Proceedings of the 2017 25th International Conference on Nuclear Engineering
ICONE25
July 2-6, 2017, Shanghai

ICONE25-66584

TOWARD AN OPEN-SOURCE NEUTRONICS CODE FOR CIRCULATING-FUEL REACTORS

Julien de Troullioud de Lanversin
Princeton University
Princeton, NJ, U.S.A.

Alexander Glaser
Princeton University
Princeton, NJ, U.S.A.

Malte Göttsche
Princeton University
Princeton, NJ, U.S.A.

ABSTRACT

In circulating fuel reactors, such as the Molten Salt Reactor, the fuel circulates throughout the reactor instead of being immobile as in solid fuel reactors. The vast majority of nuclear simulation codes are primarily designed to simulate solid fuel reactors. Hence, many features unique to circulating fuel reactors, such as fuel injection and removal, cannot be properly modeled with these codes. The work presented here focuses on developing a numerical simulation package that can effectively and accurately model these reactors. This package consists of the coupling of the Monte Carlo particle transport code OpenMC with a modified version of ORIGEN-S, and uses a novel algorithm that calculates the optimal fuel injection and removal schemes for such reactors to achieve certain conditions such as a stable reactivity. We demonstrate our code's accuracy by benchmarking the coupling module with the MCODE coupling code, and by simulating the operation of the ORNL Denatured Molten Salt Reactor using the coupling and fuel injection modules. The resulting fuel injection scheme is in agreement with the original study of that reactor while offering a much finer resolution for the injection scheme over time. This work is part of a broader project to develop an open-source neutronics code for circulating fuel reactors that will couple OpenMC with an in-house open-source depletion module.

INTRODUCTION

Molten Salt Reactors (MSRs) differ from the more classical and widely used solid-fuel reactors by the use of a liquid fuel that circulates inside and outside the reactor core. Research on this type of reactor was underway until the 1970s to assess its potential for commercial use in the United States. However, a preference for plutonium-fueled fast-neutron reactors and a lack of government funding ended research efforts at that time. Recently, with the call for advanced types of nuclear reactors (Generation IV) capable of meeting challenges such as enhanced safety and security, environmental protection, and economic viability, the MSR is considered a serious candidate technology again.

Conceptual MSR designs often envision a graphite matrix that acts as a static moderator through which the molten salt fuel flows. Because the fuels typically contain light elements that slow down neutrons, MSRs are more suited to be operated at thermal energy but fast systems have been proposed as well.

The liquid nature of the fuel in a MSR allows for a whole array of new features and processes that are very interesting from the neutronics point of view. For example, it is possible to inject and remove material on a continuous basis in order to keep the core critical, to achieve breeding, or to maintain a certain chemical balance in the liquid. Simple gas bubbling removes gaseous fission products, such as the neutron poison xenon-135. Elements that remain liquid in the fuel are either kept in the fuel or can be extracted with more complex chemical separation, which would be done outside the main loop in dedicated facilities. The possibility of continuously removing neutron poisons enables higher burnup and thus improves the resource utilization of MSRs.

While promising, this type of reactor also presents specific challenges: The fact that the heat carrier is the highly radioactive fuel itself complicates handling the cooling circuit. Since the fuel is liquid and circulating, a separate reactor structure is needed. Consequently, the irradiation exposure and component longevity must be well understood before this type of reactor can be considered for commercial use. Lastly, if MSRs are used as part of a thorium fuel cycle, they would continuously breed uranium-233 (from thorium-232), which can be used for nuclear weapons, thereby posing a proliferation concern similar to plutonium buildup in uranium-fueled reactors.

In order to address these proliferation concerns, the Denatured Molten Salt Reactor (DMSR) has been proposed and studied in a report by the Oak Ridge National Laboratory (ORNL) [1]. The main idea is to maintain the in-core uranium denatured (< 12% uranium-233 or < 20% uranium-235) so that the fuel is considered non-weapon-usable at any burnup. In addition, to demonstrate the viability of this type of MSR, the report also examined the performance of the DMSR without removal of neutron poisons and other fission products, which would otherwise need to be chemically separated and extracted in a normal MSR. Hence, this greatly simplifies the operation and improves the proliferation-resistance of the DMSR compared to other MSR designs.

In order to further examine the properties and operation of MSRs, including injection/removal schemes, there is a need for computational tools to accurately model the processes that are unique for this reactor type. In particular, the circulation of the liquid fuel (in-core and out-of-core) and the continuous transfer of elements, i.e., injection into and extraction from the fuel, are not the focus of existing codes, which were originally written primarily to simulate solid-fuel reactors.

