ANALYSIS OF AND COMPARISON BETWEEN REPROCESSING METHODS IN THE MOLTEN SALT BREEDER REACTOR

BY

LUKE SEIFERT

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Nuclear, Plasma, Radiological Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2023

Urbana, Illinois

Master's Committee:

Madicken Munk
Tomasz Kozlowski

Abstract

Molten salt reactor modeling can be challenging in liquid fueled cores as the fuel composition varies temporally and spatially. Although these models are challenging, they are important for predicting reactor safety and performance during operation. One of the reasons these models are challenging is the difficulty in capturing different physical phenomena which occur in different time scales. Over short time scales, one of the physical phenomena is the movement of delayed neutron precursors to less important regions of the reactor. For longer time scales, one of the physical phenomena is online reprocessing, which includes adding fresh fuel and removing fission products from the reactor during operation and is the focus of this work. Reprocessing is useful in liquid fueled molten salt reactors because it limits chemical corrosion and improves the neutron economy. The reprocessing can be performed continuously during operation, referred to as online reprocessing, or the reprocessing can be performed in batches during outages, referred to as batchwise reprocessing. In order to simulate reprocessing computationally, there are two different mathematical approaches which are commonly implemented in the literature, called batchwise and continuous reprocessing. The continuous reprocessing method is more physically reflective of the operation in reality for any reactor which uses a continuous chemical reprocessing scheme. However, there are many works which have implemented batchwise reprocessing to simulate a continuous reprocessing process. In this thesis, I show that continuous reprocessing and batchwise reprocessing methods are not interchangeable, and there is a fairly large difference in the results when using each approach for the same system. I also investigate the computational cost of different reprocessing methods by performing a depletion time step refinement study. In this study, I found that continuous reprocessing allows for significantly larger time steps without large increases in error, which reduces computational cost. However, continuous reprocessing does not necessarily keep the overall mass constant, thus potentially leading to an nonphysical solution. I compare the differences between both methods while determining the effect of this nonphysical mass change caused by continuous reprocessing.

Acknowledgments

This material is based upon work supported under an Integrated University Program Graduate Fellowship. Any opinions, findings, conclusions or recommendations expressed in this publication are those of the author(s) and do not necessarily reflect the views of the Department of Energy Office of Nuclear Energy.

I would like to thank Andrei Rykhlevskii for his assistance with SaltProc and helpful advice. I would also like to thank Oleksandr Yardas for proofreading this work. Thanks should also go to Madicken Munk and Tomasz Kozlowski for reviewing this work. I would like to thank Ondrej Chvala and Katy Huff for their assistance in the early stages in this work. Finally, I thank the Advanced Reactors and Fuel Cycles group for their code reviews and comments on my results.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Molten Salt Reactors

One of the generation IV reactor designs is the molten salt reactor, or MSR [22]. In this work, the main MSR which will be discussed is the liquid fueled MSR. Moving forward, I will use the term MSR when referring to the liquid fueled variant unless specified otherwise. The MSR is an advanced reactor design which is intended to improve safety and efficiency over previous reactor designs.

MSRs, instead of a solid fuel, use a liquid fuel composed of a molten salt mixed with some fissile isotope, such as uranium-235. Some of the benefits of MSRs include high operating temperature, low operating pressure, and high thermal conductivity and capacity of the salts. Other benefits include a high coefficient of thermal expansion for negative temperature coefficient of reactivity, high resource utilization through higher burn-up, reduced cost of fabricating and transporting fuel elements, and the removal of fission products to improve the neutron economy [41].

MSRs also have several barriers to deployment which must be addressed before they are commercialized. Corrosion and material degradation is an issue for MSRs, as the high temperature salts are highly corrosive. Another barrier to deployment is in regulation, as the MSR operates very differently from light water reactors on the grid. In order to properly regulate MSRs, we must have models which are able to simulate them correctly. One type of simulation which is important for determining fuel composition and reactor properties after long periods of operation are depletion simulations, which are of key interest for this work.

1.2 Physical Depletion and Reprocessing

Depletion is the process of fission over time, leading to fission products and decay chains. Fission product yields are dependent on both the fuel composition and the interacting neutron energy. Because the core neutron energy spectrum may shift spatially in the core, this means fuel and fission products are constantly changing in both composition and location. For an MSR with online reprocessing, the additions and losses from reprocessing have to

be considered within depletion.

Reprocessing can be considered through two methods: either as a continuous process or as a batchwise process. Continuous reprocessing is a process which involves adding fresh fuel to the system, as well as chemically removing fission products. MSRs are also capable of batchwise reprocessing, which is a process which occurs in discrete intervals. An example of this is salt disposal after some set amount of time, which requires shutdown of the reactor. In general, online reprocessing schemes use continuous reprocessing.

The usefulness of online reprocessing is that the reactor does not need excess reactivity in order to continue operating for a long period of time. In a typical light water reactor, burnable poisons and control rods are used to lower reactivity at the beginning of the core lifetime so the reactor can remain at a safe critical level over the core lifetime. In an MSR, the parasitic absorbers, such as xenon-135, can be removed from the reactor during operation to reduce absorption and losses in the core to further improve the neutron economy. Additionally, fresh fuel salt can be added to the reactor in order to maintain stable operation.

The fresh fuel feed can include fissile isotopes, such as uranium-233 or uranium-235; fertile isotopes, such as uranium-238 or thorium-232; or some combination of both. Adding a fissile isotope to the reactor will allow the reactor to continue operating through fission directly. In fertile isotopes, the neutron first absorbs a neutron, then the product of that reaction decays and produces a fissile isotope. This process is illustrated for thorium-232 breeding in Equation (1.1), which is the breeding process used in the Molten Salt Breeder Reactor (MSBR). This process creates fissile fuel on a time delay related to the fluence that the fuel is exposed to. By using a balance of fissile and fertile feeds, the reactor can breed new fuel while it continues to operate with fissile fuel.

$$^{232}_{90}\mathrm{Th} + ^{1}_{0}\mathrm{n} \longrightarrow ^{233}_{90}\mathrm{Th} \xrightarrow{\beta^-} ^{233}_{91}\mathrm{Pa} \xrightarrow{\beta^-} ^{233}_{92}\mathrm{U} \tag{1.1}$$

The half life of thorium-233 via beta decay is on the order of tens of minutes, while the half life of protactinium-233 via beta decay is roughly 27 days. If the protactinium-233 remains in the core, some of it will be removed by competing neutron absorption. This will impact the breeding efficiency and also removed neutrons from fissioning in the core. To minimize this, the MSBR continuously chemically extracts the protactinium from the fuel salt, allowing it to decay in a separate tank until it decays into a fissile fuel. Once it has decayed, another continuous chemical process removes the fissile fuel from the tank and reloads it into the reactor.

Overall, the two different physical approaches in MSR reprocessing are continuous and batchwise. The reprocessing schemes can be online, while the reactor is operating, or offline, when the reactor is shutdown. Of particular importance for MSR behaviour is online reprocessing, which changes the fuel composition while the reactor is operating. Because existing software may do depletion or reprocessing but not both, this is an important type of problem to be able to model in MSR research.

1.3 Computational Depletion and Reprocessing

Depletion calculations require knowledge of the reactor composition and the reactor power. Through this, we can calculate the number of fissions and thus the fission products created by depleting the fuel. This is commonly done with neutron transport simulations. The transport simulation itself can be performed using a stochastic method, namely Monte Carlo, or a deterministic method, such as diffusion. However, traditional depletion software does not account for material changes in the core as a result of online or offline reprocessing of the fuel.

In order to model online reprocessing in a depletion simulation, there are two different computational methods which can be implemented. These methods reflect the physically continuous and batchwise reprocessing methods and have the same naming convention. However, it is fairly common for researchers modeling MSR simulations with a physically continuous reprocessing scheme to use approximate batchwise mathematical models in their simulations.

In this work, I will compare and investigate the differences between continuous and batchwise computational reprocessing methods. Because the effects of depletion and reprocessing are temporal, I will vary the depletion step size and compare the effects of this on reactor physics. I will compare the validity of batchwise methods approximating continuous methods, and I will quantify the effects of different variations of the methods within both batchwise and continuous.

Chapter 2

Literature Review

In this chapter, I will provide an overview of MSR depletion modeling in the literature. First, I will discuss the different reprocessing methods and treatment of delayed neutron precursors. I will then provide information on different depletion codes and how they have been applied to various MSRs. This will cover the state of the literature in terms of handling reprocessing in depletion simulations. This set of works will also show the number of applications which have employed batchwise methods to approximate continuous reprocessing schemes. This is important, because a key takeaway of this work is that this approximation causes a buildup of error in the material composition. I will also discuss the MSBR specifically, as the MSBR is the reactor used to generate the results in this work.

2.1 Modeling Requirements for Molten Salt Reactors

Modeling and simulation of liquid fueled molten salt reactors (MSRs) differs from solid fueled reactors in two main areas. The first is in the fresh fuel feed and fission product removal streams used during operation in MSRs; which is referred to as online reprocessing. Modeling an MSR without online reprocessing will cause discrepancies that will increase as the net depletion time increases. This is a result of neutronic differences that arise from a lack of fresh fuel and buildup of fission products. In order to simulate this reprocessing functionality, software can use batchwise and continuous reprocessing methods.

The second difference is in the movement of the fuel salt, which causes the fuel and, consequently, the delayed neutron precursors (DNPs) to move in the core. Because the DNPs have different half lives before the delayed neutrons are born, the effect of the fuel movement on each precursor group varies. However, the overall result is that some of the delayed neutrons are birthed in regions outside of the active core, such as external piping, which means there is a reduced effective delayed neutron fraction in the core. This fraction is important for safe operation and reactivity control, so modeling it accurately is important to advancing this technology.

2.1.1 Online Reprocessing

The online reprocessing functionality of MSRs, such as fresh fuel feed and/or fission product removal, is specified in the literature using the elemental cycle times from a reactor's reprocessing scheme. These elemental cycle times specify the rate of removal of that given element, where the mass removal rate will scale based on the cycle time and the amount of that element present in the reactor. From this data, the online reprocessing is simulated in different ways depending on the particular software used, and depending on the methods employed by that software. The two main ways to approach online reprocessing are to use either batchwise reprocessing or continuous reprocessing methods. A more in depth explanation of the reprocessing methods will be provided in the Methodology section, but in general, the batchwise method performs discrete compositional updates at given time steps while the continuous method modifies the depletion matrix such that the modifications to the composition due to online reprocessing happen continuously independent of the time steps.

A purely batchwise approach without a sufficiently short time step will not fit the reprocessing design used by an MSR, as online reprocessing is carried out continuously. For the batchwise method to fit a continuous reprocessing scheme, the time step would need to be on the same time scale as the shortest cycle time, where the cycle time is time it takes for a given element to be completely removed. For the MSBR, the shortest cycle time is 20 seconds, which would make a depletion simulation on the order of years much more computationally expensive than it would otherwise be. Even if transport simulations are not repeated, a single simulated year with 20 second time steps would require 21,915 depletion calculations to be completed. The batchwise method can accurately model a reprocessing scheme which uses batchwise reprocessing, such as a scheme which has salt entirely replaced after some given amount of time. An example of this is a salt discard, which occurs after a set amount of time. An entirely batchwise reprocessing scheme for an MSR would be neglecting the benefit of a liquid fuel, so such a scheme is not common in the literature.

Two different codes which treat batchwise reprocessing are SaltProc, which is discussed later in this chapter, and older versions of ChemTriton. Although they are distinct and have different use cases, for the purposes of this work, I am interested in the treatment of batchwise reprocessing in these codes. These codes both implement batchwise reprocessing by running the depletion simulation for a fixed amount of time, then stopping, and then adding and removing materials [39, 8]. These codes iterate through this process for each depletion step until they reach the final time.

This method is useful because it is fairly straightforward to implement, easy to customize, and makes mass balancing of the core straightforward. Some issues with the batchwise approach are that using large time steps will make the results less accurate; it has to rerun the transport simulation for each time step for pre-reprocessing and post-reprocessing; and it is an approximation of the actual physical process of reprocessing. In an operating

MSR, the fission products are continuously removed while any feed flows are continuously fed into the reactor, not removed in batches. The reason that large time steps make the results less accurate can be made clear by imagining the limiting case, a single very large depletion step. Taking a very large time step would mean that all of the reprocessing that should have taken place during that time step does not. For example, instead of removing the xenon every 20 seconds, there is only a single removal at the very end after potentially years. This means that the parasitic absorption of neutrons by fission products is not at all reduced over the entire depletion step, but instead the composition is only adjusted at the very end. This example makes clear why small depletion steps are requiredIn this chapter, I will provide an overview of MSR depletion modeling in the literature. First, I will discuss the different reprocessing methods and treatment of delayed neutron precursors. I will then provide information on different depletion codes and how they have been applied to various MSRs. This will cover the state of the literature in terms of handling reprocessing in depletion simulations. This set of works will also show the number of applications which have employed batchwise methods to approximate continuous reprocessing schemes. This is important, because a key takeaway of this work is that this approximation causes a buildup of error in the material composition. I will also discuss the MSBR specifically, as the MSBR is the reactor used to generate the results in this work.

2.2 Modeling Requirements for Molten Salt Reactors

Modeling and simulation of liquid fueled molten salt reactors (MSRs) differs from solid fueled reactors in two main areas. The first is in the fresh fuel feed and fission product removal streams used during operation in MSRs; which is referred to as online reprocessing. Modeling an MSR without online reprocessing will cause discrepancies that will increase as the net depletion time increases. This is a result of neutronic differences that arise from a lack of fresh fuel and buildup of fission products. In order to simulate this reprocessing functionality, software can use batchwise and continuous reprocessing methods.

The second difference is in the movement of the fuel salt, which causes the fuel and, consequently, the delayed neutron precursors (DNPs) to move in the core. Because the DNPs have different half lives before the delayed neutrons are born, the effect of the fuel movement on each precursor group varies. However, the overall result is that some of the delayed neutrons are birthed in regions outside of the active core, such as external piping, which means there is a reduced effective delayed neutron fraction in the core. This fraction is important for safe operation and reactivity control, so modeling it accurately is important to advancing this technology.

2.2.1 Online Reprocessing

The online reprocessing functionality of MSRs, such as fresh fuel feed and/or fission product removal, is specified in the literature using the elemental cycle times from a reactor's reprocessing scheme. These elemental cycle times specify the rate of removal of that given element, where the mass removal rate will scale based on the cycle time and the amount of that element present in the reactor. From this data, the online reprocessing is simulated in different ways depending on the particular software used, and depending on the methods employed by that software. The two main ways to approach online reprocessing are to use either batchwise reprocessing or continuous reprocessing methods. A more in depth explanation of the reprocessing methods will be provided in the Methodology section, but in general, the batchwise method performs discrete compositional updates at given time steps while the continuous method modifies the depletion matrix such that the modifications to the composition due to online reprocessing happen continuously independent of the time steps.

A purely batchwise approach without a sufficiently short time step will not fit the reprocessing design used by an MSR, as online reprocessing is carried out continuously. For the batchwise method to fit a continuous reprocessing scheme, the time step would need to be on the same time scale as the shortest cycle time, where the cycle time is time it takes for a given element to be completely removed. For the MSBR, the shortest cycle time is 20 seconds, which would make a depletion simulation on the order of years much more computationally expensive than it would otherwise be. Even if transport simulations are not repeated, a single simulated year with 20 second time steps would in order to generate useful results using the batchwise method.

Two different codes which treat continuous reprocessing are Serpent2 and ChemTriton. Serpent2 is a continuous energy Monte Carlo transport code which also has depletion functionality and a more recently added continuous reprocessing functionality [4]. ChemTriton is a Python script developed to interface with SCALE/TRITON, where the more recent updates have been directly implemented into SCALE to handle continuous reprocessing [21].

These codes implement continuous reprocessing by adding terms to the Bateman equation similar to an extra decay term, as can be seen in Equation (2.1). The benefits of using continuous reprocessing are that the model is more physically accurate and it allows for larger time steps to be used without increasing error. However, the largest time step size cannot become too large, or the simulation will lose accuracy due to the lack of updated cross section data from transport simulations. The exact value of the largest time step while remaining sufficiently accurate is dependent upon the reactor, the rate at which the material composition changes, and the level of accuracy which is desired. This is because the neutron spectrum and cross section data does not update to the new material compositions until a new depletion step begins.

$$\frac{dN_j}{dt}_{net} = \left(\frac{dN_j}{dt}\right)_{no\ reprocessing} - \lambda_{r,j}N_j + \sum_{feed} \lambda_{f,j}N_j \tag{2.1}$$

Another aspect of online reprocessing that must be considered is mass balancing. As mentioned previously, mass balancing is straightforward for batchwise reprocessing and can be approached in several different ways. One method is to have the net mass of the feed rate over some depletion step be equivalent to the mass removed over that same depletion step, which is the current approach used by SaltProc [39]. Alternatively, the volume of the system can be adjusted so that constant density is maintained in order to keep cross section data consistent [33]. Another approach is to move excess fuel salt into a bleed-off tank [33].

For mass balancing using continuous reprocessing, the depletion time steps can be much larger than those used in batchwise reprocessing. This can cause a mass balancing issue if the system has a large feed rate but a small removal rate. For example, if 1 kg is added per day, and 0.2 kgs are removed per day from fission product removal, then the net mass of the system is increasing. This could cause the simulation to continuously increase in mass, thus containing more fuel salt than intended and generating invalid results. In Serpent2, this increase of mass during a depletion step directly leads to an increase in density rather than a change in volume. One way this can be treated, assuming the net change in mass in non-negligible, is to use batchwise reprocessing to balance the mass while continuous reprocessing handles the actual reprocessing. However, implementing batchwise reprocessing increases computational cost. This is because stopping and starting depletion simulations in order to modify compositions is slower than having the computation simultaneously solved in the depletion calculation. Additionally, in codes such as OpenMC and Serpent2, having multiple depletion steps in a single input file reduces the amount of time spent loading in data. If the user also wants to have transport run at each step, these codes will also run transport before reprocessing. This means that the batchwise reprocessing will roughly double the computational cost due to the additional transport simulations that are performed before reprocessing is performed. This limitation is not absolutely tied into batchwise reprocessing, and this extra transport simulation could be removed.

2.2.1.1 Batchwise Reprocessing

Batchwise reprocessing starts with some removal rate where some fraction, f, of a given element is removed from the fuel salt in the system over some time period, T_{cyc}, referred to as the cycle time. Over a given depletion simulation time step, Δt, a different fraction, γ, of the element's mass is removed, where γ is related to f as a function of the time and cycle time as shown in Equation (2.2). The scaling term on f for the removal efficiency is a linear approximation based on the time step adjustment in Equation (2.2). If the time step used causes the fraction removed, γ, to be greater than 1, the amount removed remains at 100% so as to not have negative mass. This method is called Steady batchwise reprocessing since it uses a steady removal amount during each depletion

step. An example of this approach is in the work by Hombourger et al, where the derivation for the a very similar equation is also shown [19].

Examples of the Steady batchwise method can be seen in Table 2.1. Both examples show what the resulting mass removal per step, or γ, term should be for different cycle time values. The first shows a cycle time of 20s, which is shorter than the step time of three days. This results in a removal rate over 100%, which is set to 100%. The second shows a cycle time of 60 days, longer than the step size. This results in a 5% removal per step. An alternative approach to the second example, which is implemented in the earlier versions of SaltProc, is to perform no removal until 60 days have elapsed and then remove 100% of material.

f	T	Δt	γ_{calc}	γ_{actual}
100%	20 s	3 d	12,960%	100%
100%	60 d	3 d	5%	5%

Table 2.1: Examples of steady batchwise reprocessing.

