for valid integer input (e.g., 1.0E6 for the integer 1,000,000). ADDER v1.0.1 now correctly processes these valid MCNP input formats.
13. ADDER v1.0.0 would incorrectly not ignore comments in the directory section of an MCNP xsdir file, potentially leading to an error. ADDER v1.0.1 now correctly recognizes and ignores these comments.
14. Certain cards in MCNP input may contain single or multiple *jump* entries. ADDER can already handle such jump entries, however, ADDER v1.0.0 was missing this *jump* handling for MCNP’s

kcode and print cards. ADDER v1.0.1 will correctly handle such kcode and prdmp cards with these jumps.

15. ADDER v1.0.0 execution would fail when performing a stochastic volume calculation on models initially provided by the user with lost, run, or nps cards. ADDER v1.0.1 now executes such cases successfully by removing these cards and replacing when done with the stochastic volume calculation.
16. ADDER v1.0.1 will shorten the title provided by the user in the initial MCNP input file to make sure that the total length of the final title for each depletion step, including the ADDER-generated information about the step, doesn't exceed 80 characters. ADDER v1.0.0 would not perform such truncations nor provide any warning.
17. ADDER v1.0.0 and its post-processing adder_extract_materials.py script incorrectly extracts the *num_copies* field from material data in ADDER HDF5 files such that a failure was encountered if a material is processed which has been copied more than twice during fuel management operations. ADDER v1.0.1 now correctly processes this parameter and avoids such failure.
18. ADDER automatically creates clones of cells filled with depleting materials that are present in multiple locations in a model. When this is done, ADDER cannot determine how the volume of the divided cell can change and in this case the undivided cell volumes will remain. ADDER v1.0.1 now provides a warning in such a case so the user can be aware that they should check the volumes and potentially run a stochastic volume calculation.
19. MCNP cells can be defined using the complement operator (#n) where n is a previously defined cell number. If cell n is cloned and given a new id such as n+1, ADDER v1.0.0 would erroneously not update the id of such a complement operator to be #n+1 as necessary for creating copies of shuffled universes. In ADDER v1.0.1, if a complement operator is used in a shuffled universe, then a warning will be provided telling the user that they should check that the region definitions are correct.
20. Users do not have to provide material volumes in the ADDER/MCNP input for depleting materials if these volumes are automatically calculated by MCNP. ADDER v1.0.0 will crash if a user does not give a volume for a material and then shuffles that material prior to an MCNP calculation. Such a situation may be perfectly acceptable and therefore this crash will be corrected and instead a warning relayed explaining that a shuffled material cannot be checked for consistent volumes with items in the chain of shuffled materials.

ADDER v1.0.0 (April 2021)

Version 1.0.0 is the first release of ADDER, and therefore there are no modifications to report. However, a synopsis of features is provided below. Further information is present in the ADDER User Guide [9].

The inputs to ADDER are: (1) a neutronics input file; (2) an ADDER input file; and (3) a depletion data library. The neutronics code input file contains no ADDER-specific information. This approach allows for the trivial addition of both depletion and fuel management capabilities to existing neutronics models. The ADDER input file provides information specific to ADDER, including the execution options, the power/flux histories, and fuel management operations. Finally, the depletion data library is an HDF5 file containing the decay data, available transmutation reaction channels, and the cross-sections for the transmutation reaction channels. The ADDER capabilities related to depletion, fuel management, and critical searches are described briefly below. Further information is available in the user guide.

Depletion

ADDER performs point depletion on the user-specified fuel and non-fuel depleting material regions using a power-normalized flux and (optionally) region-specific cross sections calculated by the neutronics code. The power normalization is performed using the flux scaling factor (c) formulation and recoverable energy from fission (Q_{rec}) correlation provided in Equations (1) and (2), respectively. These are consistent with the approach taken in ORIGEN2.

$$c = \frac{\nu P}{k_{eff} Q_{rec}} \quad (1)$$

$$Q_{rec} = 0.00129927 Z^2 \sqrt{A} + 33.12 \text{ MeV} \quad (2)$$

The user can select to perform either constant flux (i.e., predictor) method or the Constant Extrapolation/Constant Midpoint (CE/CM) method for time stepping [10]. These methods provide linear and quadratic time step convergence, respectively. These depletion computations are performed in parallel to minimize the computation expense in large reactor models with many depleting regions.