The work presented in this paper focuses on developing a numerical simulation package that can model reactors with circulating fuels much more accurately than existing neutronics codes, which provide limited "add-on" options to mimic some of the characteristics of circulating fuel reactors. We develop a specific algorithm that calculates optimal fuel injection and removal rates. Compared to previously developed algorithms, it has the advantage of calculating injection and removal at every simulated time step, relying solely on an analytical derivation while vastly improving performance and speed. The algorithm is integrated into ORIGEN-S from the SCALE6.2 release [2]. This modified version of ORIGEN-S is coupled with OpenMC, an open-source transport code [3].

CONCEPTS AND ALGORITHMS

Modelling the online fuel transfer

Fuel depletion in a reactor is governed by the Bateman depletion equation [4]

$$\frac{dN_i}{dt} = \sum_{k\neq i}^{n} \lambda_{k\to i} N_k - \lambda_i N_i \quad (1)$$

where N_i is the concentration of nuclide i, λ_i includes the disintegration constant as well as the neutron absorption reaction rates of nuclide i, and $\lambda_{k\to i}$ includes the disintegration constant as well as the neutron-induced reaction rate of a parent nuclide k that lead to the formation of nuclide i. It is important to note that the term involving the nuclides k is a non-homogenous term, i.e. a term involving other nuclide densities than N_i. Each nuclide i in the reactor has its own version of (1) and the resulting system of equations can be lumped into a single matrix differential equation. Usually, this matrix equation is then solved with a matrix exponential.

There are multiple ways to model fuel injection and removal inside of (1). The simplest but also perhaps the crudest method is to periodically update the density of some nuclides [5-8]. This will produce discontinuous isotopic and criticality evolution patterns. A more accurate method would be to add production-rate and removal-rate terms in (1) [9, 10]. This way of representing fuel transfer is very close to the actual reactor operation and has the advantage of enabling continuous injection or removal. Well-established depletion codes such as ORIGEN use this kind of representation for fuel element transfer. However, adding these production or destruction rates in equation (1) tends to be cumbersome when one seeks a matrix exponential solution to the system of depletion equations. Finally, it is also possible to represent element's transfer via the addition of an artificial disintegration constant λ_{art}. This method has the advantage of not bringing any non-homogeneous terms to (1), and it has already been implemented successfully in several works [11-13]. This is the method we choose for this work.

Calculating the correct λ_{art} to obtain the desired reactor conditions (criticality, isotopic ratio, or chemical balance) remains the main challenge for any neutronics code focusing on fuel injection and removal. Previous implementations of this method had to compute λ_{art} with hundreds of iterations [14].

It is indeed not possible to find an exact closed-form relation between the fuel injection/removal rate and the desired reactor conditions based on (1). A closed-form expression can, however, be found when using an approximation to (1),

$$\frac{dN_i}{dt} = \lambda N_i \quad (2)$$

where λ is calculated so that solving this equation yields the exact same density for nuclide i as with equation (1) after a specific period of time. We note that equation (2), unlike (1), is homogeneous. Since the solution of this equation is a pure exponential, we will call this approximation the exponential approximation. The main idea of this work is that, instead of adding λ_{art} in (1) and to search for the right value of λ_{art} through iterations, we will add it to the exponential approximation of (1) which will enable to find a close form relation. In other words, we will use the following equation:

$$\frac{dN_i}{dt} = (\lambda + \lambda_{art})N_i \quad (3)$$

Once one obtains a value for λ_{art}, it can be added to the exact depletion equation (1). As long as (2) is a close approximation to (1), the value one finds for λ_{art} will be adequate to bring the system to the targeted condition. It happens that the fissile materials one may inject or remove (uranium-235,

uranium-238, plutonium) have an isotopic evolution that is rather close to an exponential.

The algorithm

In this work, online fuel transfer is implemented to ensure the criticality of the reactor. The algorithm finds the optimal λ_{art} for the Bateman equation to maintain criticality close to one. In principle, the algorithm presented here could also be adapted for other objectives, such as chemical balance or uranium isotopic composition in the fuel.