An alternative approach is to establish a depletion time step, Δt, such that the cycle time, T_{cyc}, is a multiple of Δt. Then, the fractional batchwise removal, f, occurs at the steps when Δt is a multiple of T_{cyc}, removing the need for scaling. This method is called Bulk batchwise reprocessing because it performs reprocessing all at once in a bulk manner. This approach was implemented in the early versions of SaltProc [39].

$$\gamma = f \frac{\Delta t}{T_{cyc}} \tag{2.2}$$

One example of implementing bulk batchwise reprocessing can be seen in the depletion simulation of the Molten Salt Breeder Reactor using SaltProc [39]. This simulation uses bulk batchwise reprocessing every three days with a linear approximation of the cycle times given by Robertson et al. [34, 39]. The xenon and krypton removals in this work use a gas separation system equation, while the protactinium uses a modeled liquid-liquid reductive extraction process. These efficiencies differ from the cycle time values, causing slight differences in those terms. The MSBR also has two varying feed inputs of ^{232}Th and ^{233}U.

2.2.1.2 Continuous Reprocessing

The continuous reprocessing method involves directly modifying the Bateman equation, shown in Equation (2.3). Within this method, there are different approaches. For example, Serpent2 offers three different approaches, including a decay-like term, a constant term over each depletion time step, and a constant term which does not conserve mass. These approaches will be referred to as "decay", "step", and "constant".

One of the continuous reprocessing methods used in Serpent2 continuous reprocessing, an extension developed by Aufiero et al, is a decay-like approach and is referred to here as "decay" continuous reprocessing [4]. Equation (2.3) shows the Bateman equation with production and loss exclusively from fission, absorption, and decay; and Equation (2.4) shows the Bateman equation with the removal from reprocessing subtracted and addition from feed sources added.

$$\frac{dN_j}{dt}_{base} = \sum_{i \neq j} \left[\left(\gamma_{i \to j} \sigma_{f,i} \Phi + \lambda_{i \to j} + \sigma_{i \to j} \Phi \right) N_i \right] - \left(\lambda_j + \sigma_j \Phi \right) N_j \tag{2.3}$$

$$\frac{dN_j}{dt}_{net} = \frac{dN_j}{dt}_{base} - \lambda_{r,j} N_j + \sum_{feed} \lambda_{f,j} N_j \tag{2.4}$$

The symbols given in the equations are defined as follows [23]:

N_j is the atomic density of isotope j $\left[\frac{atoms}{cm^3} \right]$.

$\gamma_{i \to j}$ is the fractional fission product yield of j in the fission of isotope i.

$\sigma_{f,i}$ is the microscopic fission cross section of isotope i $\left[cm^2 \right]$.

Φ is the spectrum-averaged scalar flux in the fuel region $\left[\frac{neutrons}{cm^2 s} \right]$.

$\lambda_{i \to j}$ is the decay constant of decay $i \to j$ $\left[s^{-1} \right]$.

$\sigma_{i \to j}$ is the microscopic transmution cross section of reaction $i \to j$ $\left[cm^2 \right]$.

N_i is the atomic density of isotope i $\left[\frac{atoms}{cm^3} \right]$.

λ_j is the decay constant of isotope j $\left[s^{-1} \right]$.

$\lambda_{r,j}$ is the reprocessing constant for removal of isotope j $\left[s^{-1} \right]$.

σ_j is the microscopic total transmutation cross section of isotope j $\left[cm^2 \right]$.

$\lambda_{f,j}$ is the feed constant for feed of isotope i $\left[s^{-1} \right]$.

Equation (2.4) shows that the reprocessing removal has the same mathematical operation as the decay rate. Unlike decay, the isotopes removed by reprocessing instead are transfered to a different material, which operates as a feed for that material. This can be seen in the summation term, which sums over the different feeds. Figure 2.1

shows how these reprocessing and feed constants fit into the reactor operation. Additionally, the figure shows that these terms relate to reprocessing and feed mass flow rates of $\dot{m}_{r,j}$ and $\dot{m}_{f,j}$, respectively.

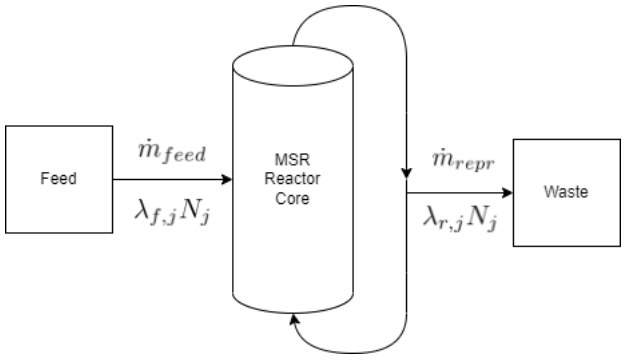


Figure 2.1: Simplified model of how Bateman equation relates to mass flow rates.

The constant approach does not conserve mass and will not be used in this work. Overall, it operates similarly to the decay approach by adding a constant term, but any negative terms become zero. This setting is intended for refueling.

The decay approach, which adds a decay-like term to the Bateman equation, is shown in Equation (2.4). This approach of adding a decay-like term to the Bateman equation has been used with other codes such as SCALE and MCNP [21, 35]. For this setting, three different approaches are implemented in order to compare their results, and are discussed later in this work. This setting is the most important for this work, and is the only setting implemented.

The step continuous reprocessing approach used in Serpent2 removes a constant value from each isotopic Bateman equation. Since the chemistry provides a removal rate for elements, the straightforward approach would be to generate a removal rate for each element. This approach can be seen in Equation 2.5, where the C term represents the constant value being removed by continuous reprocessing.

$$\frac{dN_j}{dt} = \sum_{i \neq j} \left[\left(\gamma_{i \to j} \sigma_{f,i} \Phi + \lambda_{i \to j} + \sigma_{i \to j} \Phi \right) N_i \right] - \left(\lambda_j + \sigma_j \Phi \right) N_j - C \tag{2.5}$$

This approach has a flaw which can be directly demonstrated through an example if the reprocessing constants are defined for an element and not for each isotope, shown in Table 2.2. Imagine an element exists with two isotopes α and β. There is a removal rate of 10% per second defined for the element. A batchwise process could remove 10% of each isotopes relative abundance after a second by using a one second time step, which is physical in terms of fractional removal. This can be seen directly in Table 2.2, where 10% of both isotopes and the net count are removed in the "Batch" column.

For the continuous reprocessing, both isotopes use the same constant removal rate C. To perform reprocessing, the first attempt might be to try and remove 10% of the net by splitting it amongst the isotopes evenly, as shown in

Table 2.2: Serpent2 Step Approach Example Results

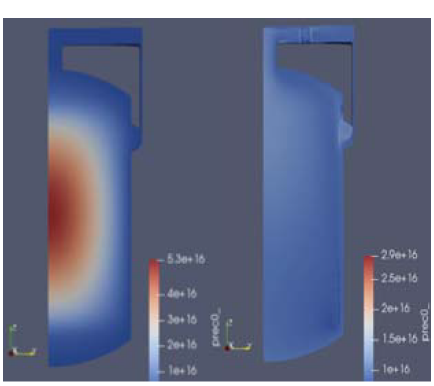
Labels	Initial	Batch	Both Fit	α Fit	β Fit	Zeroing Fit
α [g]	1000	900	949.5	900	999	990
β [g]	10	9	-40.5	-90	9	0
Net [g]	1010	909	909	810	1008	990

the "Both Fit" column. The issue with this approach is immediately noticeable, which is that the concentration of isotope β becomes negative. A logical next approach would be to try and make one of the isotopes exactly correct while bringing the other along with it, which can be seen in the α and β Fit columns. The results of this approach are seen in Table 2.2, and it can be seen that only by looking at the isotope with the smallest concentration and using that to determine the amount to remove can the result be ensured to be non-negative. One final approach to consider is to set the smallest concentration to 0, and use the same constant removal rate for the other isotopes. This approach can be seen in the "Zeroing Fit" column, and it also does not work very well.

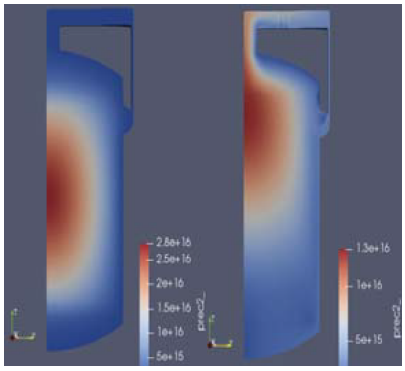
This analysis shows that, for this reprocessing approach to function properly, the reprocessing constants would need to be generated for every isotope individually in order to establish an online reprocessing scheme. However, this would require collecting data on the quantity of each isotope and calibrating reprocessing constants at every depletion time step. Stopping the depletion simulation to collect this data and then restarting would add computational cost overhead to this approach, reducing its applicability in this situation.

2.2.2 DNP Movement

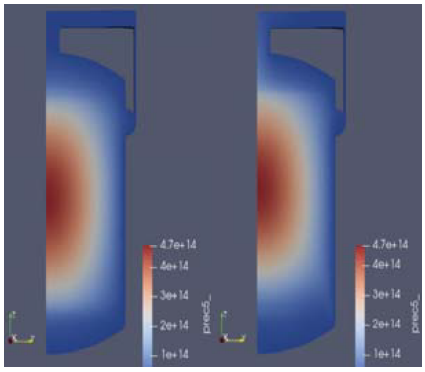
The movement of DNPs in liquid fueled molten salt reactors impacts the operation of the reactor [46, 3]. There are several effects which this movement can have on the reactor. As the core salt recirculates, there is a probability of fission product decay in non-core regions of the reactor, meaning there are fewer delayed neutrons in the core. This results in a reduced effective delayed neutron fraction, which reduces controllability of the reactor. Additionally, as delayed neutrons have a softer spectrum than prompt neutrons, reactors with recirculating fuel will have a spectrum-hardening effect. In a solid fueled reactor, the flux and the concentration of delayed neutron precursors have the same profile. However, a moving fluid fuel causes the precursors to have a shifted profile based on the fluid flow rate, mixing, and decay rate of the precursors. This was shown by Jun Shi and Massimiliano Fratoni in their work, which is illustrated in Figure 2.2 [43].



(a) Longest lived delayed neutron precursor group.



(b) Third longest lived delayed neutron precursor group.



(c) Shortest lived delayed neutron precursor group.

Figure 2.2: Plots of DNP concentrations in the molten salt reactor experiment [43]. The left side of each image shows concentration with no flow, while the right shows the concentration with a 1200 gallon per minute flow rate.

Figure 2.2 shows that for the shortest lived DNPs, the movement of the fuel does not have any significant impact upon the DNP distribution when compared to static fuel. This is reasonable since the precursors do not live for very long in this group, so they do not have much chance to travel. For the intermediate DNP group, there is a clear shift in the direction of the fuel flow, which causes more precursors to produce delayed neutrons in the less important piping and upper plenum regions. The longest lived DNP group seems to be spread almost equally throughout the entire reactor, which means that those delayed neutrons are contributing significantly less to the overall neutron economy. This reduces the overall effective delayed neutron fraction since the delayed neutrons are in less important regions. This directly affects the controllability of the reactor through the reactor period, which is heavily dependent upon the delayed neutrons.

2.2.3 Depletion

Depletion is the process of burning the fuel, calculating the changes to the material composition this causes. Simulating depletion requires specific information to accurately model the depletion process, such as decay

constants, fission yields, and fission and transmutation cross section data. The fission and transmutation cross section data can come from a transport calculation, while the other data can be read from a data library [23]. Running depletion entails numerically solving the Bateman equations; a generalized form of which is in Equation 2.3.

The matrix form can be seen in (2.6), where N is a vector of compositions, and A is the depletion matrix. This matrix contains the gain and loss terms for each nuclide in N, which vary continuously as a function of time and depend on the concentrations as well. Further documentation on treatment of depletion of these terms can be found in the OpenMC documentation, an open source Monte Carlo transport code which has built-in depletion solvers [36].

$$\frac{dn}{dt} = An \tag{2.6}$$

There are several different approaches to solving this equation, such as the Transmutation Trajectory Analysis (TTA) method and the Chebyshev Rational Approximation Method (CRAM). Both of these methods are built into Serpent2 [24]. Other codes may use external software to compute depletion, such as REM, an in-house code used by Nuttin et al. [29] to implement Fourth order Runge–Kutta, used by MCNP and ORIGEN-S, a modified form of ORIGEN [18] to treat time dependent data libraries which uses a power series approximation to treat the depletion matrix exponential, used by KENO-VI [4]. Another code that contains built-in depletion solvers is ERANOS, which is a diffusion transport solver which also uses a power series approximation to handle the depletion solution generation [4]. Overall, there are many different approaches to solving the depletion equation with different computational costs and levels of accuracy. Building a code to handle depletion requires only two items. The first is an input of data to generate the depletion matrix, including flux, cross sections, branching ratios, fission yields, decay constants, and more, depending on the level of detail desired. The other part of depletion is numerically solving the matrix problem, which may be coupled for different materials depending on if materials interact with each other.

The purpose behind solving these equations and having depletion models is to generate an accurate representation of the composition of the target materials during reactor operation. This is important in determining how long the reactor can run, how the safety of the reactor develops over time, and how operation may have to change to adjust to the new reactor state. For molten salt reactors, depletion also allows for information regarding fresh fuel feed rate and fission product removal rates, as well as how variations of those parameters can affect reactor performance and behaviour.

2.3 The Molten Salt Breeder Reactor

In this work, I will apply the previously described methods to the Molten Salt Breeder Reactor (MSBR). The MSBR is a useful design to analyze for several reasons. Because it is a liquid-fueled molten salt reactor, online reprocessing is used within the design. In addition, it contains an inner and outer zone, zone I and II, respectively, each of which has different neutronic behaviour [34]. A portion of the core geometry is shown in Figure 2.3. Of note in this figure are the inner and outer core zones, with explicit differences in neutronic behavior. More specifically, the inner zone has a softer spectrum, a higher fission rate, and a 13% fuel-to-graphite ratio [30]. The outer zone has a harder spectrum, a higher breeding rate, and a 37% fuel-to-graphite ratio. The differences between the inner and outer zones prove to be an issue when accurately modeling using a unit-cell or one-region approach, as those models are unable to capture the different characteristics of each region [39]. The geometric differences in the regions can also be seen in Figure 2.3, which allows for the difference in the fuel-to-graphite ratio to be visually noticeable.

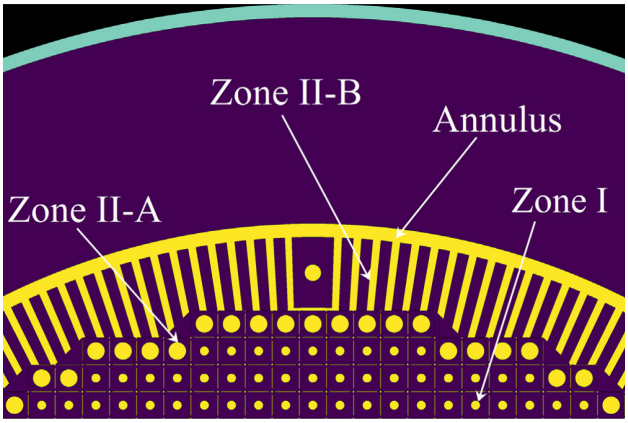


Figure 2.3: MSBR core axial slice showing the different regions from [39]. Zones II-A and II-B are where the spectrum is harder and there is increased breeding. Zone I is where there is more fission and a softer spectrum. Zone I is made up of smaller fuel channels, Zone II-A has larger fuel channels, and Zone II-B has long sections where fuel flows. The yellow is fuel salt, the purple is graphite, and the cyan is the reactor vessel.

Table 2.3 has physical specifications on the MSBR, while Table 2.4 has specific information on the distribution of the fuel salt during operation. Some parameters of note in Table 2.3 are the salt inventory cycle time is 10 days, which means that the entire salt inventory is cycled in 10 days. However, the loop cycle time, or time it takes for the salt to complete a full loop in the primary system, is 11 seconds [34]. Some other data of note in the table are the thermal capacity and fuel salt volume. The thermal capacity is the power which is used during depletion simulations when generating results. The fuel salt volume for the primary system is the irradiated salt volume used in this work. The other data in the table are useful parameters for understanding the properties of the reactor, such as the pressure, height, diameter, flow velocity, and salt fractions. The salt fractions represent the fraction of cross sectional area in each region which is composed of fuel salt.

Table 2.3: Summary of Principle Data for MSBR [34]

Component	Data
Thermal Capacity	2250 MW_{th}
Vessel Inner Diameter	6.77 m
Core Height	3.96 m
Vessel Pressure	0.52 MPa
Central Core Zone Salt Fraction	0.13
Outer Core Zone Salt Fraction	0.37
Maximum Flow Velocity (Core)	2.6 $\frac{m}{s}$
Fuel Salt (Vessel)	30.4 m^3
Fuel Salt (Primary System)	48.7 m^3
Thorium Inventory	68,100 kg
Breeding Ratio	1.06
Processing Rate	1 gpm
Salt Inventory Cycle Time	10 days
Salt Components	$^7LiF-BeF_2-ThF_4-UF_4$
Salt Composition	71.7-16-12-0.3 mole %
Salt Density	3.35 $\frac{g}{cm^3}$

Table 2.4 shows that the salt inventory in the core is approximately 19 m^3, with 8.2 m^3 from zone I and 10.8 m^3 from zone II. The table also shows that the total volume of fuel salt in the reactor is 30.4 m^3.

Table 2.4: MSBR Primary System Salt Inventory [34]

Region	Volume [m^3]
Reactor	
Core Zone I	8.2
Core Zone II	10.8
Plenums, Inlets, Outlets	6.2
Annulus	3.8
Reflectors	1.4
Primary Heat Exchangers	
Tubes	7.6
Inlets, Outlets	0.8
Miscellaneous	
Pump Bowls	5.2
Piping (including drain line)	4.1
Off-gas bypass loop	0.3
Tank heels and miscellaneous	0.3
Total	
Fuel Salt (Primary System)	48.7

Table 2.5: MSBR Online Reprocessing Cycle Times [34]

Reprocessing Group	Element(s)	Cycle Time (Full Power)
Rare Earths	Y, La, Ce, Pr, Nd, Pm, Sm, Gd	50 days
Rare Earths	Eu	500 days
Noble Metals	Se, Nb, Mo, Tc, Ru, Rh, Pd, Ag, Sb, Te	20 seconds
Seminoble Metals	Zr, Cd, In, Sn	200 days
Gases	Kr, Xe	20 seconds
Volatile Fluorides	Br, I	60 days
Discard	Rb, Sr, Cs, Ba	3435 days
Salt Discard	Th, Li, Be, F	3435 days
Protactinium	Pa	3 days
Higher Nuclides	Np, Pu	16 years

Shown in Table 2.5 are the cycle times for processed elements in the MSBR. Here, I define the cycle time as the time it takes for a given element to be completely removed, which is a common way to define the cycle time in the literature [8, 39, 34].

The fuel processing of the MSBR is defined to operate continuously with processing methods [34]. These processes are not explicitly modeled in this work, but rather are approximated using a single irradiated salt volume with continuous or batchwise reprocessing methods. It is still useful to understand the process of simplification from actual flow and chemical processing to cycle time values implemented into reprocessing methods.