Depletion data is obtained from ADDER's depletion library in HDF5 format. This library initially includes group-wise cross sections with an arbitrary group structure. ADDER can deplete using these cross sections directly (as in legacy ORIGEN2 depletion computations), or the neutronics code can be used to provide flux spectrum-weighted cross sections for each region, isotope, and their reaction channels (as in OpenMC [11] and MCNP version 6.2). In either case, a depletion library is necessary to provide decay data and the available reaction channels.

Some users will utilize ORIGEN2-based libraries as the source of the depletion library data. The user should be aware of the discrepancies in that data compared with more recent sources. Specifically: (1) the number of isotopes are limited, (2) the number of decay and neutron-induced reaction channels available for each isotope are limited, (3) the base decay and cross section data is inconsistent with more recent evaluations, and (4) the flux-spectra used to collapse the cross sections may be different than in the problem at hand. Items (3) and (4) in this list can be resolved (for neutron-induced cross sections) by requesting ADDER to extract flux-weighted cross sections from the MCNP model at times of interest; this is enabled with the `use_depletion_library_xs` parameter discussed in the user guide. The remaining items cannot be improved on without updating the libraries. In any case ADDER will automatically add the required tallies to the neutronics solver inputs that it creates without requiring any user definition.

ADDER can be used in a manner similar to the VESTA [12] fuel management tool because the depletion library cross sections can be provided using a group structure with as many groups as desired. In this case, if a depletion library is defined with an ultra-fine group depletion library, and the user input instructs ADDER to use the depletion library cross sections, then ADDER will use the neutronics code to compute the region-wise flux spectra but not the more costly region- and isotope-wise reaction rates. This may be useful for a reduction of the computation time, however, it may increase memory usage.

The depletion library HDF5 file discussed above can be created manually or converted from other sources. Scripts are available with ADDER which perform the conversion for existing ORIGEN2 libraries.

As stated, the depletion solver can either be ORIGEN2.2 or an internal CRAM solver. The applicable methods are thoroughly discussed in [4] and [5, 10], respectively. Note, however, that ADDER only allows the 16th and 48th-order CRAM approaches discussed in [5]. Both solvers can be used with predictor or CE/CM time-stepping.

Fuel Management

ADDER provides the ability to shuffle and rotate/flip components in the neutronics model (e.g., elements and/or assemblies) between depletion steps. ADDER can also apply a coordinate transform to any surface, universe, or cell in the model. These components can also be discharged to storage, reloaded from storage, and replaced with fresh components (referred to as “supply”) as needed. In-core components undergo neutron flux-induced irradiation and radioactive decay. Components in storage are allowed to undergo radioactive decay for a duration consistent with the in-core operation period, but no neutron flux is applied. Supply components do not decay or transmute.

ADDER can perform fuel management through the following means:

shuffle: A shuffle operation moves a component within a reactor. This can be done with either materials or universes. The input for a shuffle operation includes a *moves* parameter. This is a list of components to be moved in their order, and must include at least 2 entries. Specifics of this operation are best explained by example. If the *moves* parameter is “1, 2, 3”, then ADDER will move component 1 to the location of component 2, component 2 to the location of component 3, and component 3 to the location of component 1. If component 1 is either a supply or storage component, then the final move of component 3 will be to storage as well. Further, if component 1 is a supply component, then a copy of component 1 will be introduced and named “1[1]” (or “1[2]” if this is the second copy, and so on).

revolve: A revolve operation is a shuffle that mimics a rotation or a flip. It should be noted that a revolve operation does not actually change the orientation of components, it simply rearranges their locations to represent a rotation or flip. For example, if the revolve set parameter is provided the following 1x3x3 (z, y, x) matrix of components:

1 2 3
4 5 6
7 8 9,

and the user requests an *xy_degrees* rotation of 90 degrees, the resultant arrangement is effectively the following:

3 6 9

The revolve command also includes the ability to perform a *z-flip*. A *z-flip* is similar to the example above except the axial slices of a matrix will be flipped instead of the x-y plane rotated. The revolve operation can only be performed on material or universe components.

transform: A transform operation can be performed on surfaces, cells, or universes. A transform specifically applies a coordinate transform card with the specified displacement and rotation to the given set of components. If a component already has a coordinate transform applied, then ADDER will apply the *transform*'s operation afterwards (i.e., the new displacement vector is added to the old and the new rotation matrix is left-multiplied by the old matrix).