The first step in developing the algorithm is to introduce a model for the ensemble of nuclides and the criticality of the system. For this purpose, we divide the system of nuclides in the fuel into two subsystems S_1 and S_2. S_1 contains an ensemble of nuclides so that the criticality $k_1$of S_1 (the rate of neutron production divided by the rate of neutron absorption for the nuclides of S_1) is higher than the criticality k_{inf} of the whole system. S_1 includes the fissile materials to be removed or to be injected. S_2 consists of all other nuclides. We then have $k_2 < k_{inf} < k_1$ and

$$k_{inf} = w_1 \cdot k_1 + w_2 \cdot k_2 \quad (4)$$

where w_1 and w_2 are weighting parameters that depend on the concentration of S_1 elements and S_2 elements in the whole system. By changing $w_1$or w_2, k_{inf} can be adjusted. In this work, we are only going to control w_1 in order to adjust k_{inf}. For example, if k_{inf} is too high, we can remove some S_1 elements to decrease w_1 and thus decrease k_{inf}. Likewise, if k_{inf} is too low, we will add S_1 nuclides so that w_1 increases and this will in turn increase k_{inf}.

To represent the injection or removal of S_1 elements, we make use of λ_{art} and equation (1). Hence, when we want to increase w_1, a positive λ_{art} will be added in equation (1) for the nuclides of the subsystem S_1. Adding a negative λ_{art} to (1) will thus decrease w_1.

Next, we find an analytical expression for λ_{art} in order to avoid using iterations as in Auferio [14]: Considering the time step $\Delta t = t_1 - t_0$ over which we aim to solve (2) and (3) and using the relation in (4), we find a simple and explicit equation that relates λ_{art} to the criticality we want to reach at t_1, k_{opt}:

$$\lambda_{art} = \frac{1}{\Delta t}\left[\log\left(\frac{k_{opt}-k_2(t_1)}{k_1(t_1)-k_{opt}}\right) - \log\left(\frac{k_{inf}(t_1)-k_2(t_1)}{k_1(t_1)-k_{inf}(t_1)}\right)\right] \quad (5)$$

IMPLEMENTING THE ALGORITHM INTO A NEUTRONICS CODE

In order to verify its viability, the proposed model and algorithm are integrated into ORIGEN. To solve the depletion equation, ORIGEN divides the overall time into small depletion time steps. Within each of these, the system of equations constituted of each nuclide's depletion equation (1) can be solved using two different methods: the Taylor expansion [15] or the CRAM method [16]. For this work, the Taylor expansion method is used. The new set of nuclide densities found will then serve as the initial condition for the next depletion time step.

The online injection/removal algorithm is integrated into this sequential approach. Figure 1 summarizes the process of one depletion time step.

First, the depletion module runs a first time from t_0 to t_1 to obtain the criticality of the whole system $k_{inf}(t_1)$, the criticality of S_1 $k_1(t_1)$ and the criticality of S_2 $k_2(t_1)$ if no injection or removal is operated (Step A in Fig.1). With this information, λ_{art} can be computed (Step B). We then add λ_{art} to the depletion equation (1) in the ORIGEN system and run a new depletion calculation (Step C).

Because the algorithm relies on a very simple formula to characterize a very complex system there must be a price to pay. The algorithm indeed appears to be somewhat unstable in certain circumstances. First, while λ_{art} as a function of k_{opt} (5) is almost linear in the vicinity of k_{inf} (the criticality obtained with no injection or removal), the function diverges for values of k_{opt} approaching k_1 or k_2, because of the function's logarithmic nature. To understand this effect, we need to remember that (5) is obtained via approximating (1) with (2). It happens that (5) is no longer a valid approximation of the system when k_{opt} is chosen to be close to k_1 or k_2. Secondly, since the system to which λ_{art} is added is equation (1) and not (2), the criticality obtained after injection will be slightly different than k_{opt}. In practice, the calculated λ_{art} will often be bigger than what is needed thus resulting in criticality adjustments that are too drastic. This effect results in spikes in the criticality evolution.

To control and reduce the effects of these phenomena, formula (5) is weighted with two stabilizing coefficients. The first one aims at preventing λ_{art} from diverging when the value k_{opt} is close to k_1 or k_2. The second one is added to mitigate over-adjustments of the criticality. This second coefficient is determined by a function of the deviation of k_{inf} from k_{opt} and takes a value between 0 and 1. When the criticality starts to deviate from its desired value, this coefficient is going to be equal to 0. The more the criticality deviates, the more it increases and gets closer to one. This results in a gradual, smoother injection. While this coefficient helps reducing the magnitude of the spikes in the criticality evolution, it does not totally remove them.