For protactinium removal, fluorination is first used to remove uranium before protactinium isolation. Next is countercurrent bismuth with lithium and thorium for stripping any remaining uranium and the protactinium from the bismuth. This is followed by hydrofluorination to separate the uranium and protactinium. This process can be seen in Figure 2.4, which shows a flow diagram of the protactinium processing scheme.

For a 10 day protactinium cycle time, a fuel salt flow rate of 0.88 gallons per minute (gpm) is used. The processed salt is processed for rare earth fission products before being fed back into the reactor.

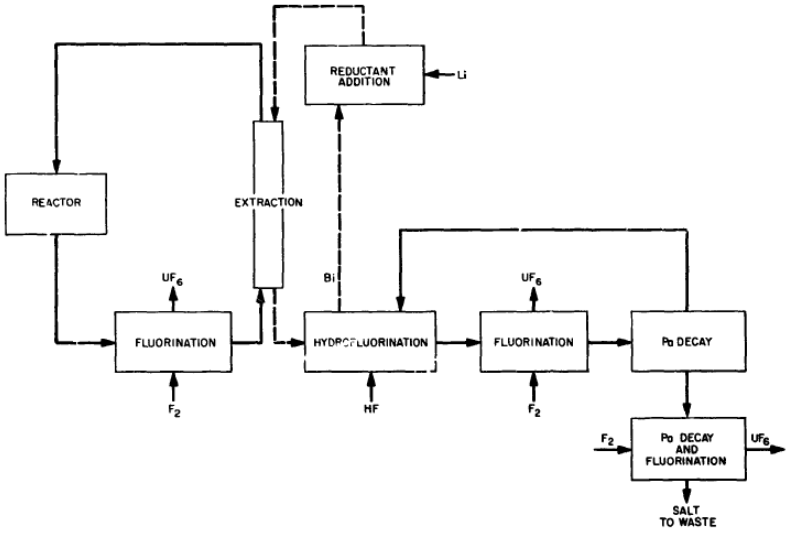


Figure 2.4: MSBR protactinium processing scheme from Robertson et al. [34].

Rare earth fission product removal is performed using the metal-transfer process, which uses lithium chloride and bismuth containing a reductant. More specifically, bismuth containing a reductant of thorium and lithium is used to strip the rare earth fission products from the fuel salt. The rare earth fission products are then transported to the lithium chloride acceptor salt, though lithium bromide or a mix of both could be used. Figure 2.5 shows this process, and Figure 2.6 shows the protactinium and rare earth processing schemes together.

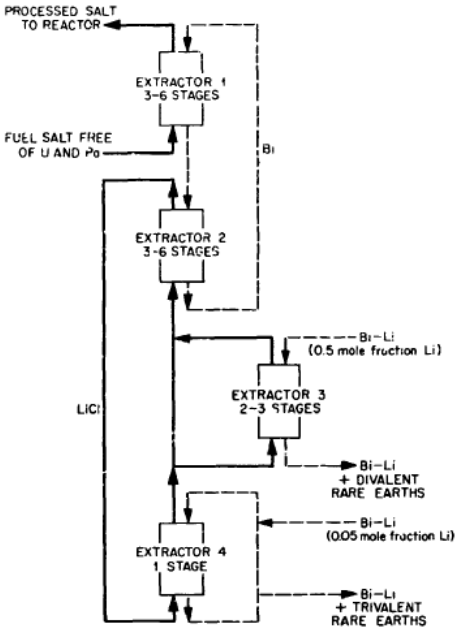


Figure 2.5: MSBR rare earth processing scheme from Robertson et al. [34].

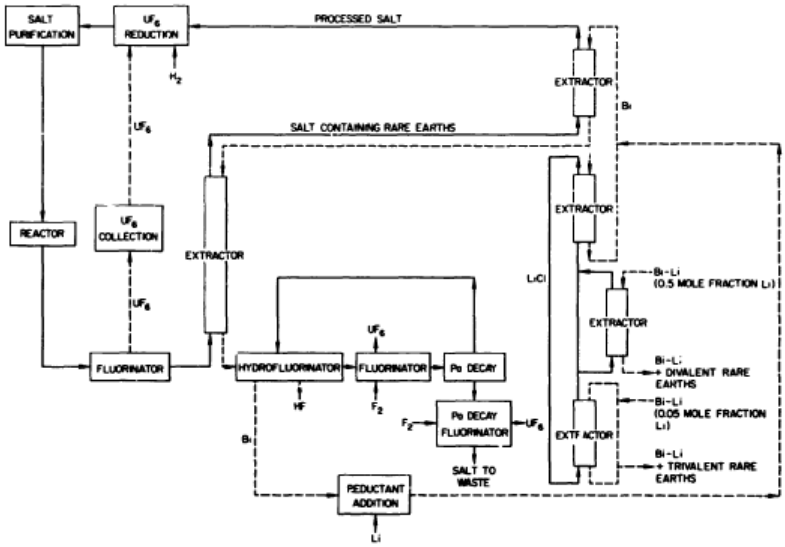


Figure 2.6: Combined MSBR rare earth and protactinium processing schemes from Robertson et al. [34].

To strip gaseous fission products, a bubble generator introduces helium sparging gas in 15 to 20 mil bubbles. 10 percent of the flow is redirected to the bypass loop which contains a gas separator. This gas separator strips the fission products with approximately 100 percent efficiency. The off-gas system also includes an approximately two hour holdup during which noble metals deposit on the fuel salt drain tank surface. Figure 2.7 shows this process, beginning with the number per time, or flux, of the noble gasses in the salt migrating to the graphite voids and helium sparging bubbles. This is followed by the volume holdup into more processing to handle the noble metal fission product cleanup.

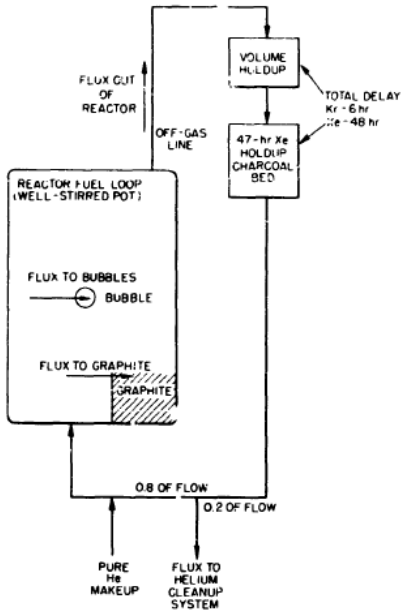


Figure 2.7: MSBR noble metal flux and off-gas system from Robertson et al. [34].

2.4 SaltProc

2.4.1 MSBR Model

SaltProc comes with a full 3D model of the core of the MSBR created in Serpent2. The MSBR model provided in SaltProc uses the volume of the primary system, 48.7 cubic meters, for its calculations rather than the 19 cubic meters of the core [39]. This is an approximation which allows the depletion simulation to use the correct amount of net material, though it also means that material which would not be irradiated in the piping is present in the core. The core geometry uses the correct dimensions, which allows cross sections to remain as if the correct core volume were implemented.

2.4.2 Reprocessing

SaltProc is used to handle batchwise reprocessing in this work because it uses Serpent2, which has built-in continuous reprocessing; comes with the MSBR geometry in Serpent2 as an example; and has multiple versions. Of these different versions of SaltProc, they have slightly different methods in how the depletion data is stored as well as how reprocessing rates are calculated. However, the two versions of interest are version 0.1 and 0.3, which use different batchwise reprocessing approaches. These two different approaches are referred to here as the "Bulk" approach and the "Steady" approach.

SaltProc version 0.1 was the first full version of SaltProc publicly released, and provides all the functionality necessary for modeling the MSBR with batchwise reprocessing. Using this version, Rykhlevskii uses three day depletion steps and does not perform fractional reprocessing at each step, instead performing reprocessing according to the cycle times [37]. This approach is termed as the "Bulk" approach to batchwise reprocessing, and is explained in more detail below. This is the same approach used by Gehin and Powers for an analysis of the MSBR [15]. This version of SaltProc, version 0.1, makes use of the h5py Python package to handle the hdf5 data files which contain depletion data [11]. Within the hdf5 file, the atom density of each isotope before and after batchwise reprocessing is recorded as well as different reactor parameters, such as the neutron multiplication factor.

SaltProc version 0.3 includes an example MSBR which users can run as soon as SaltProc is installed, though some of the run parameters such as neutrons per generation and run time have to be altered to generate a more valid model. This model also uses three day steps, but instead of performing the batch removal only at the cycle time value, this model removes a fraction every three days, which is called the "Steady" approach for batchwise reprocessing. Additionally, this version of SaltProc uses the PyTables Python package for handling the hdf5 data files. The data files are structured similarly to v0.1 of SaltProc, but instead of atom density they use mass, which makes analysis of the files slightly more user friendly.

2.5 MSR Modeling Approaches

For modeling of MSRs, it is important to model discrete behavior, without considering fuel depletion, and temporal behavior, where fuel depletion is considered, in core physics. With discrete modeling, one may aim to focus on aspects such as DNP movement [13, 43] or reactor physics, such as kinetics and dynamics [45, 10, 5, 12, 44]. This type of model typically is focused on the order of seconds.

Temporal modeling typically focuses on time scales ranging from days to years, such as fuel depletion. Because the online reprocessing of an MSR has such large effects on its fuel composition during a long depletion simulation, these depletion simulations of MSRs typically incorporate a method to simulate online reprocessing by using batchwise or continuous online reprocessing methods. These methods do not add any additional physics to the depletion simulation, but modify the compositions either with an additional term in the system of equations or by multiplying elements by a factor based on the cycle time of that element and the time step used. These simulations can also account for other factors, such as DNP movement, as shown by Zhou et al. [48]. The movement of DNPs can have a noticeable effect on reactor performance by affecting neutron energy spectra and localized reactivity effects, though it has been shown they do not have an impact when looking at longer time scales [7]. Accounting for coupled full core fluid flow and performing depletion in a single simulation would be exceedingly computationally expensive.

Another desired component of MSRs to model is the ex-core piping, where one method used to approximate spatial dependence of fuel salt without modeling the ex-core region explicitly is the scaled flux method [6]. Most MSR depletion simulations, however, focus primarily on depletion and do not consider the DNP movement during depletion in-core or ex-core.

Table 2.6 contains many different depletion simulations of MSRs which have approximated online reprocessing. Table 2.6 shows which MSR depletion simulations have modeled DNPs in the "DNP" column, which shows most models which do not include DNP movement. The table is sorted by the "Reprocessing" column, which signifies the method(s) used in the different MSR models and is also sorted by the publication year in the "Year" column. The table shows that there are many different MSRs, shown in the "Reactor" column, which have been modeled using a variety of spatial refinement levels, shown in the "Model" column, ranging from unit cells to full 3D models. For all of the continuous reprocessing models given in Table 2.6, all of them use the "fictitious decay constant" approach, which is shown in Equation (2.4). For the batchwise models, most use the Steady batchwise approach shown in Equation (2.2) and remove a fraction at each depletion step rather than bulk removal at certain steps.

A key takeaway from Table 2.6 is that batchwise reprocessing methods are commonly used to a similar extent as continuous reprocessing methods to handle MSR online reprocessing. This is important because, as previously explained, the batchwise reprocessing methods are a less accurate way to simulate online reprocessing for a physically continuous scheme. Therefore, because it is still common for researchers to use this method, it is important that the differences between continuous and batchwise methods are better understood.

In Sections 2.5.1, 2.5.2, and 2.5.3, I will discuss the results of each of the studies through the lens of the current work. That is, I will summarize the effects of using those depletion and reprocessing methods and note what differences may be observed from the literature. Then, in Section 2.5.4, I will comment on the gaps that this work will address.

Table 2.6: Molten Salt Reactor Models Analyzed with Methods Approximating Online Reprocessing

Publication	Reprocessing	Model	Reactor	DNP	Year
Valdez et al. [21]	Continuous	Unit Cell	N/A	No	2020
Zhuang et al. [49]	Continuous	Full 3D	MSFR	No	2020
Xia et al. [47]	Continuous	Full 3D	N/A	No	2019
Zhou et al. [48]	Continuous	Full 3D	MSBR	Yes	2018
Aufiero et al. [4]	Continuous	Full 3D	MSFR	No	2013
Rodriguez-Vieitez [35]	Continuous	Unit Cell	ADNA	No	2002
Rykhlevskii [39]	Batchwise	Full 3D	MSBR	No	2019
Betzler et al. [8]	Batchwise	Unit Cell	MSBR	No	2017
Ridley and Chvala [33]	Batchwise	Full 3D	Toy	No	2017
Ahmad et al. [1]	Batchwise	Unit Cell	DMSR	No	2015
Park et al. [30]	Batchwise	Full 3D	MSBR	No	2015
Sheu et al. [42]	Batchwise	Full 3D	MOSART	No	2013
Nuttin et al. [29]	Batchwise	Full 3D	MSBR	No	2005
Fiorina et al. [14]	Mixed	2D Slice	MSFR	No	2012
Nagy et al. [28]	Mixed	Unit Cell	Toy	Yes	2008
Merle-Lucotte et al. [27]	Mixed	Full 3D	TMSR	No	2007
Bauman et al. [17]	Mixed	2D Slice	N/A	Yes	1971

2.5.1 Continuous Reprocessing

The first subgroup of studies shown in Table 2.6 shows that many different codes have been used to simulate continuous reprocessing in MSRs. One common aspect between all of them is that they use long depletion step sizes (on the order of months) and long net depletion times (on the order of years).

Valdez et al. [21] demonstrates the efficacy of the updated version of ChemTriton. The version was added after TRITON implemented new tools for continuous material feeds and removals. Valdez et al. discusses in depth how

the removal rates for isotopes can be calculated using a well mixed approximation and accounting for non-modeled piping by introducing a correcting decay factor. Although it does not account for DNP movement, it does account for neutron important regions without having to model the piping regions.

Zhuang et al. [49] discusses an OpenMC extension called OpenMCB-MSR to implement continuous reprocessing in an analysis of the MSFR. The depletion step sizes used are not specified in the work, though they appear to be on the order of decades, with a final depletion time of 200 years.

Xia et al. [47] demonstrates the MOlten salt reactor specific DEpletion Code (MODEC). This code uses continuous reprocessing, and has been used to analyze the Molten Chloride salt Fast Reactor (MCFR) by Liaoyuan et al. [25]. The MCFR analysis uses SCALE6.1 for transport and MODEC for depletion.

Zhou et al. [48] covers FAMOS (Fuel cycle Analysis code for MOlten Salt reactors). This code uses OpenMC to generate homogenized cross section data followed by the DIF3D diffusion solver along with extensions to model DNP movement, thermal hydraulics, depletion, and continuous reprocessing. Zhou et al. analyzes the molten salt fast reactor over 200 operational years and the molten salt breeder reactor for 20 years. The depletion step size is not specifically given in the work, but from analysis of the plots, small time steps seem to have been used at the beginning of cycle and then larger time steps of 100 days are then used for the rest of the steps.

Aufiero et al. [4] demonstrates the Serpent2 newly developed continuous reprocessing functionality. A full 3D simplified core layout of the molten salt fast reactor (MSFR) is used, and continuous reprocessing is used to operate the removal and feed processes. It evaluates the model at various scales, from steady state uranium concentrations at over 60 years to radiotoxicity over 10^8 years. The step sizes vary as well, though not specifically mentioned, appearing very small at beginning of cycle and expanding to around 10 years at steady state for one of the figures presented.

Rodriguez-Vieitez [35] analyzes the Accelerator-Driven Neutron Applications (ADNA) Tier-1 reactor using MCNP. Although the model Rodriguez-Vieitez used does implement continuous reprocessing in the Bateman equation, it is not used for the purpose of fuel cycle analysis and depletion. Rather, the work iteratively works towards the equilibrium fuel composition in order to analyze the effect of varying different reactor parameters, such as geometry.

2.5.2 Batchwise Reprocessing

The second category of studies summarized in Table 2.6 use batchwise reprocessing. This method uses depletion step sizes on the order of several days, and uses net depletion times on the order of years.

Rykhlevskii [39] uses SaltProc to simulate the MSBR. Rykhlevskii uses Serpent2, a full 3D core of the MSBR, three day time steps, and runs for 60 years. Rykhlevskii calculated the average thorium-232 refueling rate, compared safety parameters at beginning of cycle and equilibrium, and collected other useful data for understanding and

further simulating the MSBR. The batchwise reprocessing in SaltProc includes both the removal and feed processes.

Betzler et al. [8] uses ChemTriton, a tool designed for SCALE to perform online reprocessing of MSRs. It contains work on multiple reactors, such as the MSBR and the MSRE. The tool iteratively runs SCALE/TRITON over small time steps to simulate continuous reprocessing, and includes enhanced time-dependent feed and separations. Betzler et al. use this tool to optimize the thorium concentration to maintain a critical configuration while maintaining breeding capability, show the impact of fission product removal on core lifetime and fuel utilization, and more. For the MSBR case, Betzler et al. uses three day time, similar to Rykhlevskii and Park et al. This time step is selected because previous parametric studies showed that it is sufficiently short enough to model the online removal of protactinium [32].

Ridley and Chvala [33] use batchwise reprocessing on a toy MSR using Serpent2 and a custom in-house python library designed to adjust feed rates and volumes in Serpent2. The feed rates were set to fluctuate in order to maintain criticality while also accounting for density imbalance in Serpent2 by increasing the volume of the material. The full core 3D model was then analyzed using seven day depletion steps for a reactor operating period of ten years.

Ahmad et al. [1] analyzes single fluid and two fluid denatured molten salt reactors (DMSRs). Ahmad et al. use MCNP5 and ORIGEN2 as the codes to perform transport and depletion over 30 years. Although the steps at which fuel is adjusted is at five year intervals, the burnup intervals are set to six months. The reason such large steps were used is because the work analyzes proliferation and resource requirements for the reactors. This means that the neutronics and safety parameters were not the focus of the study. Additionally, only gaseous fission products and fission products removed via sparging are included in the reprocessing scheme. No other chemical extraction takes place.

Park et al. [30] is very similar to the MSBR example in SaltProc, with the primary difference being MCNP6 is used for transportation calculations instead of Serpent2. The depletion was performed using CINDER90 and custom Python scripts. Another difference is that SaltProc uses adjusted values for the removal efficiency rates of xenon, krypton, and protactinium, while the work by Park et al. uses the cycle time values from Robertson et al. [34]. This model uses three day time steps and models the fuel composition over 20 years.

Sheu et al. [42] analyzes the Molten Salt Actinide Recycler & Transmuter (MOSART) reactor and uses custom scripts to implement batchwise reprocessing. The codes used are SCALE6 for transport and TRITON for depletion. The step sizes used include 15, 30, and 60 days, in which the authors found that 30 days gave the best results in terms of accuracy and run time balancing for the 30 year simulation.

Nuttin et al. [29] analyzes the MSBR using a slightly simplified geometry with MCNP. The reprocessing approach used here is also the same as the approach used in SaltProc, which is to take the inverse of the cycle time to

determine the removal rate. Depletion is performed in this work by coupling MCNP to an evolution program and the NJOY code. However, Nuttin et al. use ten day steps instead of three day steps that SaltProc uses. It also runs for 100 years instead of 20.