MCNP limits the number of coordinate transformation cards (TR cards) to 999. ADDER will combine equivalent TR cards to efficiently utilize this limited resource. However, this merging is performed at the end of each transform operation. The user should be aware of this limitation when developing their models.

[Criticality Search](#)

ADDER can use the coordinate transform capability to perform a criticality search for the user. This algorithm is explained in the ADDER User Guide [9], but is wholly consistent with the Regula-Falsi approach to stochastic searches described in [13]. This feature can only be used with groups of neutronics model surfaces, referred to in the ADDER user guide and input file as 'control groups'.

[Molten Salt Reactor Analysis Capability](#)

ADDER contains a flowing-fuel depletion capability for the analysis of molten salt reactors (MSRs). While tested equivalently to the solid-fuel portions of ADDER, the MSR capability was not reviewed with the same scrutiny as the typical solid-fuel capability. Therefore, this functionality should not be used for QA-compliant work. If a user requests this functionality, a warning will be written to the screen stating that the calculation will exercise non-QA compliant portions of the software.

References

1. Python Software Foundation, *Python Language Reference, version 3.7*. Available at <https://www.python.org>.
2. X-5 Monte Carlo Team, *MCNP- A General Monte Carlo N-Particle Transport Code, Version 5, Volume I* (LA-UR-03-1987), *Volume II* (LA-CP-03-0245), *Volume III* (LA-CP-03-0284), Los Alamos National Laboratory, Los Alamos, USA, Apr. 2003 (Revised Feb. 2008).
3. C.J. Werner (editor), *MCNP User's Manual - Code Version 6.2*, LA-UR-17-29981, Los Alamos National Laboratory, Los Alamos, USA, Oct. 2017.
4. A.G. Croff, *A User's Manual for the ORIGEN2 Computer Code*, ORNL/TM-7175, Oak Ridge National Laboratory, Oak Ridge, USA, Jul. 1980.
5. M. Pusa, "Higher-Order Chebyshev Rational Approximation Method and Application to Burnup Equations", Nucl. Sci. Eng., 182:3, 297-318 (2016).
6. V. Mascolino, K. Anderson, C. Castagna, M. Elsawi, and E. Wilson, *ADDER v1.1.0 User Guide*, ANL/RTR/TM-24/25, Argonne National Laboratory, Lemont, USA, Apr. 2025.
7. Krekel et al., <https://github.com/pytest-dev/pytest>.
8. K. Anderson, V. Mascolino, and A. G. Nelson, User Guide to the Advanced Dimensional Depletion for Engineering of Reactors (ADDER), Release Version 1.0.1, ANL/RTR/TM-21/8 Rev. 1, Argonne National Laboratory, Lemont, USA, Apr. 2022.
9. A. G. Nelson, User Guide to the Advanced Dimensional Depletion for Engineering of Reactors (ADDER), Release Version 1.0.0, ANL/RTR/TM-21/8, Argonne National Laboratory, Lemont, USA, Apr. 2021.
10. C. Josey, "Development and analysis of high order neutron transport-depletion coupling algorithms", Diss. Massachusetts Institute of Technology (2017).
11. P. K. Romano, N. E. Horelik, B. R. Herman, A. G. Nelson, B. Forget, K. Smith, "OpenMC: A State-of-the-Art Monte Carlo Code for Research and Development," Ann. Nucl. Energy, 82, 90–97 (2015).
12. W. Haeck, B. Cochet, L. Aguiar, "Monte Carlo depletion calculations using VESTA 2.1 new features and perspectives", PHYSOR 2012 Advances in Reactor Physics , Knoxville, USA, Apr. 15-20, 2012.
13. D.F. Gill, B.R. Nease, and D.P. Griesheimer, "Movable geometry and eigenvalue search capability in the MC21 Monte Carlo Code," Proc. Intl. Conf. on Mathematics and Computational Methods Applied to Nuclear Science and Engineering, Sun Valley, USA, May 5-9, 2013.