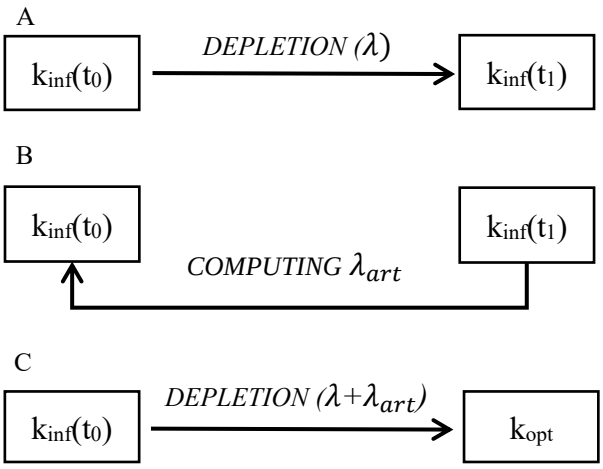

Figure 1: The online injection and removal algorithm for one depletion time step

In order to update the neutron flux spectrum and the single group cross-sections, ORIGEN has been coupled with the open-source Monte Carlo transport code OpenMC. The interface that couples them ("Kadabra") has been written in Python and uses the Pyne suite of numerical tools for nuclear engineering [17].

PHYSICAL MODEL

We test our neutronics code on a DMSR design developed by ORNL [1] with the thorium fuel cycle. Fig. 2 shows the lattice configuration for the DMSR. The molten salt flows inside graphite tubes as well as in the space between them. An infinite lattice of these unit cells is simulated in OpenMC.

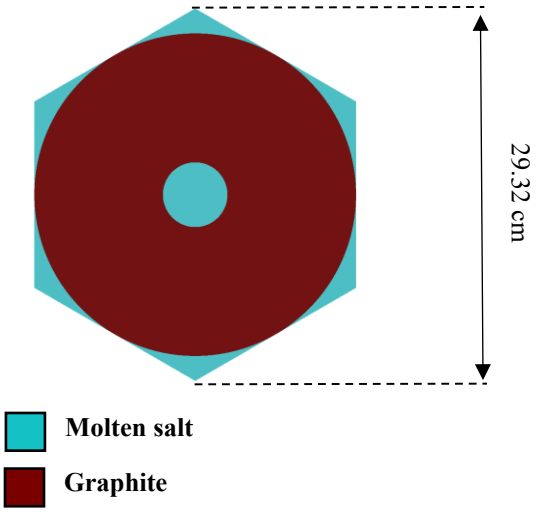

Figure 2: The lattice configuration as used in the OpenMC simulation

The fuel salt is composed of 74 mol% of LiF, 16.5 mol% of BeF2, 8.2 mol% of ThF4 and 1.3 mol% of UF4. The initial uranium enrichment is 20% uranium-235. The power density is 16 kW per liter. The cross-section library used is ENDF/B-VII.1 and since a MSR operates at higher temperature, the cross-sections were taken at 900 Kelvin.

BENCHMARK

In order to assess the validity and the reliability of the neutronics code used in this work, it has been compared to MCODE [18] which couples MCNP5 [19] with ORIGEN2.2 [20], an older version of ORIGEN. We simulated the described lattice using the same cross-section libraries. It is important to note that since MCODE does not offer the option to model continuous injection and removal, the version of Kadabra used for the benchmark has its injection/removal module deactivated.

Figure 3 is a comparison of the neutron flux spectrum obtained with MCODE and with Kadabra. Both spectra are taken at around 1000 days of operation. The plotted value is $\Phi \cdot E$, where Φ and E are the neutron flux and energy, respectively. This enables us to remark that the flux spectrum is thermal as expected. We can also see that the spectra of the two codes match well.