2.5.3 Mixed Batchwise and Continuous Reprocessing

The main reason why a combination of batchwise and continuous reprocessing methods would be used in tandem is for treatment of a reprocessing scheme which has some elements removed in batch steps rather than continuously. Including batchwise methods for those elements would allow for a more realistic model. Another reason could be if mass balancing is an issue for the model. In that case, a batchwise method could be added to ensure the net mass remains constant in the core by removing or adding material as needed.

Fiorina et al. [14] is on the Molten Salt Fast Reactor (MSFR) and uses a modified version of the ERANOS-based EQL3D procedure. Fiorina et al. make use of continuous reprocessing for non-soluble fission products, but uses batchwise reprocessing for soluble fission products. The batchwise reprocessing was incorporated in order to more closely match a simpler model for comparison [26]. The time steps used are shorter at the beginning of cycle and extend out to roughly 30 year steps for a 200 year depletion. Additionally, the authors also performed a radiotoxicity analysis which extends out to one million years.

Nagy et al. [28] uses a toy model to go through different reprocessing schemes based on the MSBR and MOSART reprocessing designs [34, 20]. SCALE-5 XSDRN[31] is used for neutron transport and ORIGEN-S is used for depletion. To account for DNPs, the authors assume that one-third of the delayed neutron precursors decay outside the core. Using a delayed neutron fraction of 0.0027 for ^{233}U, the authors claim that 0.0009, or 90 pcm, should be subtracted from the eigenvalue to approximate the impact from the movement of delayed neutron precursors. Depletion steps are performed every five days, during which continuous reprocessing was implemented for only gaseous fission product removal, while refueling and other fission products used batchwise reprocessing. The batchwise reprocessing was executed by altering the binary output of ORIGEN.

Merle-Lucotte et al. [27] covers depletion of the Thorium Molten Salt Reactor (TMSR) using MCNP4 [16] for transport and REM for material evolution. Continuous reprocessing is used for gaseous fission product extraction, while the other fission products are processed with batchwise reprocessing.

Bauman et al. [17] presents the Reactor Optimum Design (ROD) code for circulating-fuel reactors. ROD allows users to use continuous or batchwise reprocessing for MSRs. ROD operates by taking a one or two dimensional input and determining the equilibrium fuel composition. This is a useful code for determining a general prediction for a model, but is limited to one or two dimensional problems, a diffusion solver, 15 energy groups, and 200 fission products.

2.5.4 Discussion

Overall, from these works it can be seen that using continuous reprocessing allows for larger time steps, which in turn allow for decreased computational cost, which enables the studies that use continuous reprocessing to simulate results over a longer time period. However, continuous reprocessing is more complex to implement than batchwise reprocessing.

For codes which do not have continuous reprocessing built-in, authors will find it more efficient to use a batchwise method which can be handled externally rather than implement continuous reprocessing into a code. This is particularly the case if the authors have a time constraint and need to generate results quickly. Implementation of continuous reprocessing into an existing code is a work in itself, as shown by authors such as Aufiero et al. and Zhuang et al.

Thus, unless every depletion code used in MSR modeling is updated to handle continuous reprocessing, batchwise approximations will continue to be implemented. Because this approximation is common and is likely to continue in the future, it is of increasing importance to determine the impacts of this approximation. It is well understood that smaller step sizes are required for batchwise methods to approximate continuous methods, but there has yet to be a full analysis on how the results for an MSR using the two different methods compare. This is the gap that this work seeks to fill. I will show how the eigenvalue and various isotopic compositions vary while altering only the reprocessing methods implemented.

Chapter 3

Methodology

In the literature review, I showed how various papers have employed different codes, reactors, reprocessing methods, and tracking of DNPs. I also discussed how a reason for employing both continuous and batchwise reprocessing could be to ensure the mass in the reactor remains fairly constant. In this chapter, I will discuss the methodology used for the Serpent2 MSBR model used to generate results, the different reprocessing methods, mass balancing, and delayed neutron precursors in this work. More specifically, I will describe in detail first two different approaches for batchwise reprocessing which have been implemented in SaltProc, the tool used in this work to handle batchwise reprocessing [2]. I will then give a summary of the two approaches and provide estimates for the expected error and computational costs compared to continuous methods. After that, I will then discuss three different continuous reprocessing methods. I will discuss several approaches which can be implemented for one of these methods. Finally, I will provide information on the mass balancing and the effects of delayed neutrons on depletion results.

3.1 Serpent MSBR Model

The results generated in this work come from the MSBR, particularly the model developed by Rykhlevskii [37]. The geometry of the model has remained unchanged, while the reprocessing has been overhauled using the continuous reprocessing functionality in Serpent2 which was previously undocumented. Previously, SaltProc's batchwise reprocessing was used, but the continuous reprocessing in Serpent2 is incorporated in this work.

3.1.1 Reprocessing Structure

There are two different reprocessing schemes developed in this work. Both schemes follow the MSBR reprocessing scheme, and vary how refueling functions. The first method matches the method of Rykhlevskii, where the uranium-233 feed is treated as equivalent to the protactinium-233 removal rate. In this case, the average uranium-233 feed rate from Rykhlevskii is used over the entire simulation time [37]. A simplified overview of this scheme can be seen in Figure 3.1.

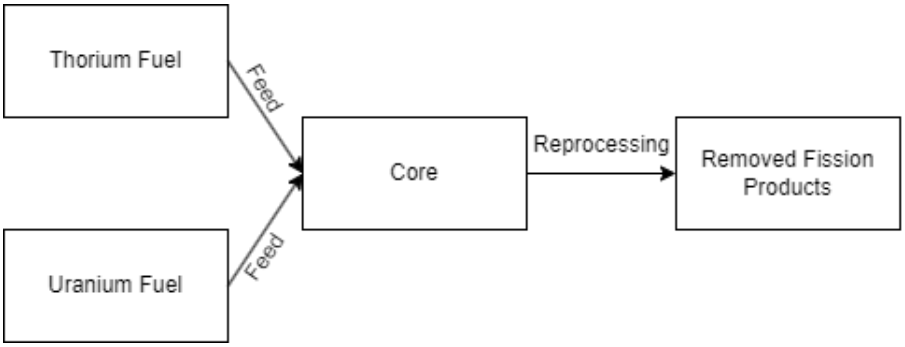


Figure 3.1: Simplified reprocessing scheme based on SaltProc.

The second reprocessing scheme reflects the physical MSBR model, and instead uses a protactinium decay tank. From this tank, any bred uranium is continuously removed and sent back to the core, which is the intended design scheme for the MSBR [34]. A simplified version of the phyiscally realistic uranium feed can be seen in Figure 3.2.

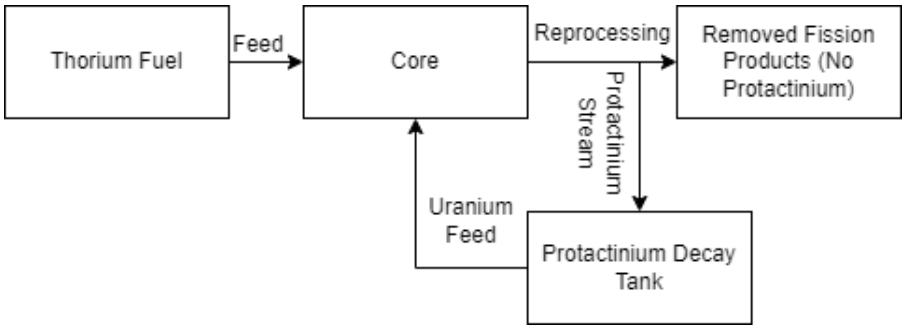


Figure 3.2: Simplified reprocessing scheme based on the phyiscal MSBR processes.

Overall, it can be anticipated that the two methods will be approximately equal at steady state, since at that point the assumption by Rykhlevskii that the uranium input is the same as the protactinium output should be valid. However, there are expected to be differences for the first several months of operation, as the half life of protactinium-233 is on the order of one month, and the initial decrease in fissile material will impact reactor performance. This means that the uranium feed will take some time before it is providing a steady source of fissile material to the core. This also means the reactor physics will be different, which is an important when considering the safety of the reactor.

3.2 Batchwise Reprocessing in the MSBR

As previously mentioned in Section 2.4.2, SaltProc has two different versions which use two different batchwise reprocessing approaches. These approaches are the bulk batchwise approach, used in version 0.1 of SaltProc, and

the steady batchwise approach, used in version 0.3 of SaltProc. In the next two subsections, I will describe each of these approaches and how they differ.

3.2.1 Bulk Reprocessing Approach

The bulk approach operates by removing 100% of the target once a cycle time, or time it takes for a given element to be completely removed, is met or exceeded. For example, consider a simulation with a three day depletion step size. An element with a 30 day cycle time would have no reprocessing during the first 27 days, then full 100% removal of the target at the 30 day mark. This works well for reactor processes which are performed batchwise, such as adding uranium to the reactor which can be performed as a batch process. However, this approach is not ideal for approximating online reprocessing, as it does not capture physics that occur due to frequent reprocessing.

For the refueling of the reactor, the thorium feed rate was set to maintain constant mass in the reactor. This means that after a bulk removal of a large amount of fission products, there will be an equivalent amount of thorium added. Using a steady state approximation, the uranium feed rate is set to be equivalent to the protactinium removal rate at each time step. Specifically, the protactinium is removed first, and then the same mass of fresh uranium is added. This carries an implicit assumption that there is already a backlog of decayed protactinium which has transmuted into uranium which is ready to be pumped back into the core. Both the thorium feed and protactinium to uranium refueling processes occur every 3 days. These refueling rates can be seen in Figures 3.3 and 3.4, which are recreated from the data from Rykhlevskii [37].

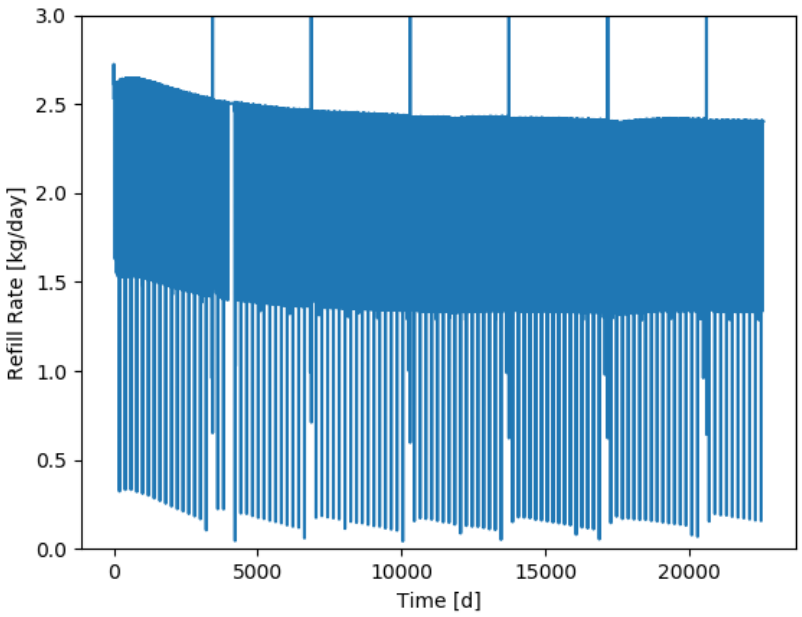


Figure 3.3: Thorium feed rate in the MSBR as a function of time while using bulk batchwise reprocessing [37].

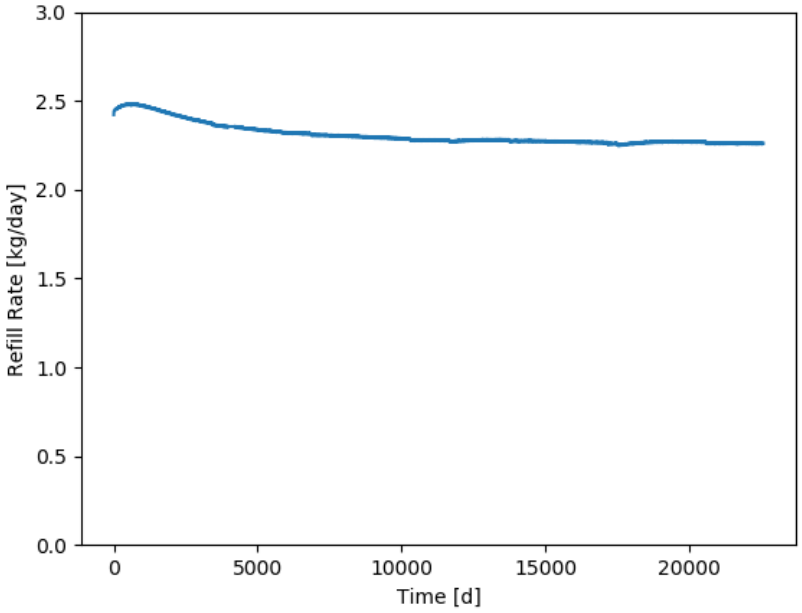


Figure 3.4: Uranium feed rate in the MSBR as a function of time while using bulk batchwise reprocessing [37].

The large jumps in the thorium feed rate are due to the bulk removal of rubidium, strontium, caesium, and barium; this causes jumps in the thorium feed rate because the thorium feed rate in this version of SaltProc is based

on mass balancing. Removing a large amount of fission products results in that mass being replaced by thorium. This also leads to spikes in the effective multiplication factor, as the removal of the fission products inserts more reactivity than the addition of thorium removes [37]. The uranium feed rate is smooth because it is based purely on the outflow of protactinium and does not fluctuate like the thorium, which is the only external feed in the MSBR system. The average thorium-232 feed rate is 2.38 kg/day over 20,000 days, while the average uranium-233 feed rate is 2.31 kg/day over the same 20,000 days. These average values are used as the baseline for the feed rates in the rest of this work.

3.2.2 Steady Reprocessing Approach

Steady batchwise removes a fraction of the target at each depletion time step rather than 100% removal at certain time steps. For example, a 30 day cycle time would result in 10% removal every 3 days. This more accurately models online reprocessing, which is primarily what the MSBR employs in its reprocessing scheme. However, this steady version of SaltProc also makes some minor changes to the reprocessing scheme by adding in efficiency terms. These changes act as a multiplier on the reprocessing constant, where 100% efficiency makes no change and 0% makes the reprocessing constant become 0, as shown in Equation (2.2). The changes used in this work to replicate the work by Rykhlevskii are as follows: xenon uses 91.2% removal over three days instead of 100%, krypton uses 91.5% removal over three days instead of 100%, protactinium uses 9.5% removal over three days instead of 100%, and the discard of rubidium, strontium, caesium, and barium have a removal of 0.9% over three days instead of 0.09% [38, 2]. These rates come from the cycle times in Table 2.5 and Equation (2.2).

The reactor refueling in steady batchwise reprocessing is performed slightly differently from the bulk batchwise method as well. The former assumes a constant ratio for the refueling quantities of uranium and thorium. The net refueling for each 3 day depletion step is set to maintain the mass of the system. Thus, the average values of 2.38 kg/day and 2.31 kg/day for thorium-232 and uranium-233, respectively, are used to determine the ratio of uranium to thorium which is implemented.

Figures 3.5 and 3.6 show the refueling rate of thorium-232 and uranium-233, respectively, using a depletion step size of 3 days over a net depletion simulation time of 90 days. The average feed rates of the thorium-232 and uranium-233 are 1.24 and 1.21 kg/day. The reasons for the difference are because the smaller efficiencies cause less mass loss, a shorter time frame is covered, and the protactinium removal is much less, resulting in a much smaller uranium-233 feed.

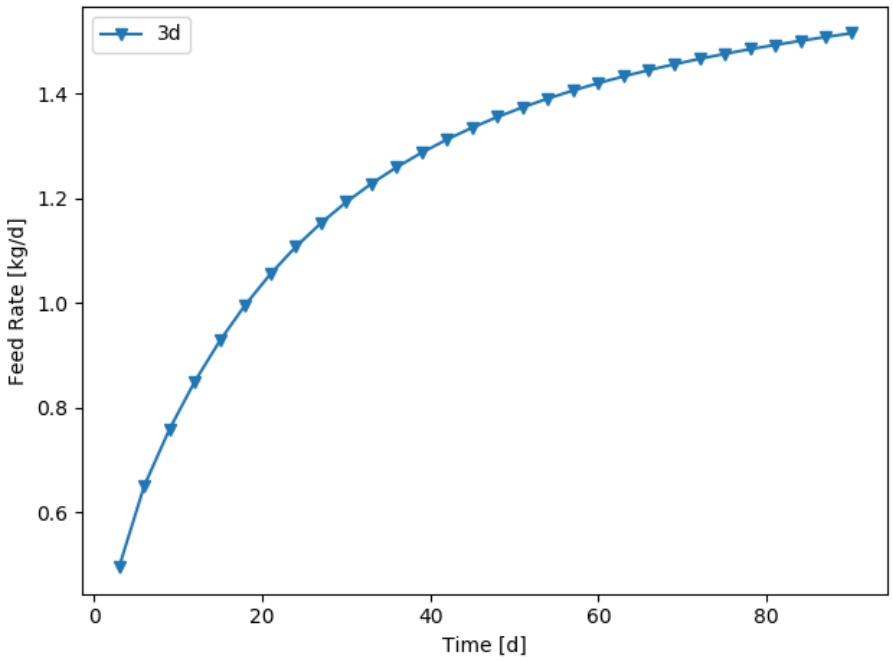


Figure 3.5: Thorium feed rate with a depletion time step of 3 days in the MSBR as a function of time while using steady batchwise reprocessing.

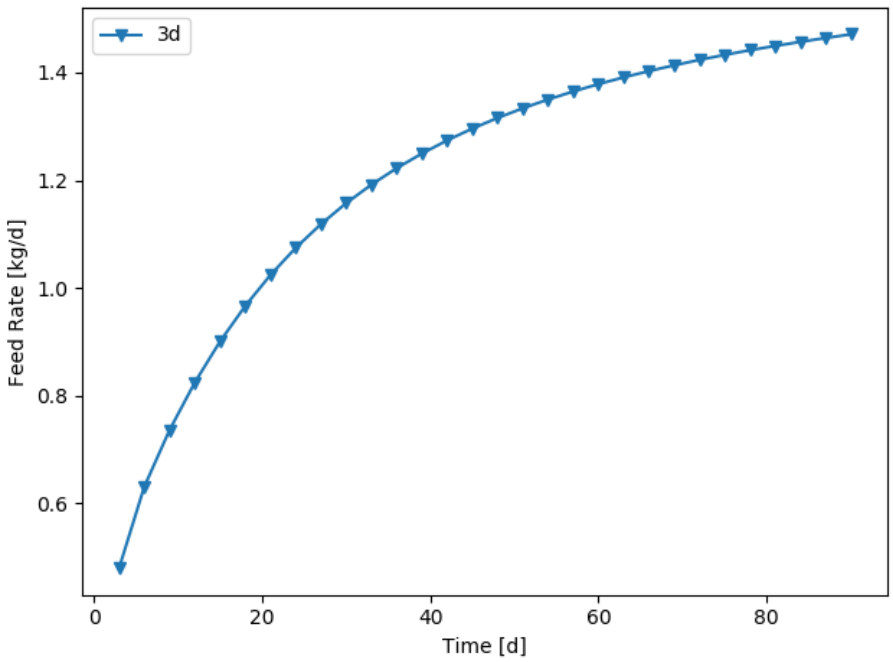


Figure 3.6: Uranium feed rate with a depletion time step of 3 days in the MSBR as a function of time while using steady batchwise reprocessing.