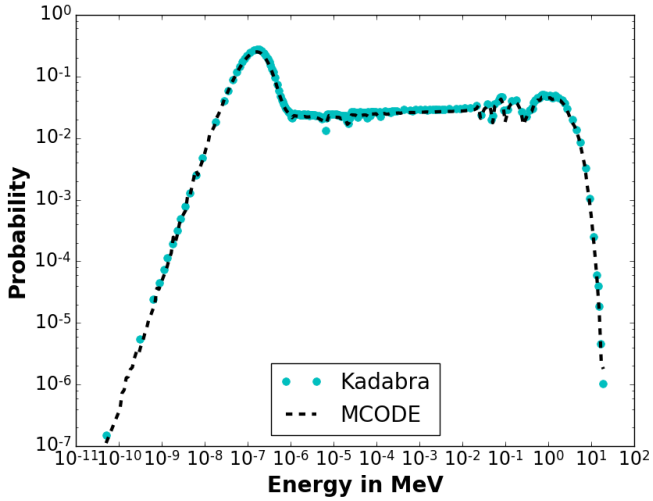

Figure 3: Comparison of the neutron flux spectrum $\Phi \cdot E$

Figure 4 shows the evolution of the plutonium-239 and plutonium-240 concentrations as calculated by both codes. While there is very good agreement for plutonium-240, Kadabra's plutonium-239 concentration is somewhat lower. However, we see that this discrepancy is stable and does not increase with time.

Figure 5 shows additional comparison results for the nuclides that are the most important from the neutronics point of view.

The graph indicates the relative differences of the nuclide densities between MCODE and Kadabra at the time where the reactor has been operational for 10 years. We see that the relative differences for are acceptably small, at most 3% (for plutonium-239). The same is true for the other nuclides not represented in the graph.

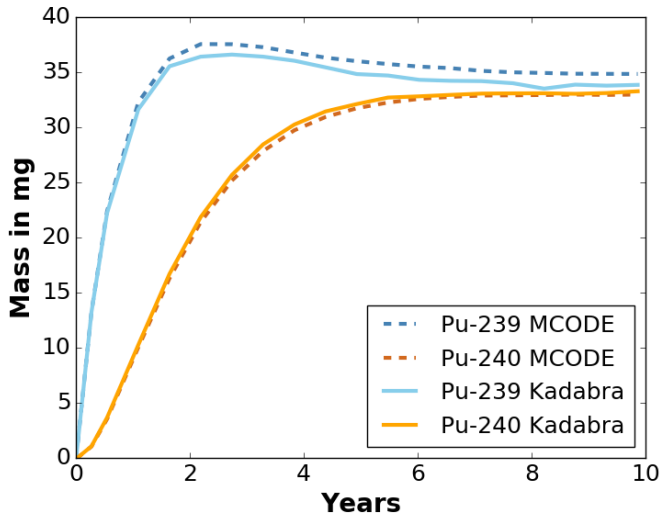

Figure 4: Comparison of plutonium isotopes mass evolution between MCODE and Kadabra

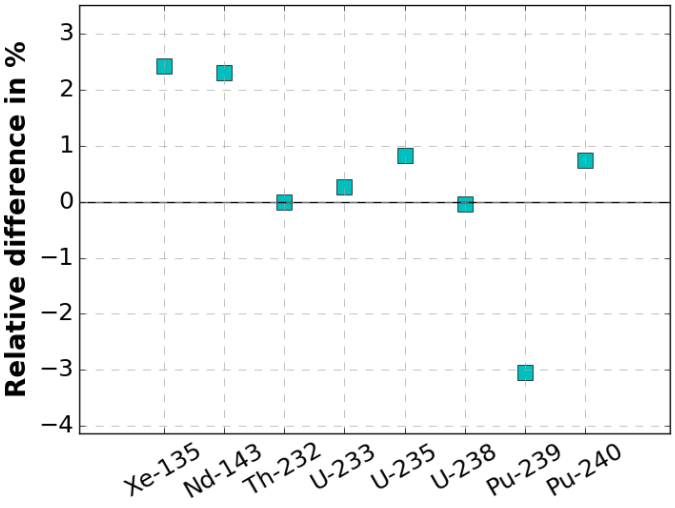

Figure 5: Relative difference of isotopic densities for several important nuclides between MCODE and Kadabra at 10 years of reactor operation

Given the very good match of MCODE and Kadabra flux spectra as well as the acceptable relative errors for the isotope density evolutions, we can conclude that Kadabra is accurate and reliable for simulating the lattice considered in this work.

RESULTS

To simulate the DMSR operation using Kadabra, we activate the fuel injection/removal module. The simulations aim at checking the stability of the algorithm used in Kadabra, the usefulness of the stabilizing coefficients, and at verifying the accuracy and validity of Kadabra by checking if the DMSR's fissile inventory evolution obtained matches the one indicated in the ORNL report.