3.2.3 Batch Approaches Summary

To compare the different batchwise approaches, there are several key parameters to keep in mind. These include the approach used, which can be either bulk or steady for this work; the time step Δt, which is the simulated depletion time step used; and the cycle time T_{cyc}, which is inversely proportional to the rate at which a target is removed. For both steady and bulk, the step removal can then be calculated, which is the fraction of the target removed at each depletion time step. Additionally, for the steady, the fractional removal rate can be calculated. This is just the step removal in per second units. Comparisons of these values for various parameters can be seen in Table 3.1.

Table 3.1: Batchwise Reprocessing Methods

Approach	Δt [s]	T_{cyc} [s]	Fractional Removal Rate [s^{-1}]	Step Removal
Bulk	1	20	-	0/1*
Bulk	10	20	-	0/1*
Bulk	40	20	-	1
Steady	1	20	0.05	0.05
Steady	10	20	0.05	0.5
Steady	40	20	0.025	1

* Bulk removal is 0 until the depletion step is equal to the cycle time, at which point it removes 100%.

Table 3.1 reflects the information shown in Figure 3.7, which shows the fractional removal rate at each depletion time step. The table shows how the bulk method will only remove 100% at the cycle time, and otherwise removes nothing. Additionally, the bulk method will remove 100% as soon as possible if the depletion step size, Δt, is larger than the cycle time, T_{cyc}.

The table also describes how the steady method fractional removal is constant up until the depletion step size becomes larger than the cycle time, at which point the removal at each step is 100%, causing the steady method to then be equivalent to the bulk method. However, for shorter times, the steady method allows for a consistent fractional removal rate by adjusting the removal at each step.

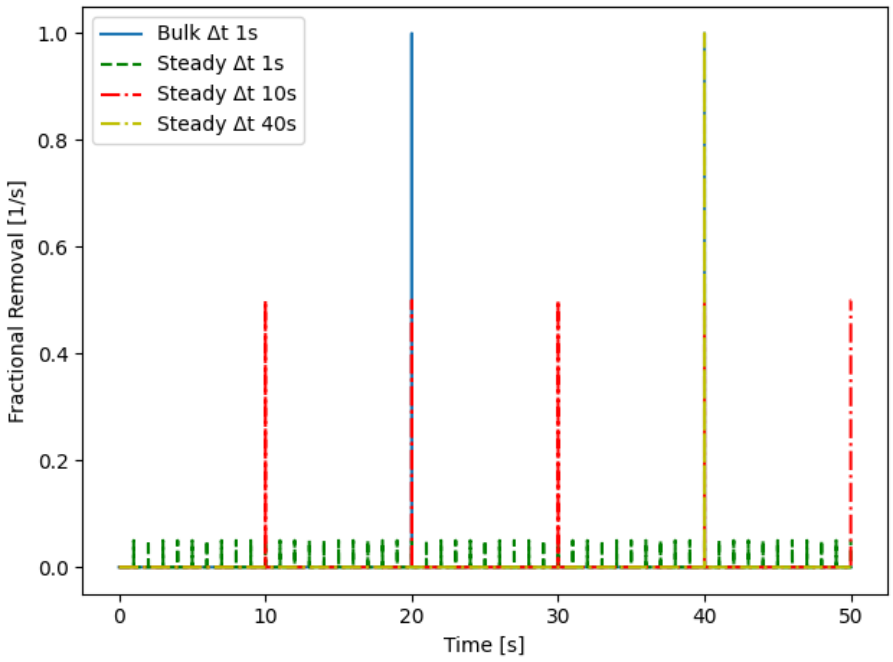


Figure 3.7: Plot showing how bulk and steady batchwise reprocessing removal works as a function of time for an example with a cycle time of 20 seconds.

Figure 3.7 shows an example fractional removal scheme for a fission product with a cycle time of 20 seconds. The bulk method simply extracts 100% of the product every 20 seconds, no matter how small the depletion step size is. The steady method removes some fraction every depletion step, which allows a semi-continuous process as the depletion step size becomes smaller and smaller. This can be seen in how a 10 second depletion step results in 50% removal every 10 seconds, whereas a 1 second step allows for 5% removal per depletion step.

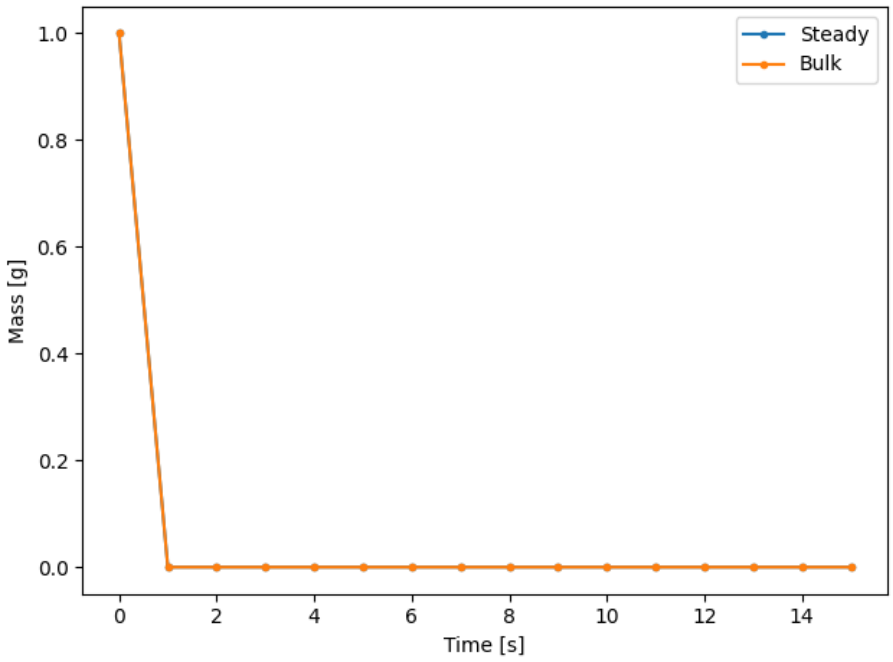


Figure 3.8: Plot showing how bulk and steady batchwise reprocessing are identical in the cases where the cycle time is shorter than or equal to the depletion step size.

A useful piece of information to note is that if the cycle time is shorter than or equal to the depletion step size, then the steady and bulk batchwise methods will be identical to each other. This is demonstrated in Figure 3.8, which shows a simple model case with 1 second depletion steps and 1 second cycle times for an imaginary isotope. For the MSBR simulation, this means that there will be no mass differences due to reprocessing in the noble metals, gasses, and protactinium while using a three day depletion time step, which is longer than or equal to the cycle time for each of those groups. However, the longer cycle time groups such as the rare earths and volatile fluorides have cycle times which are 10-20 times longer than the depletion step size. These elements will have differences between both methods. As a result, the full reactor system will have elements with varying levels of error.

Figure 3.9 shows a simulation of this for a situation where an isotope is generated at a rate of 1 gram per second, and is simulated with some combination of cycle time and depletion step size where the depletion step size is smaller than the cycle time.

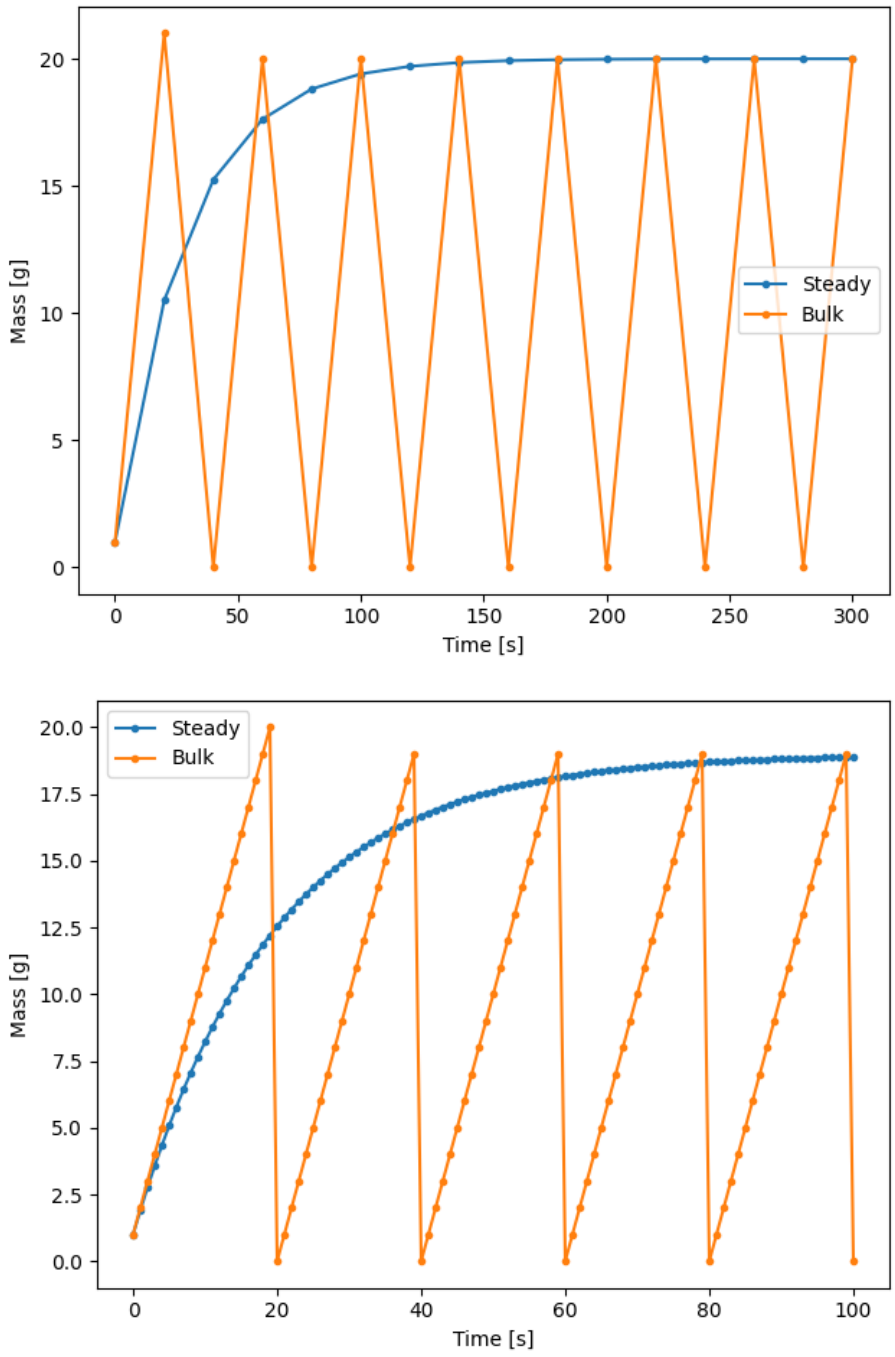


Figure 3.9: Plot showing how bulk and steady batchwise reprocessing are different while the cycle time is longer than the depletion step size. In this system, the element has a 40 second cycle time and 20 second depletion step (top) and a 20 second cycle time with a 1 second depletion step (bottom).

One interesting aspect to note from this figure is that the steady method approaches a steady state value which is at the peak of the bulk method. This result can be confirmed by checking that the solutions for both the steady and bulk methods are valid. To check the solution, the 40 second cycle time and 20 second depletion step will be

used.

For the bulk method, after 100% removal after the first cycle time, or 40 seconds, the mass then oscillates between 20 and 0 grams. This makes sense since 20 grams are generated over 20 seconds while every 40 seconds there is 100% removal.

For the steady method, the initial mass of 1 gram gains 20 grams over 20 seconds, yielding 21 grams. However, the steady removal then requires removal of 50%, previously discussed and shown in Figure 3.7. This 50% removal then drops the mass to 10.5 grams. 20 more grams are added to this value and 50% is removed again, continuing iteratively. This can then be re-written as an infinite sum in order to determine the steady state solution. The first three iterations are shown in Equations (3.1) and (3.2), where the value of 20 is the mass added during each depletion step and the value of 0.5 is the fractional removal performed each depletion step.

$$m_{ss} = \left(\left((N_0 + 20)\left(\frac{1}{2}\right) + 20\right)\left(\frac{1}{2}\right) + 20\right)\left(\frac{1}{2}\right) + \ldots \tag{3.1}$$

$$m_{ss} = \frac{1}{2}^3 N_0 + \frac{1}{2}^3 (20) + \frac{1}{2}^2 (20) + \frac{1}{2}(20) + \ldots \tag{3.2}$$

Continuing this form infinitely yields Equation (3.3), where the infinite sum yields 1, resulting in a net value of 20, the result of which is unaffected by the initial mass of the isotope and instead depends on the generation and reprocessing rates.

$$m_{ss} = 20 \sum_{n=1}^{\infty} \frac{1}{2}^n \tag{3.3}$$

In order to generate a generic form for this equation, a short derivation is required, which is provided in Appendix A. From this derivation, the mass at the $n+1^{th}$ step is given by Equation (3.4).

$$m_{n+1} = \left[\left(m_n + \frac{C}{\lambda - \lambda_r}\right) e^{-(\lambda - \lambda_r)\Delta t} + \frac{C}{\lambda - \lambda_r}\right](1 - \Delta t \lambda_r), \tag{3.4}$$

In this equation, C represents the gain terms in the Bateman equation, Δt represents the depletion step size, λ represents the loss terms in the Bateman equation, and λ_r represents the reprocessing constant. This equation assumes a constant C, λ, and λ_r, which is a reasonable assumption at steady state when the changes over time are negligible. This equation is used to generated the steady state equation as shown in Equations (3.5) through (3.9).

The $n+1^{th}$ step can be written in the form of a sum over n terms, which allows for the steady state mass to be written as an infinite sum

$$m_{ss} = \frac{C}{\lambda - \lambda_r}\left(e^{(\lambda-\lambda_r)\Delta t} - 1\right)\sum_{n=1}^{\infty}\left((1-\lambda_r\Delta t)\, e^{-(\lambda-\lambda_r)\Delta t}\right)^n. \tag{3.5}$$

This infinite sum is of the more general form

$$\sum_{n=1}^{\infty}(x)^n = \frac{x}{1-x} \ni |x| < 1, \tag{3.6}$$

which has a known converged solution and associated constraint to use that converged solution. I can then rewrite the infinite sum as

$$m_{ss} = \frac{C(e^{(\lambda-\lambda_r)\Delta t} - 1)}{\lambda - \lambda_r}\left(\frac{\eta}{1-\eta}\right) \ni \left|\eta\right| < 1, \tag{3.7}$$

where

$$\eta = (1-\lambda_r\Delta t)e^{-(\lambda-\lambda_r)\Delta t}. \tag{3.8}$$

In these equations, C still represents the gain terms in the Bateman equation, Δt represents the depletion step size, λ represents the loss terms in the Bateman equation including the loss from reprocessing, and λ_r represents the reprocessing constant.

The constraint listed in Equation (3.7) that $\left|\eta\right| < 1$ shows when the equation no longer holds. However, this constraint only shows when the infinite series becomes divergent. The more physical constraint, $0 < \lambda_r\Delta t < 1$ is given in Equation (3.9), which shows the completed form of the steady state mass. If the depletion step size, Δt, is equal to zero, then the simulation will never end, as no progress is made. Thus, the concept of steady state does not apply. For any depletion step sizes which are greater than the cycle time, the steady state mass will always be zero, as negative masses are non-physical.

$$m_{ss} = \frac{C(e^{(\lambda-\lambda_r)\Delta t} - 1)}{\lambda - \lambda_r}\left(\frac{\eta}{1-\eta}\right) \ni 0 < \lambda_r\Delta t < 1 \tag{3.9}$$

Overall, this shows that the steady and bulk batchwise methods have some similarities, but for elements with longer cycle times, the bulk method will experience oscillations whereas the steady method will level off smoothly. Additionally, the average value of the steady method will be higher than the oscillating bulk method. Finally, if I take the limit of Equation (3.9) as the time step approaches zero, Equation (3.10) is generated.

$$\lim_{\Delta t\to 0} m_{ss} = \frac{C}{\lambda} \tag{3.10}$$

This is the same steady state mass as generated from a continuous reprocessing method with the same gain, loss, and reprocessing terms in the Bateman equation.

3.2.3.1 Comparison with Continuous Reprocessing

Error Estimation The difference between the bulk and steady batchwise reprocessing methods has been demonstrated, though how these methods compare to the continuous reprocessing methods has not been shown. For a simple example comparison, the steady batchwise method will be compared against the direct linear continuous method, which gives the relationship shown in Equation (3.11). The derivation for this equation can be seen in Section 3.3.2.5.

$$\lambda_r = \frac{1}{T_{cyc}} \tag{3.11}$$

This comparison will assume there is some isotope which is generated at a constant rate, C, and is only removed due to reprocessing. This is a simple case to compare, as there is no parasitic absorption, decay chains, or other more complex physics to treat. Additionally, the impact of the reprocessing differences will be more noticable, as there won't be competing removal effects.

The relevant continuous method equations are given in Equations (3.12) and (3.13), where the variables are the same as those discussed in Equation (3.9). Equation (3.12) shows the generic differential equation, while Equation (3.13) shows the mass as a function of time.

$$\frac{dm}{dt} = C - \lambda m \tag{3.12}$$

$$m(t) = \left(m_0 - \frac{C}{\lambda}\right) e^{-\lambda t} + \frac{C}{\lambda} \tag{3.13}$$

Equation (3.14) shows the equation for the steady batchwise method mass at the $n+1^{th}$ step. This equation is dependent upon the time step, so several time steps will need to be included in order to properly compare the steady batchwise and direct linear continuous methods.

$$m_{n+1} = (C\Delta t + m_n)(1 - \lambda_r \Delta t) \tag{3.14}$$

The previous example is continued here, using a constant rate of 1, an initial concentration of 1, a cycle time of 40 seconds, and a varying depletion step size. The results of this comparison can be seen in Figure 3.10, which is generated by plotting the masses over time for the given equations. In this figure, the continuous initially has a

sharp increase in mass which is more closely matched by the steady batchwise approaches which use larger time steps. However, as the simulation time continues, the smaller time step steady batchwise result has a closer value to the continuous than the larger time step approaches.

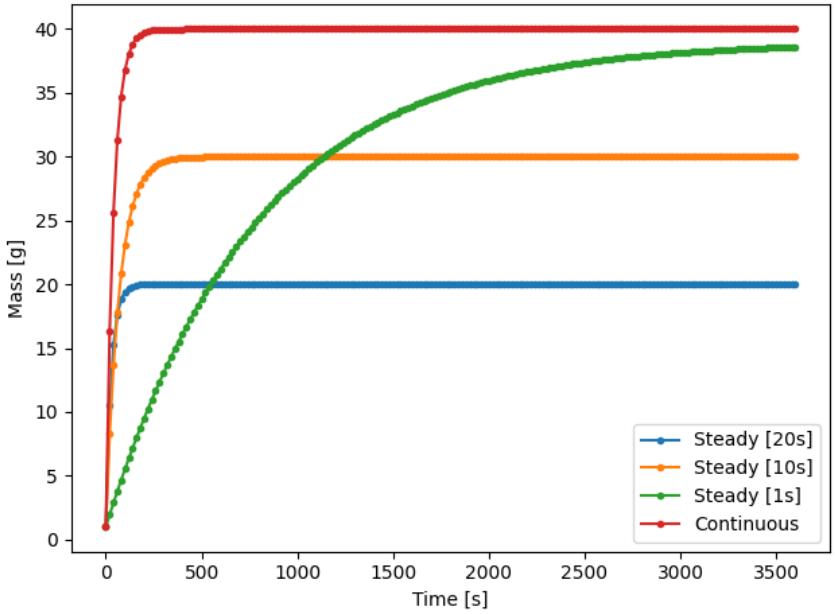


Figure 3.10: Plot comparing the values of continuous and batchwise reprocessing methods with a 40 second cycle time for an arbitrary nuclide.