To test the stability of our algorithm which is designed to optimize fuel processing to keep the criticality at a desired constant value, we compare three simulations of the DMSR lattice. The first does not inject or remove fuel, which results in a reduction of k_{inf} over time (an eventually unphysical system). The second maintains k_{inf} at 1.05 by injecting 20%-enriched uranium but does not use the stabilizing coefficients introduced previously. Finally, the last simulation keeps k_{inf} at 1.05 while using the coefficients. While the algorithm is also able to remove nuclides (plutonium for example) to decrease criticality, this is not necessary for the system we simulate and thus removal has been deactivated. The depletion time step, i.e. the time step over which ORIGEN solved the system of depletion equations (1) is set to 10 days. The transport code OpenMC is called every 200 days to update the flux and the cross-sections.

Figure 6 shows the evolution of the reactivity ρ. The upper plot shows that the injection algorithm works as the criticality is effectively maintained around k_{inf}=1.05 (ρ = 4761 pcm) while the simulation with no injection has a reactivity that goes well below this value. However, we see that the reactivities of the simulations with fuel injection present an edgy pattern, especially at the beginning. These spikes stem from the fact that the injection is calculated via approximating (1) with (2). A way to reduce them is to use the stabilizing coefficients. We can indeed see on the lower plot that using the stabilizing coefficients enables a reduction of the magnitudes of the spikes from over 500 to below 100 over the long term. However, the stabilizing coefficients fail to mitigate the magnitude of the spikes at the beginning of the injection process.

We also notice from figure 6 that the spikes at the beginning of the injection process are larger than the spikes over the rest of the life of the reactor. A possible explanation is that the first injections occur when the reactor is in a dynamic state where many isotope densities are changing more rapidly than at later points, so that the criticality response to a given injection might be much less linear than during the rest of the life of the reactor.

None of the spikes, however, has a relevant influence on the total fissile fuel quantity that is going to be injected over the whole life of the reactor: While a large spike represents the sudden injection of more fissile material than necessary, these spikes are separated by longer periods of no injection compared to the smaller spikes, as it takes more time for the reactivity to fall below 0. The larger spikes only reduce somewhat the time resolution of the injection pattern.

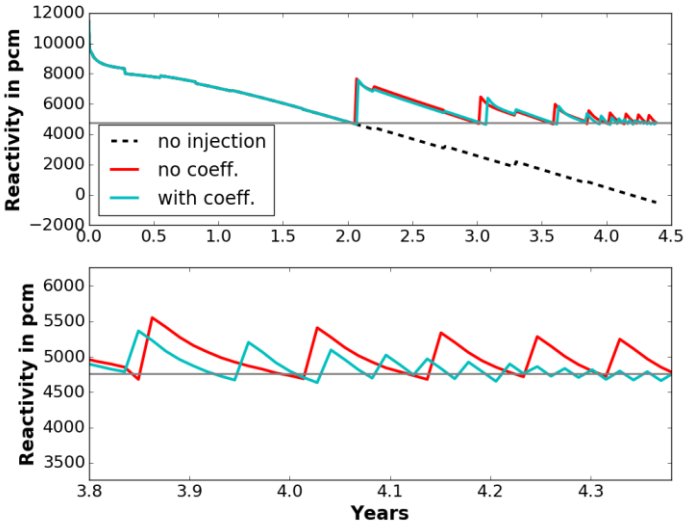

Figure 6: k-inf for a DMSR with 20% enriched uranium injection. The bottom graph contains a smaller time window. The gray horizontal line indicates $k_{inf} = 1.05$

Figure 7 shows the evolution of the most important fissile isotopes as well as the cumulative amount of uranium-235 that is injected scaled up for a reactor of 2250 MWt (same thermal power as the ORNL DMSR). During the first 5 years, we can see that the U-235 is being depleted while U-233 builds up. During the 4th year of operation, the injection of enriched uranium starts and, as a result, the near-exponential decrease of the U-235 concentration is being mitigated. The U-235 mass starts to increase in the 12th year. The curve representing the accumulated injection of U-235 is almost linear which indicates that the rate of injection during the whole life of the reactor does not significantly change.

Figure 8 compares the accumulated injection of U-235 obtained with Kadabra to the accumulated injection as reported in the ORNL report. Both curves follow the same general trend and result in a very similar value for the total injected U-235 mass: 4470 kg for ORNL and 4545 kg for Kadabra.[1] However, the two curves differ in their resolution. The simulation used in the ORNL report only injects enriched uranium once a year (203 kg) thus resulting in a rather discontinuous injection pattern, in contrast to the pattern produced by Kadabra. Kadabra is able to provide a much more refined resolution for the injection pattern which is defined by the user (the simulation used for this work used a 10 days resolution but the user could very well choose an even more refined resolution).

[1] These results are achieved by selecting a criticality factor of 1.05. Smaller criticality factors lead to lower estimates of the injected U-235 mass.

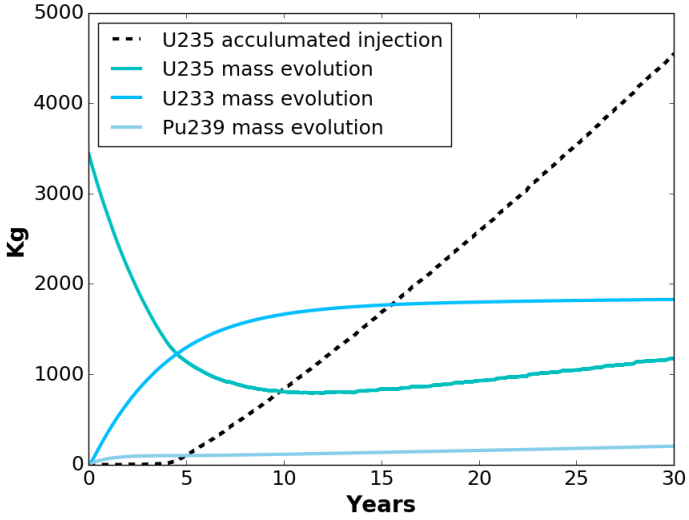

Figure 7: Fissile isotopes evolution and the accumulated injection of U-235 over reactor operation time

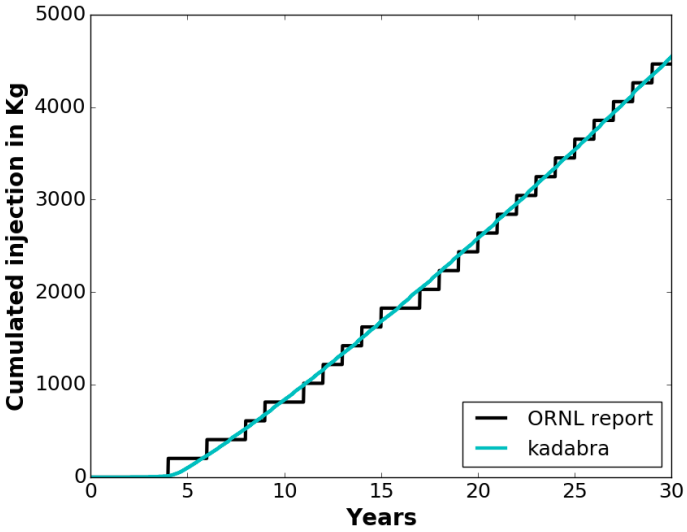

Figure 8: Comparison of the accumulated U-235 injection obtained with Kadabra and given by ORNL

DISCUSSION

In this paper, we develop and implement a novel algorithm that analytically calculates the optimal fuel injection and removal schemes to obtain a desired reactor condition (e.g. criticality, chemical balance, isotopic ratio). It significantly improves simulation capabilities for MSRs and could also be used to model other types of circulating fuel reactors such as

pebble bed reactors. Our algorithm automatically calculates fuel injection and removal by requiring a single iteration per depletion time step, while previous methods require hundreds of iterations due to the lack of an analytical expression. We obtain such analytical expression by using a homogeneous approximation of the Bateman equation. Lastly, while current neutronics codes enable the user to add a reprocessing rate to the depletion equation, these rates are only updated each time the depletion module is called. In this work, the correct injection and removal is calculated for every depletion time step within the depletion module, increasing the resolution of the optimal injection and removal behavior.

Future work on the algorithm should include improving the stabilizing coefficients to further reduce or eliminate the edgy patterns of k_{inf}. The code should also be tested on further scenarios (e.g. isotopic ratio, chemical balance).

While the work presented in this paper is mainly conceptual, the long-term goal of this project is to provide an open-source neutronics package that can accurately model the operation of an MSR. To achieve this, ORIGEN, which is not open source, would need to be replaced. The circulation of the fuel will also need to be modeled as the fuel successively becomes critical when passing through the reactor core and under-critical when flowing outside the core.