Overall, this figure shows that there are going to be differences between the batchwise and continuous methods, even with a time step which approaches zero. This is due to the differences in the slopes, which can be seen when comparing the continuous and steady for a small time step of one second. However, it can also be seen that the differences at large time values is smaller for shorter time steps and larger for longer time steps when comparing the steady batchwise and continuous methods. From this observation, I can provide a derivation for the expected difference between the methods at steady state, with some simplifying assumptions.

$$m_{ss} = \frac{C}{\lambda} \tag{3.15}$$

Equation (3.15) shows the steady state mass which will appear when using continuous reprocessing when there is a constant growth term. This can be set equal to the steady batch steady state term in order to solve for the depletion time step needed to match the result from continuous reprocessing at steady state. The steady state mass was previously calculated for a time step approaching zero in Equation (3.10). In that equation, I showed that the continuous steady state mass is equal to the steady batchwise steady state mass when the time step approaches zero.

Correspondingly, this means that the steady state error associated with implementing the steady batchwise method to approximate a continuous reprocessing scheme is proportional to the magnitude of the depletion step size used. Specifically, as shown in Equations (3.16) and (3.17), the error at steady state due to reprocessing directly scales based on the ratio of the depletion time step and the cycle time. In this work, I treat the error as the normalized difference of the batchwise mass from the continuous mass.

$$|E| = \frac{|m_{ss}^{batch} - m_{ss}^{continuous}|}{m_{ss}^{continuous}} \tag{3.16}$$

$$|E| = \frac{\lambda}{C}\left|\frac{C}{\lambda} - \frac{\lambda(e^{(\lambda-\lambda_r)\Delta t}-1)}{\lambda-\lambda_r}\left(\frac{(1-\lambda_r\Delta t)e^{-(\lambda-\lambda_r)\Delta t}}{1-(1-\lambda_r\Delta t)e^{-(\lambda-\lambda_r)\Delta t}}\right)\right| \tag{3.17}$$

An exmaple of this relative error can be seen in Figure 3.10, as the depletion time step to cycle time ratio is exactly 2 for the steady batchwise 20 second step size. Correspondingly, the steady state value is twice is low as the continuous steady state value. Although these equations show only the simplest case of a constant growth, they demonstrate how the important using a small depletion step size is for batchwise methods to generate an accurate result when approximating a continuous scheme.

This method also fits well for complicated interactions which are present in the molten salt reactor model. As previosuly discussed, for this equation to fit, there are several assumptions which must be made, such as a constant gain and loss terms. For this simplified model to fit the MSBR, I assume there is a constant amount of fissile mass in the system, in the case of the MSBR this is primarily uranium-233, m_{u233}; I assume the fission yield, γ, does not change significantly due to spectral changes over time; and I assume a constant neutron flux, ϕ. The equations can then be expanded using the variables shown in Equation (3.18) and (3.19). The terms in these equations come from the Bateman equation.

$$C_i = \phi\sigma_{u233}\gamma N_{u233} + \sum_j N_j\left(\lambda_{j\to i} + \sigma_{j\to i}\phi\right) \tag{3.18}$$

$$\lambda_i = N_i\left(\lambda_{i,d} + \lambda_{i,r} + \sigma_i\phi\right) \tag{3.19}$$

In these equations, the reprocessing constant, λ_r, is rewritten from its previous form of λ because the decay constant, λ_d, is now present. Due to the simplifying assumptions made, which are valid over a short net depletion time or at steady state, these equations is of the same form as the simple equation presented in the example above. Thus, by replacing the C and λ terms from the simple model for error shown in Equation (3.17), the model provides an estimate of error for the many isotopes in the reactor simulation. Realistically, the assumptions made here will

not all hold, but this is still a useful tool to gain a low-fidelity approximation for the level of error which can be expected by approximating continuous online reprocessing using a steady batchwise reprocessing method.

Computational Cost One of the weaknesses of batchwise methods is the necessarily higher computational cost compared to continuous methods. The first reason for this is because batchwise methods which approximate a continuous process have reprocessing based error which scales with depletion step size, as shown in Equation (3.17). This means that smaller depletion step sizes are required for the same error as a continuous method.

The second reason is because of the "double-running" of simulations which is required when using a batchwise method in Serpent2. When running a batchwise depletion program, the first time step is run and the new material compositions are generated, providing 2 complete simulation results at times 0 and Δt. However, these generated materials do not have reprocessing factored in, so the external reprocessing script, such as SaltProc, then performs the reprocessing on the materials. The updated materials are fed back in, and the simulation is run again. This results in two new simulation results at Δt and $2\Delta t$.

Because of the necessity of regenerating data based on the updated material properties after the external depletion is performed, each time step requires two simulations to be performed, essentially doubling the computational cost. With a continuous method, the materials are updated during the depletion step, which means the generated data already uses the data of reprocessing materials. This eliminates any need for double running, allowing continuous methods to be run for much less computational cost.

However, there are still motivations for using batchwise reprocessing methods. If a reactor includes batchwise reprocessing in the reprocessing scheme, then using a batchwise reprocessing method is necessary to properly capture that physical phenomena. Additionally, with clever coding, the overhead from using batchwise reprocessing methods can be reduced, which could eliminate the need to "double-run" time steps in the simulation. Batchwise might be particularly attractive in the case where fine depletion time steps are used to more fully capture spectral changes, in which case using batchwise reprocessing methods will not adversely affect run times in a significant way if double-runs are avoided. Using batchwise reprocessing methods also allows for simpler customization of mass balancing, which can be treated with an overflow volume, adjustments to refuel rates, or any other modification to the compositions.

3.3 Continuous Reprocessing in the MSBR

The continuous reprocessing functionality of Serpent2 was originally sparsely documented on user forums, but as of October 2020, more comprehensive documentation has been made available [40]. There are three different options which can be used for the Serpent2 built-in continuous reprocessing. The different options are defined in

Serpent2 as 0, 1, and 2; which are henceforth referenced as "constant", "decay", and "step" reprocessing. These names are selected because they closely describe the behaviour of each of the methods.

3.3.1 Constant Reprocessing Approach

The Constant reprocessing approach applies only positive terms to the Bateman equation, essentially treating feeds without any associated removal. This is the only method of the three options which does not conserve mass. This option instead generates mass the same way as the Decay reprocessing method for the initial step, then continues using that mass rate from that point onwards. This method is useful for handling a constant fresh fuel feed rate, or any other situation where a constant addition of mass is desired. Because this method does not remove mass, it is not useful for any sort of constant removal. For example, if a reactor needed only refueling to be considered, then this approach would work well for modeling that reactor.

3.3.2 Decay Reprocessing Approach

The Decay reprocessing method is the most useful of the three options, and that is due to its flexibility across many use cases. This method adds a decay term to whatever material it is attached to, and that decay term becomes a feed term to whatever material is set to receive the flow. For fission product removal, this is likely some material which is not modeled in the geometry of the problem but only exists to receive the fission product waste. However, this term can instead be leveraged to turn it into a feed rate.

This can be performed by attaching the decay term to a material which contains a volume of the desired feed, and then having that material feed into the core. The feed then "decays" from the feed tank into the core, essentially operating as a feed. Using this same method but altering the reprocessing constant or volume of the feed tank allows for an essentially constant feed rate, where the removed mass is negligible compared to the net mass.

Additionally, for this method, there are multiple approaches which can be used to calculate the reprocessing constants that should be implemented. These approaches are referred to in this work as "cycle time decay", "cycle rate", and "direct linear" accordingly, and can be used in any code which uses the decay approach to continuous reprocessing. The cycle time decay approach treats the cycle time as if it is twice a half-life, and the cycle time is an exponential process. The cycle rate approach treats the cycle time as the time where 100% of removal occurs, and then linearly extrapolates by assuming an equal percentage removal occurs up to that point at each unit time step. The direct linear approach is the same as the cycle rate approach, but instead of using a unit time step, uses the limit as the time step approaches zero, which is more consistent with the continuous nature of the removal.

Additionally, there are "SaltProc" versions of each approach implemented as well which are designed to have the continuous reprocessing method approximate the steady batchwise results. This is useful to establish, as this

method will allow for a comparison of the difference in reprocessing method even when the reprocessing constants are adjusted based on the time step used by the steady batchwise reprocessing scheme of interest.

The reason these approaches are implemented is because molten salt reactor reprocessing schemes provide reprocessing data in terms of cycle times. The cycle time is the amount of time it takes for a given element to be removed from the system. In the next few equations, I illustrate why we cannot solve these directly with the Bateman equations. This is why I make use of different approaches which approximate the removal of an element based on this definition of the cycle time.

Firstly, Equation (3.20) shows the Bateman equation for a simple nuclide with only reprocessing removal.

$$\frac{dN}{dt} = -\lambda_r N \tag{3.20}$$

After some time step equivalent to the cycle time, this equation turns into Equation (3.21).

$$N(\Delta t) = 0 = N_0 e^{-\lambda_r \Delta t} \tag{3.21}$$

Simplifying this equation, we find in Equation (3.22) that

$$-ln(0) = \lambda_r \Delta t. \tag{3.22}$$

Thus, these equations show how the solution results in an undefined value when only continuous removal is considered. The other situation to consider is when there is some constant gain and loss only from reprocessing, as shown in Equation (3.23).

$$\frac{dN}{dt} = C - \lambda_r N \tag{3.23}$$

The same processes is followed as before, setting the atom density to 0 after some amount of time representing the cycle time, as shown in Equation (3.24).

$$N(\Delta t) = 0 = N_0 e^{-\lambda_r \Delta t} + \frac{C}{\lambda_r}\left(1 - e^{-\lambda_r \Delta t}\right) \tag{3.24}$$

This equation can be simplified further to the form shown in Equation (3.25).

$$0 = \lambda_r N_0 e^{-\lambda_r \Delta t} + C - C e^{-\lambda_r \Delta t} \tag{3.25}$$

These equations show that the only real solution to a steady accumulation and continuous removal is with a

reprocessing constant of 0. However, there are approaches which can be used to approximate this definition of cycle time while still moving closer to the physical chemical processes. In reality, the continuous online reprocessing follows an exponential decay, becoming less efficient as the elemental concentration decreases. Thus, although the definition of the cycle time is being approximated using these approaches, the physical phenomena is being modeled accurately.

3.3.2.1 Cycle Time Decay

The cycle time decay approach is a method which makes use of the mathematical form of the decay reprocessing method. A similar method of treating removal periods for reprocessing can be seen in Brovchenko et al [9]. Because it adds a "decay-like" term to the Bateman equation, this approach takes the cycle time for the target, cuts it in half, and then treats that value as the reprocessing half-life for that target, as shown in Equations (3.26) and (3.27).

Equation (3.26) shows how the reprocessing half-life term is created from the cycle time.

$$\Delta t_{1/2} = \frac{T_{cyc}}{2} \tag{3.26}$$

Equation (3.27) takes this reprocessing half life value and creates a reprocessing constant in the same manner as a decay constant.

$$\lambda_r = \frac{ln(2)}{\Delta t_{1/2}} \tag{3.27}$$

An example of implementing this approach for a 30 second cycle time would then have a 15 second reprocessing half-life. This is then converted to a reprocessing constant in the same way that a decay half-life is converted to a decay constant, which is done by plugging these values into Equation (3.27), resulting in a value of 0.0462 s^{-1}.

3.3.2.2 SaltProc Cycle Time Decay

This approach is the same as cycle time decay, but alters for any target which has a cycle time less than the batchwise reprocessing step incorporated by SaltProc. For example, a 3 day cycle time for some target would be treated the same as the standard cycle time decay. However, a cycle time shorter than the 3 day batchwise reprocessing step used by SaltProc for the MSBR has its half-life extended to the SaltProc minimum value of 1.5 days. For example, a 30 second cycle time would instead be treated as a 3 day cycle time, which would result in a half-life of 1.5 days. Plugging in, this would result in a reprocessing constant of 0.462 days^{-1}, or 5.348E-6 s^{-1}. This is the reprocessing constant for any cycle time which is less than or equal to three days. Additionally, the decay constant is multiplied by the fractional efficiency of the removal used by SaltProc. In most cases the value is 1, but it is roughly 0.91 for

xenon and krypton in order to align with Rykhlevskii's results [38].

3.3.2.3 Cycle Rate

The cycle rate approach uses a linear approximation such that the inverse of the cycle time is the rate at which material is removed. This is represented by, for example, 10% removal per second would neglect the efficiency decrease over time and give 100% removal after 10 seconds. This is calculated by investigating a unit time progression, i.e. 1 second or 1 day. Over this time period, the removal of atoms should be the fractional rate value, which means the final atom count at a time of 1 should be 1 - f, where f is the fractional removal rate. This can be seen in Equations (3.28) and (3.33).

First, I start with Equation (3.28) to define the fraction removed per unit time based on the time units used to define the cycle time.

$$f = \frac{1}{T_{cyc}} \tag{3.28}$$

Next, I define a nuclide Bateman equation with only reprocessing removal considered in Equation (3.29).

$$\frac{dN}{dt} = -\lambda_r N \tag{3.29}$$

This equation is then solved for the concentration as a function of time in Equation (3.30).

$$N(t) = N_0 e^{-\lambda_r t} \tag{3.30}$$

After solving, a unit time value is plugged in. After a unit time, the amount of the nuclide remaining should be $(1-f)$, as shown in Equation (3.31).

$$N(t=1) = (1-f)N_0 \tag{3.31}$$

After plugging in the concentration equation from Equation (3.30) into Equation (3.31), Equation (3.32) is generated.

$$-ln(1-f) = \lambda_r \tag{3.32}$$

Finally, the reprocessing constant is isolated, resulting in Equation (3.33).

$$\lambda_r = ln\left(\frac{1}{1-f}\right) \tag{3.33}$$

To better clarify the approach, here is an example. A cycle time of 30 seconds would be modeled by taking the inverse, which gives 0.033 s^{-1}. This value is then converted to a reprocessing constant by plugging it into the solved differential equation form shown in Equation (3.33), yielding a reprocessing constant of 0.0339 s^{-1}.

3.3.2.4 SaltProc Cycle Rate

The SaltProc cycle rate approach is the same as the cycle rate approach, but takes into account the limiting nature of the 3 day batchwise reprocessing step used by SaltProc. For example, a 6 day cycle time target would be modeled the same using the SaltProc cycle rate approach as the standard cycle rate approach. However, anything shorter than 3 days would be modeled differently, since that is the batchwise reprocessing step inocorporated by SaltProc for modeling the MSBR. For example, a 30 second cycle time would be analyzed instead as a 3 day cycle time, since SaltProc can only remove 100% of material after a minimum of 3 days. This means the inverse would be 0.333 $days^{-1}$, or 3.858E-4 s^{-1}. Converting to a reprocessing constant gives a value of 3.858E-6 s^{-1}. This is the reprocessing constant for any cycle time which is less than or equal to three days. Additionally, the decay constant is multiplied by the fractional efficiency of the removal used by SaltProc. In most cases the value is 1, but it is roughly 0.91 for xenon and krypton in order to align with Rykhlevskii's results [38].

3.3.2.5 Direct Linear Approach

Another approach is to directly apply the cycle rate removal rate, which is the inverse of the cycle time, as the reprocessing constant [19]. This is referred to as the direct linear approach. This approach is similar to the cycle rate approach, though the derivation for it is slightly different. This can be seen in Equations (3.34) through (3.44).

To start, the derivation begins with the same fractional removal rate as the cycle rate approach in Equation (3.34).

$$f = \frac{1}{T_{cyc}} \tag{3.34}$$

Then, the Bateman equation for a nuclide with only losses due to reprocessing is solved, with the concentration as a function of time given in Equation (3.35).

$$N(t) = N_0 e^{-\lambda_r t} \tag{3.35}$$

This equation can be rewritten as multiplying the previous concentration by a value dependent on the time step,

as shown in Equation (3.36).

$$N_{cur} = N_{prev} e^{-\lambda_r \Delta t} \tag{3.36}$$

In addition, after each step, the fractional removal is applied over that time step, allowing for the current concentration to be calculated as shown in Equation (3.37).

$$N_{cur} = (1 - \Delta t f) N_{prev} \tag{3.37}$$

Equations (3.36) and (3.38) are then set equal to each other, yielding Equation (3.38).

$$(1 - \Delta t f) N_{prev} = N_{prev} e^{-\lambda_r \Delta t} \tag{3.38}$$

This equation is simplified to the form shown in Equation (3.39).

$$-ln\big((1 - \Delta t f)\big) = \lambda_r \Delta t \tag{3.39}$$

Solving for λ_r yields Equation (3.40).

$$\lambda_r = \frac{-ln\big((1 - \Delta t f)\big)}{\Delta t} \tag{3.40}$$

Instead of seeking the reprocessing constant for a unit time step as found using the cycle rate approach, I instead seek the reprocessing constant as the time step approaches zero as shown in Equation (3.41).

$$\lambda_r = lim_{\Delta t \to 0} \frac{-ln\big((1 - \Delta t f)\big)}{\Delta t} \tag{3.41}$$

Because this results in a fraction of zero over zero, as shown in Equation (3.42), I use L'Hôpital's rule to change to the form shown in Equation (3.43).

$$\lambda_r = \frac{0}{0} \tag{3.42}$$

$$\lambda_r = lim_{\Delta t \to 0} \frac{f}{1 - f \Delta t} \tag{3.43}$$

This form can then be directly solved, yielding the reprocessing constant equation shown in Equation (3.44).

$$\lambda_r = f = \frac{1}{T_{cyc}} \tag{3.44}$$

These equations follow a similar path to the cycle rate approach, but instead of using the linear approximation value after a unit time step, this approach generates the reprocessing constant while implementing the linear approximation as the time step goes to zero.

The differences between the cycle rate and direct linear reprocessing constants can be seen in Table 3.2, where for longer cycle times, the difference is negligible, but for shorter cycle times, the difference becomes larger. However, since extremely short cycle times are not realistically practical, the two approaches are approximately equivalent. Overall, the direct linear approach is more numerically stable, however, since it does not have asymptotic behavior until reaching a cycle time of zero seconds, whereas the cycle rate method has asymptotic behaviour for a one second cycle time. This can be seen from the form of the equations for the reprocessing constants and in Figure 3.11, which displays how the reprocessing constants vary for various cycle times. The asymptotic behaviour for a zero second cycle time is not an issue because as the cycle time goes to zero, that would mean that the target is removed over zero seconds. This means an infinite removal rate would be physical. Additionally, there are no negative cycle times, meaning the value is only approached from the positive side and behaves as physically expected. Realistically, reprocessing scheme cycle times are longer than the order of seconds, which means for most practical use cases these two approaches are interchangable.