REFERENCES

[1] Engel, J.E., Baumann, H.F., Dearing, J.F., Grimes, W.R., McCoy, H.E., Rhoades, W.A., 1980, "Conceptual design characteristics of a denatured molten-salt reactor with once-through fueling," ORNL/TM-7207, Oak Ridge National Laboratory.

[2] Rearden, B.T. and Jesse, M.A., Eds., SCA LE Code system, ORNL/TM-2005/39, Version 6.2, Oak Ridge National Laboratory, Oak Ridge, Tennessee (2016). Available from Radiation Safety Information Computational Center as CCC-834.

[3] Romano, P.K., Horelik, N.E., Herman, B.R., Nelson, A.G., Forget, B., Smith, K., 2015, "OpenMC: A state-of-the-art Monte Carlo code for research and development," Annals of Nuclear Energy 82, pp. 90-97.

[4] Bateman, H., 1910, "The solution of a system of differential equations occurring in the theory of radio-active transformations," Proceedings of the Cambridge Philosiphical Society, Mathematica and physical sciences, 15, pp. 423-427.

[5] Sheu, R.J., Chang, C.H., Chao, C.C, Liu, Y.-W.H., 2012, "Depletion analysis on long-term operation of the conceptual Molten Salt Actinide Recycler & Transmuter (MOSART) by using a special sequence based on SCALE6/TRITON," Annals of Nuclear Energy 53, pp. 1-8.

[6] Powers, J. J., Harrison, T. J., Gehin, J. C., 2008, "A new approach for modeling and analysis of molten salt reactors using SCALE," Oak Ridge National Laboratory.

[7] Zhuang, K. et al., 2014, "Studies on the molten salt reactor: code development and neutronics analysis of MSRE-type design," Journal of Nuclear Science and Technology 52:2, pp. 251-263.

[8] Ahmad, A., McClamrock, E.B., Glaser, A., 2015, "Neutronics calculations for denatured molten salt reactors: Assessing resource requirements and proliferation-risk attributes," Annals of Nuclear Energy 75, pp. 261-267.

[9] Bauman, H. F., Cunningham III, G. W., Lucius, J. L., Kerr, H. T., Craven, C. W., 1971, "ROD: A nuclear fuel-cycle analysis code for circulating fuel reactors," ORNL/TM-3359, Oak Ridge National Laboratory.

[10] Nuttin, A. 2002, "Potentialiés du concept de réacteur à sels fondus pour un production durable d'énergie nucléaire basée sur le cycle thorium en spectre épithermique," Ph.D. thesis, Université Joseph-Fourier – Grenoble I.

[11] Mathieu, L., 2005 "Cycle thorium et réacteurs à sel fondu. Exploration du champ des paramètres et des contraintes définissant le 'Thorium Molten Salt Reactor '" Ph.D. thesis, Institut National Polytechnique de Grenoble.

[12] Nagy, K., Kloosterman, J.L., Lathouwers, D., van der Hagen, T.H.J.J., 2011, "New breeding gain definitions and their application to the optimization of a molten salt reactor design," Annals of Nuclear Energy 38, pp. 601-609.

[13] Aufiero, M., Cammi, A., Fiorina, C., Leppänen, J., Luzzi, L., Ricotti, M.E., 2013, "An extended version of the SERPENT-2 code to investigate fuel burn-up and core material evolution of the Molten Salt Fast Reactor," Journal of Nuclear Materials 441, pp. 473-486.

[14] Aufiero, M., 2014, "Development of advanced simulation tools for circulating-fuel nuclear reactors," Ph.D. thesis, Politecnico di Milano.

[15] Moler, C., Van Loan, C., 2003, "Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later," SIAM review 45, pp. 3-49.

[16] Pusa, M., Leppänen, J., 2009, "Computing the Matrix Exponential in Burnup Calculations," Nuclear Science and Engineering 164, pp. 140-150.

[17] Bates, C., Biondo, E., Huff, K., et al., 2014, "PyNE Progress Report," Am. Nucl. Soc. Winter Meeting 2014 111.

[18] Xu, Z., 2003, "Design strategies for optimizing high burnup fuel in pressurized water reactors," Ph.D. Thesis, Massachusetts Institute of Technology.

[19] X-5 Monte Carlo Team, 2008, "MCNP – A general Monte Carlo N-particle transport code, version 5," LA-UR-03-1987, Los Alamos National Laboratory.

[20] Ludwig, S.B., Renier, J.P., 1989, “Standard- and extended-burnup PWR and BWR reactor models for the ORIGEN2 computer code," ORNL/TM-11018.