Table 3.2: Subset of Decay Reprocessing Approaches

Cycle Time	Removal Rate $[s^{-1}]$	CR λ_r $[s^{-1}]$	DL λ_r $[s^{-1}]$	$\Delta\lambda_r$
3 d	3.86E-6	3.86E-6	3.86E-6	7.44E-12
20 s	0.05	5.13E-2	5.00E-2	1.29E-3
5 s	0.2	2.23E-1	2.00E-1	2.31E-2
2 s	0.5	6.93E-1	5.00E-1	1.93E-1
1 s	1	-	1	-

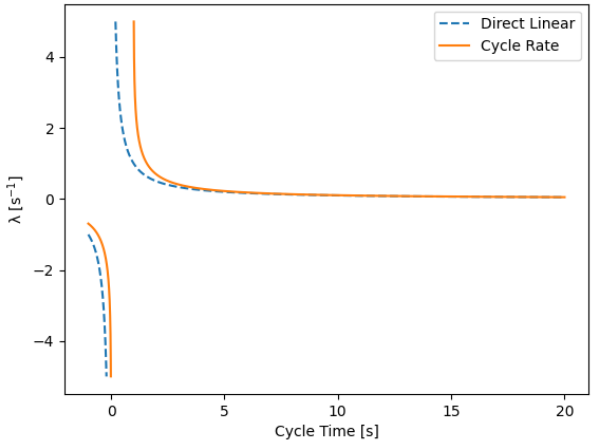


Figure 3.11: An illustrated comparison of the direct linear and cycle rate reprocessing constants for different cycle times.

Because the shortest cycle time for the MSBR is 20 seconds, this shows that the cycle rate and direct linear approaches are roughly equivalent for this reprocessing scheme.

3.3.2.6 SaltProc Direct Linear Approach

The SaltProc direct linear approach is the same as the direct linear approach, but takes into account the limiting nature of the 3 day batchwise reprocessing step used by SaltProc. For example, a 6 day cycle time target would be modeled the same using the SaltProc direct linear approach as the standard direct linear approach. However, anything shorter than 3 days would be modeled differently, since that is the batchwise reprocessing step inocorporated by SaltProc for modeling the MSBR. For example, a 30 second cycle time would be analyzed instead as a 3 day cycle time, since SaltProc can only remove 100% of material after a minimum of 3 days. This means the inverse would be 0.333 days^{-1}, or 3.858E-4 s^{-1}. Converting to a reprocessing constant gives a value of 3.858E-6 s^{-1}. This is the reprocessing constant for any cycle time which is less than or equal to three days. Additionally, the decay constant is multiplied by the fractional efficiency of the removal used by SaltProc. In most cases the value is 1, but it is roughly 0.91 for xenon and krypton in order to align with Rykhlevskii's results [38].

3.3.2.7 Decay Approaches Summary

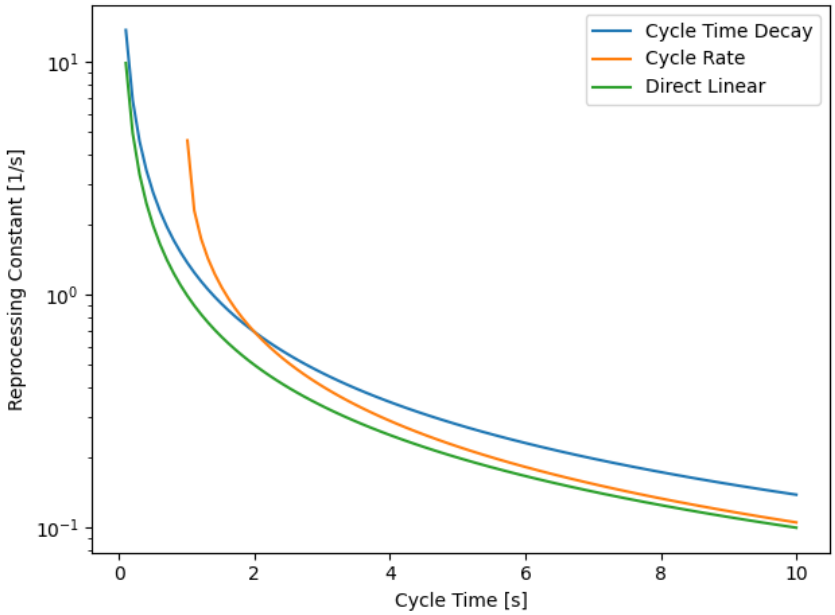


Figure 3.12: Plot of how reprocessing constants for different approaches vary with cycle time.

Figure 3.12 shows how the reprocessing constants for the different approaches vary as a function of cycle time. This figure includes the three different continuous decay based methods for reprocessing as a function of various cycle times. From the figure, it can be seen that the cycle rate method does not have results for cycle times less than or equal to 1 second, which is a significant weakness of the method since physical processes could exist with a cycle time in that range. The other two methods provide results down to a 0 second cycle time, which allows full coverage of possible cycle time values.

The direct linear method behaves similarly to the cycle time decay method during small cycle times and the cycle rate method for longer cycle times. This is due to the general form of the equations for each, where at small cycle times the inverse cycle time relationship with the cycle time decay method dominates, while at larger cycle times the cycle rate log of the inverse term matches more closely.

Table 3.3 shows all the different decay based continuous reprocessing approaches for various cycle times. From this table, it can be seen that the direct linear and cycle rate methods are very close, while the cycle time decay method is off by a small amount at all cycle times. In terms of the SaltProc variants, the main difference is the fact that the cycle times below three days have been assigned the reprocessing constant of the three day cycle time.

Table 3.3: Full Set of Decay Reprocessing Approaches for Various Cycle Times

Method	$T_{cyc} = 1s$	$T_{cyc} = 20s$	$T_{cyc} = 3d$	$T_{cyc} = 50d$
DL	1	5.00E-2	3.86E-6	2.31E-7
CR	-	5.13E-2	3.86E-6	2.31E-7
CTD	1.39	6.93E-2	5.35E-6	3.21E-7
SPDL	3.86E-6	3.86E-6	3.86E-6	2.31E-7
SPCR	3.86E-6	3.86E-6	3.86E-6	2.31E-7
SPCTD	5.35E-6	5.35E-6	5.35E-6	3.21E-7

The values shown in Table 3.3 for the SaltProc-type continuous methods are based on the cycle times given from the Robertson report, which matches what is used in the bulk SaltProc, version 0.1, results. The steady SaltProc results, verson 0.3, do not use these exact values [34, 37]. The differences in steady SaltProc are as follows: xenon has 91.2% removal over three days instead of 100%, krypton has 91.5% removal over three days instead of 100%, protactinium has 9.5% removal over three days instead of 100%, and the discard has a removal of 0.9% over three days instead of 0.09%.

3.3.3 Step Reprocessing Approach

The Step reprocessing method implemented in Serpent2 is mathematically very similar to the Decay model, but instead of updating continuously in time, it instead is updated during new depletion steps. In this manner, it could be considered a mix between the Constant method and the Decay method. This is because over a single depletion step, it is a constant added or subtracted from the Bateman equation, while over many depletion steps, it follows the same exponential decay form of the Decay method.

This particular method is not useful for extracting fission products, and is not needed for constant feed rates since that can be modeled using the Decay method. This method could be useful for inducing a step drop in feed rate, though this would require running only a single depletion step until the drop occurred. Additionally, this drop could be simulated by running the Decay method and reducing the reprocessing constant. This would allow for flexibility in the distribution of depletion steps as well without having to worry about changing the behaviour of the reprocessing functionality.

Another potential issue with the Step method is that the depletion step can last long enough that the constant

mass removal causes the mass to go negative. However, Serpent2 will cease running if this occurs. The Decay method does not mathematically allow for negative mass, the Constant method does not conserve mass, and the Step method stops running once there is negative mass.

The main potential use of the Step reprocessing may be for movement of material at a set rate. This could be implemented by writing a script to check the current number of atoms of the target and adjusting the reprocessing rate accordingly. However, for the MSBR, this form of reprocessing is not necessary, and is thus not implemented.

3.4 Mass Balancing

One of the useful features of batchwise reprocessing is that the net mass of the core can be balanced by adjusting the feed rates to provide the same amount into the core that is lost. Alternative methods also exist, such as removing excess mass or increasing the volume [33]. SaltProc version 0.1 does not account for mass balancing but maintains a constant thorium mass, whereas SaltProc version 0.3 has the feed rates set to maintain mass. Mass balancing is particularly important in Serpent2 due to the way masses in Serpent2 are handled.

In Serpent2, an increase in mass does not affect volume, but instead increases the density of that isotope in the material accordingly. This affects macroscopic cross section calculations, and can lead to variation in results if not accounted for, since in reality volumetric expansion could be assumed. Though there are several methods to handle mass balancing with a batch method, it not currently continuously possible in Serpent2.

In order to balance the mass in Serpent2 continuously, one approach could be to iteratively perform depletion calculations while updating feed rates until a balanced mass is found while also minimizing some other parameter, such as net mass of material added. However, the current reprocessing options available in Serpent2 only allows for pseudo-constant and decaying feed rates over the depletion step. With those two feed types available, it is not possible to have a constant mass balance.

It is possible to have the masses at the end of each depletion step remain constant, but this does handle mass fluctuations during depletion. It does, however, solve the issue of the density variation causing a difference in cross sections.

Overall, if the net mass difference is sufficiently small, it does not have to be considered since the results would not be significantly impacted. To check this for the MSBR, I will use an illustrative example. To determine the maximum possible increase in density, the mass loss due to fission and reprocessing is neglected, and only mass addition from the thorium feed rate is considered. The uranium feed is not added because it is equivalent to the protactinium removal, so it would have a negligible impact on net mass.

$$\Delta m = \dot{m}_{feed} \Delta t \approx (2.5)(6000) = 15,000 kg \tag{3.45}$$

It can be seen in Equation (3.45) that the net mass gain for an average feed rate of 2.5 kilograms per day over 6000 days is 15,000 kg [38, 8] while operating at nominal power of 2250 MW [34]. This does seem to be a very large value, but the importance of the mass in the depletion calculation is primarily in how it affects cross sections, which means that the impact on the overall thorium density is important. The thorium density is 1.46 grams per cubic centimeter, and the net volume is approximately 48.71 cubic meters. The density with the added mass is calculated using Equation (3.46).

$$\rho_f = \frac{m_0 + \Delta m}{V} = \frac{(4.871E7\left[cm^3\right])(1.45919E\text{-}3\left[\frac{kg}{cm^3}\right]) + (15,000\left[kg\right])}{4.871E7\left[cm^3\right]} \tag{3.46}$$

The final density comes out to be 1.767 grams per cubic centimeter, which is a percent difference of 21%. Although this is an upper bound on the mass difference, a 21% difference in the expected cross section would result in significant error in the results. Therefore, since it is possible for the mass balancing to have an impact, it should be investigated to ensure the mass balancing is not impacting the results in any unexpected manner.

3.5 Effects of Delayed Neutrons on Depletion

Delayed neutrons have a softer energy spectrum and drift along with the movement of the fuel salt in a fluid fueled molten salt reactor. This means that the delayed neutron precursors, of which some fraction leave the core and some move to less neutron important regions, have the potential to alter the depletion results of the MSBR model.

However, it has been shown by Zhou et al that for the MSBR, the delayed neutron precursor drift has a negligible impact on depletion results [48]. Although Zhou et al has shown this, it is still worth considering the maximum possible effect delayed neutron precursors could have on depletion results. In order to determine this, the Serpent2 functionality of disabling delayed neutrons is employed [24]. Using this method, two different models are generated of the MSBR. The first is one in which the delayed neutron precursors are evenly distributed within the core, which is the model implemented by SaltProc and this work [39]. The second model is one in which delayed neutrons no longer exist. This is beyond the greatest effects the delayed neutron precursor drift could have on depletion, so the absolute maximum effect of delayed neutron precursors on depletion results can be determined.

Because the delayed neutron precursors are fission products or come from the decay of fission products, depletion must be performed. For reprocessing, the average feed rate from SaltProc is used with continuous direct linear reprocessing, while direct linear reprocessing is used for fission product removal.

I investigated the impact on k_{eff}, and I determined that the delayed neutrons from fission products add roughly 20 pcm to k_{eff} after 6,000 days of operation, which is roughly steady state operation. This was determined by running the simulation with and without delayed neutrons enabled and comparing the k_{eff} values. I also compared the masses of various isotopes, such as uranium-235 and thorium-232, though no statistically significant difference was found. Therefore, the net impact of delayed neutrons from fission product precursors appears to be negligible on the depletion results in this work, which agrees with the results from Zhou et al.

Chapter 4

Results

4.1 Depletion Time Step Refinement

As described in Section 2.5, there are many different methods available for modeling the online reprocessing of molten salt reactors. Previously, Rykhlevskii has generated results for the molten salt breeder reactor using the bulk batchwise method [37]. This method is primarily useful due to the straightforward nature of its implementation. However, it is less accurate and more costly than other methods, as discussed in the Methodology section. This is the reason that Rykhlevskii continued to develop SaltProc from bulk batchwise reprocessing in version 0.1 into steady batchwise reprocessing in version 0.3. The steady batchwise method has approximately the same computational cost, but provides more accuracy by applying reprocessing during each depletion step available. This provides a far better approximation of continuous reprocessing.

The continuous methods are all similar in terms of computational cost. The main difference is how cycle time data is interpreted. Because the cycle times given are the amount of time to remove 100% of a target, and the continuous methods all use exponential terms, different strategies are used. The most straightforward method is direct linear, seen in Section 3.3.2.5, while the methods of cycle rate and cycle time decay, Sections 3.3.2.3 and 3.3.2.1, respectively, are based on assumptions.

In the Methodology section, I discussed the theoretical basis of these methods. In this chapter, I will demonstrate the practical application of each in order to compare them. I will first perform a depletion time step refinement study in order to determine the effects of depletion step on the effective multiplication factor and nuclide compositions. Intuitively, it is expected that the batchwise methods will perform worse than the continuous methods as the step size grows larger. This is due to the less frequent application of reprocessing on a process which is intended to be occurring continuously. The less frequent depletion steps also means the cross section and flux data is not updated as the fuel is depleted, leading to less accurate results.

To illustrate the impact of less frequent application of reprocessing, consider a simple thermal-spectrum system where uranium is fissioning and reprocessing is taking place. If xenon is physically continuously reprocessed, then the total neutron absorption in the core by xenon is reduced. However, using a longer batchwise step size results in

xenon forming and capturing neutrons after the depletion simulation forms the xenon, the transport simulation updates the cross sections for the xenon, and another depletion step is run where the xenon has a capture cross section. Only after the depletion step size time elapses is batchwise reprocessing performed and the xenon is removed. By this point, many neutrons have already been captured, affecting reactor performance.

I will show two types of results to demonstrate the effect of altering the depletion step. The first result I will show is the effect on the composition of some select isotopes, as altering the depletion step directly affects the concentration of isotopes affected by reprocessing. The other result I will show is the effective multiplication factor, which is directly affected by the change in the composition. Through these results I show the direct and indirect impacts of changing the depletion step size.

4.1.1 Batchwise Reprocessing

4.1.1.1 Bulk Batchwise

Inclusion of a depletion step refinement study of bulk batchwise is not performed, as it can only strictly perform worse than steady batchwise methods. The reason for this was discussed in Section 3.2. However, the refueling rates from the MSBR work performed by Rykhlevskii using the bulk batchwise method are used here, and the bulk batchwise method itself is implemented in Section 4.3.2.1 to compare with continuous methods [37].

4.1.1.2 Steady Batchwise

A depletion step refinement study using the steady batchwise reprocessing version of SaltProc was performed by Rykhlevskii comparing 3, 6, 12, and 24 day depletion steps over 25 years for the Transatomic Power Molten Salt Reactor [38]. Rykhlevskii used a three day depletion step in the bulk batchwise analysis of the MSBR over 20 years [37]. Rykhlevskii and Powers et al. both suggest the three day depletion time step size to correlate with the removal interval of protactinium-233 [38, 32]. In this work, the depletion time step of three days will be used for the steady batchwise method.

4.1.2 Continuous Reprocessing

For the continuous methods, the error buildup from using larger depletion steps is due to not updating various simulation data rather than from lack of updating the material compositions based on reprocessing. Due to this, larger time steps can be considered. In order to determine the impact of the updated simulation data, initially chosen depletion time steps of 1, 3, 6, 15, and 30 days are implemented. Additionally, although different continuous methods have been introduced, the depletion time step refinement only needs to consider one of the methods. This

is because the continuous methods used all have the same general form and only alter the specific reprocessing constants implemented. Thus, the direct linear method is selected to determine the optimal depletion step size for the continuous methods. Additionally, it has been used in several other works and is very similar to the cycle rate method [47, 29, 48].

4.1.2.1 Submonthly Depletion Steps

The results from using depletion steps of 1, 3, 6, 15, and 30 days can be seen in Figures 4.1 through 4.5. The results regarding the effective multiplication factor show that, after 30 days, all of the effective multiplication factors are within stochastic error of each other.

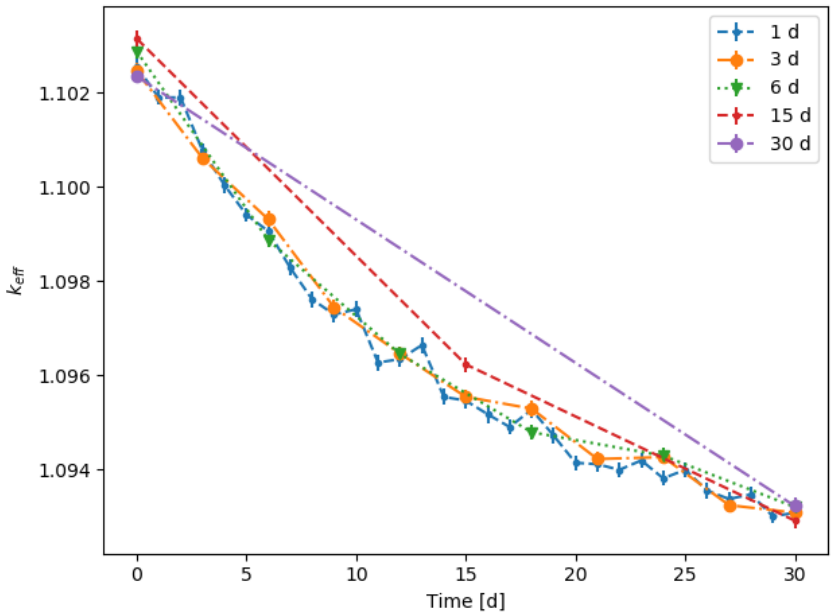


Figure 4.1: k_{eff} over time using various depletion step sizes with direct linear continuous reprocessing.

For the thorium-232 and uranium-233 masses, the values align within fractions of a percent of each other after 30 days of depletion. Additionally, it can be seen that there are sudden jumps in the thorium-232 mass. This is because of rounding which occurs due to a limit to the precision of the data which is available from Serpent2.

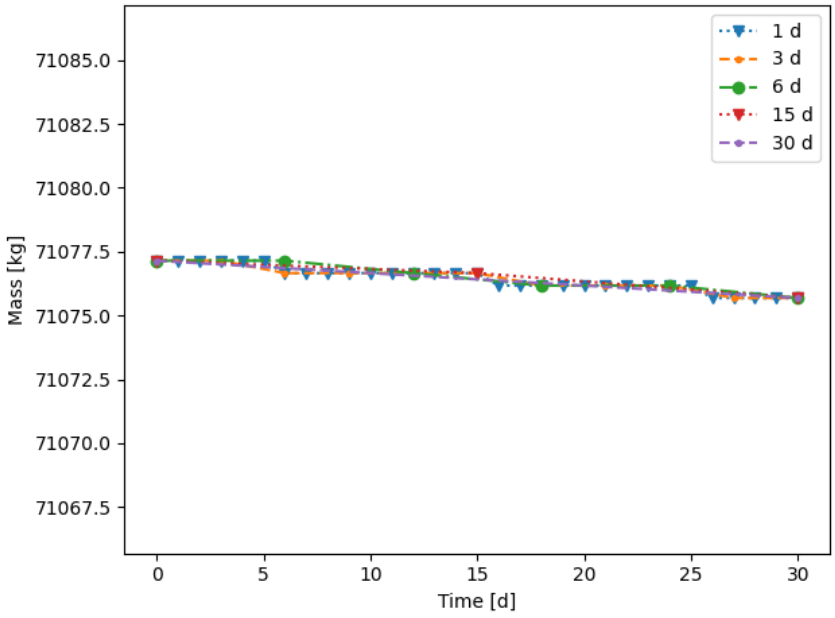


Figure 4.2: Thorium-232 mass over time using various depletion step sizes with direct linear continuous reprocessing.

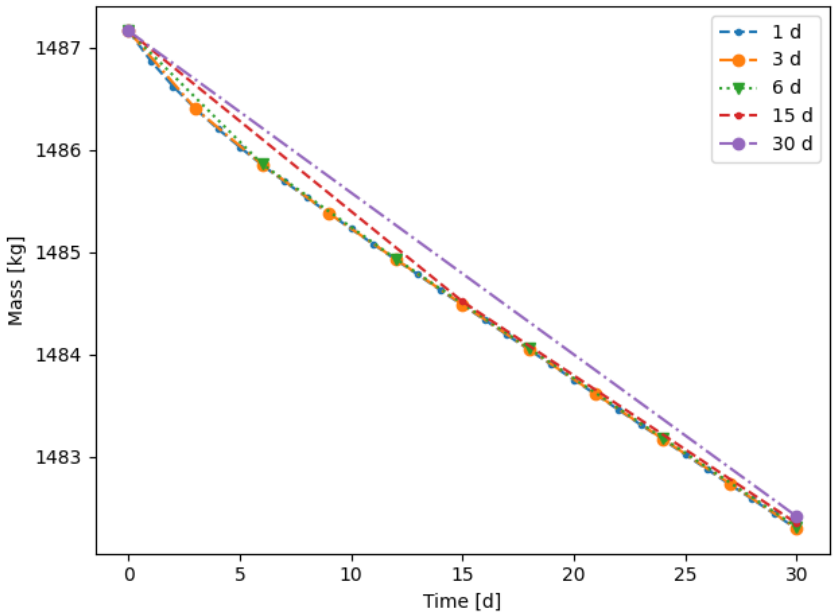


Figure 4.3: Uranium-233 mass over time using various depletion step sizes with direct linear continuous reprocessing.

The xenon-135 steady state mass is the same for each of the different step sizes, which is expected. Additionally, the xenon-136 steady state mass is also the same for the different depletion step sizes, showing that the capture rate

of the xenon-135 does not vary significantly even when adjusting the depletion step size from 1 day to 30 days.

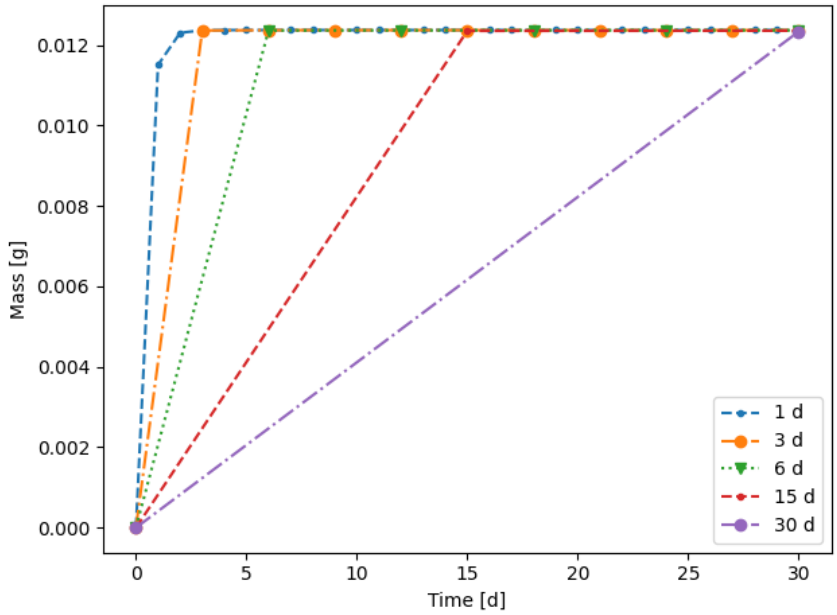


Figure 4.4: Xenon-135 mass over time using various depletion step sizes with direct linear continuous reprocessing.

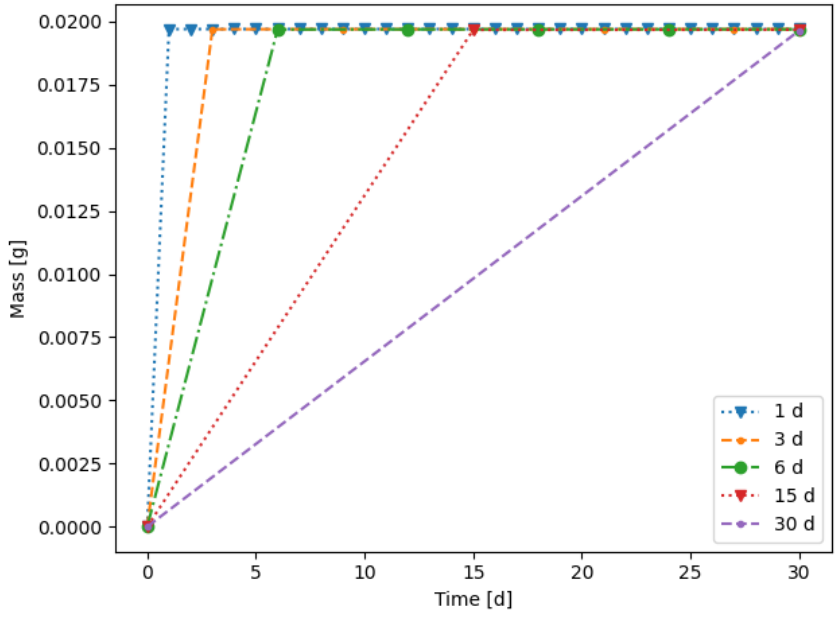


Figure 4.5: Xenon-136 mass over time using various depletion step sizes with direct linear continuous reprocessing.

Because these results are all very close to each other, it is prudent to test for larger depletion step sizes in order to determine how much computational cost reduction can be afforded while maintaining high accuracy in the

results. Although using a 30 day depletion step is already an order of magnitude larger step size than that used in the batchwise method in this work, further coarsening of the time step would allow for further improvements to the computational cost reduction.

4.1.2.2 Larger Depletion Steps

The next set of depletion step sizes selected are 30, 60, 120, and 360 days. The smallest step size, 30 days, is selected in order to properly compare the larger depletion step sizes with a step size which has been shown to give results which are very close to those of a 1 day depletion step size. The results can be seen in Figures 4.6 through 4.10.

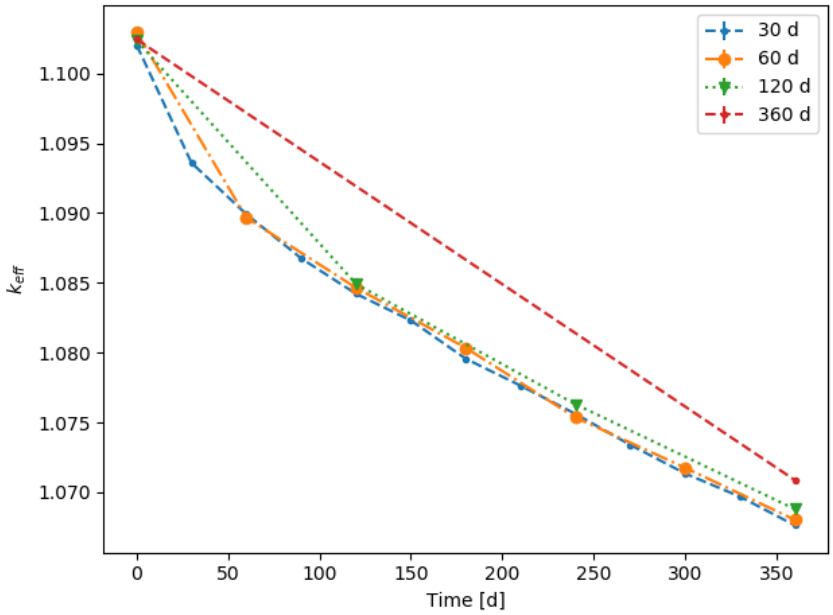


Figure 4.6: k_{eff} over time using various depletion step sizes with direct linear continuous reprocessing.

The 60 day depletion time effective multiplication factor at 360 days is not within stochastic error of the value for the 30 day depletion time. The error bounds are only 5 pcm apart, but since the computational cost decrease is only a factor of two, the 30 day depletion step size is sufficiently large for the cases implemented in this work while maintaining a high level of precision and are used as the default when not specified. The largest uncertainty in the effective multiplication factor here is 17 pcm.

The thorium-232 and uranium-233 masses follow a similar trend where the larger depletion step size has a higher mass of both isotopes. The difference in thorium can be seen by visualizing how the larger step sizes operate similarly to a tangent line from the shorter depletion step sizes. This is because the cross section and spectral data is not updated, and so the same rate of decrease is used over the entire depletion step. Because the system starts at beginning of cycle, the thorium is not yet in steady state and more is burned than added, so the decrease is

expected.

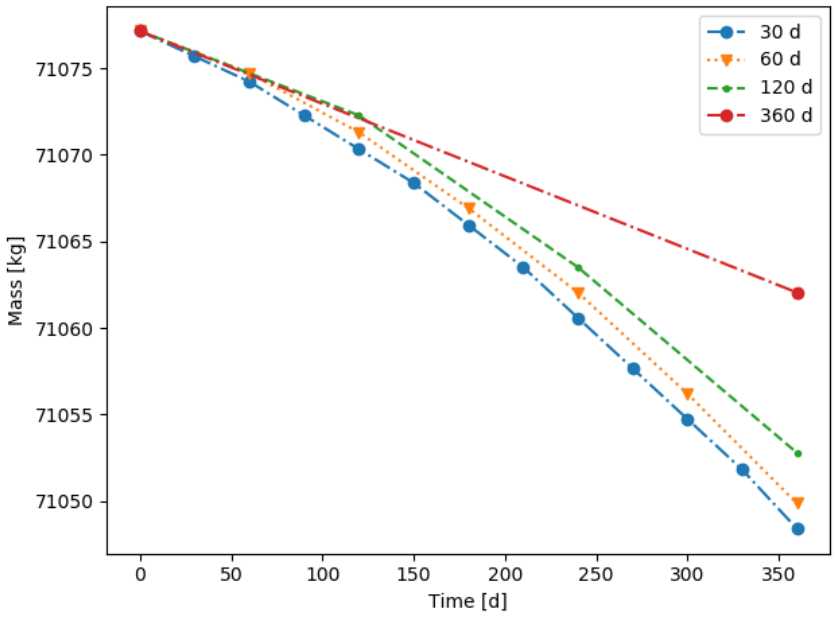


Figure 4.7: Thorium-232 mass over time using various depletion step sizes with direct linear continuous reprocessing.

For the uranium-233, a similar trend can be seen, though it is less pronounced. The main difference with thorium-232 is the fact that the thorium mass changes due to feed and neutron absorption, whereas the uranium-233 has feed, neutron absorption, and accumulation from the decay chain of the thorium-232 absorption. With this in mind, the difference of the uranium-233 and thorium-232 in Figures 4.7 and 4.8 becomes more clear. For the shorter depletion steps, a similar trend as in the thorium-232 occurs where the rate is matches for the depletion step. However, the thorium-232 is also a factor, and since the thorium-232 has a greater mass for larger depletion step sizes, this results in more uranium-233 being bred as well.

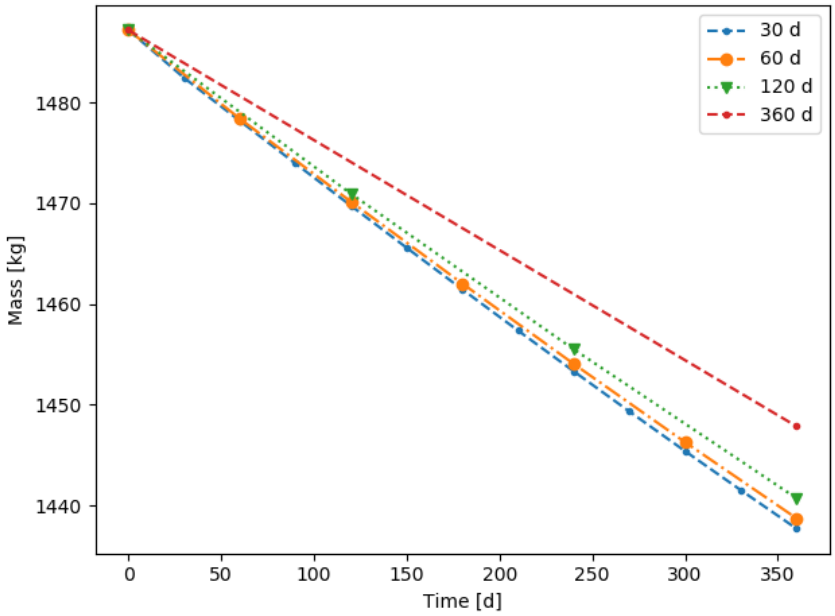


Figure 4.8: Uranium-233 mass over time using various depletion step sizes with direct linear continuous reprocessing.

The xenon-135 and xenon-136 masses match closely, with differences on the order of milligrams. This agrees well with the previous results generated as well. The reason the results agree well is because the xenon-135 concentration is expected to approach a constant result with increasing flux. This is because the xenon-135 comes from the decay of iodine-135, which scales proportionally to the flux. However, the xenon-135 has a large capture cross section, and is altered into xenon-136 proportionally to the flux as well. This shows why there is not a large difference even though there is a difference of 324 pcm between the effective multiplication factors of the 360 and 30 day depletion step sizes.

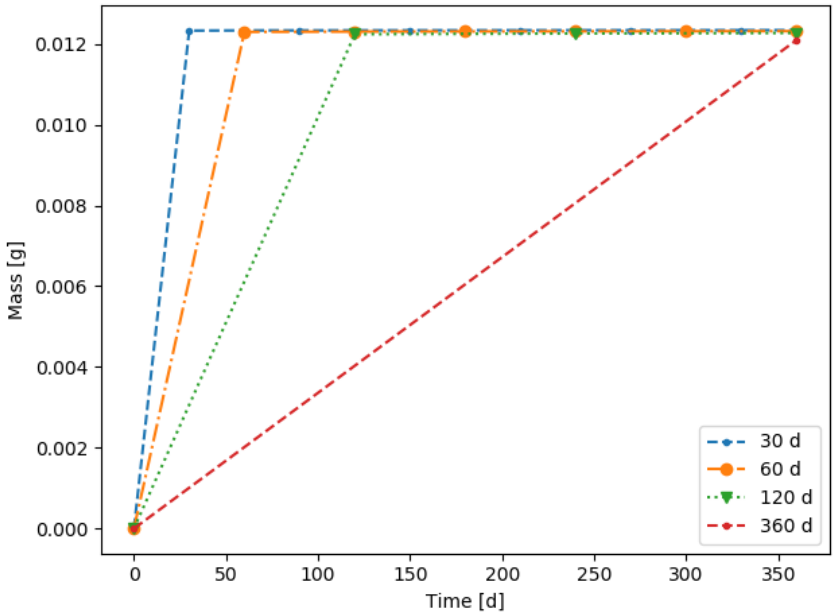


Figure 4.9: Xenon-135 mass over time using various depletion step sizes with direct linear continuous reprocessing.

The xenon-136 is also captured with a rate proportional to the flux, though the capture cross section is significantly lower than that of xenon-135 for the thermal spectrum used in the MSBR. This can be seen in the mass of xenon-136, which has a higher steady state concentration that that of xenon-135. The main limiting factor on both of these xenon isotopes is the reprocessing term, which keeps the steady state mass of both much smaller than they would be without the reprocessing.

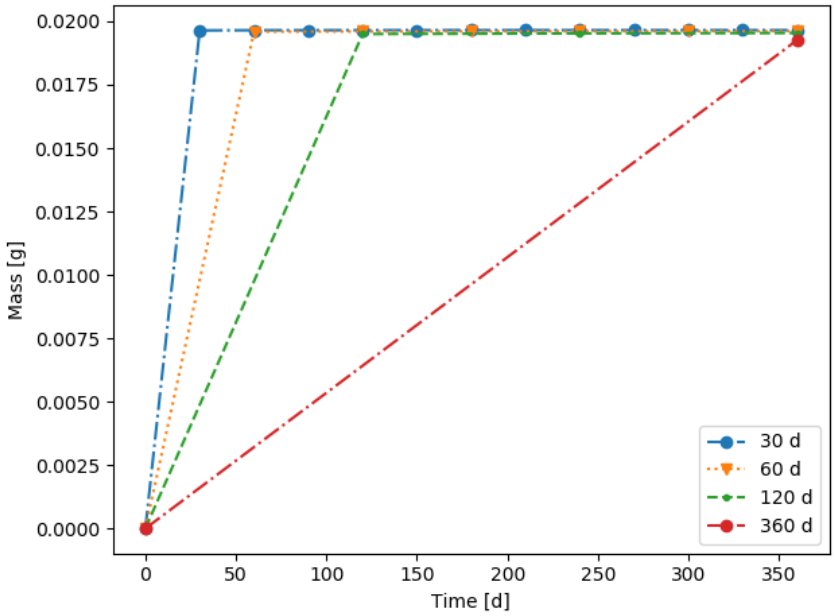


Figure 4.10: Xenon-136 mass over time using various depletion step sizes with direct linear continuous reprocessing.

4.1.3 Variable Depletion Step Size

Previously, the depletion step size needed for a reasonably converged result was investigated using a constant depletion step size. However, it is understood that the reactor will experience some rapid initial changes, after which the changes slow and eventually the reactor reaches steady state. Therefore, it is reasonable to expect that in the early times, small depletion step sizes should be used, while at later times, larger depletion step sizes should be used.

Because it was previously shown for continuous reprocessing that the depletion step sizes smaller than 30 days give results within stochastic error for the effective multiplication factor, the 30 day depletion step size will be the smallest considered step size. Several cases are considered over a net depletion time of 360 days in order to determine the effects of varying the depletion step size at various points in the simulation.

The depletion step sizes used for each case are given in Table 4.1, and the effective multiplication factors at 360 days are given in Table 4.2 for each case.

Table 4.1: Test Cases' Variable Step Sizes

Case	30 Days	90 Days	180 Days	330 Days	360 Days
A	12	0	0	0	0
B	6	0	1	0	0
C	3	1	1	0	0
D	1	0	0	1	0
E	6	0	1	0	0
F	0	0	0	0	1

By default, the cases use the shortest time step before the longer steps. However, Case E varies from this, allowing for it to be directly comparable to case B. Case E uses the 180 day step first, followed by the six 30 day steps.

Table 4.2: Variable Depletion Step Size Results

Case	k_{eff}	Δk_{eff} [pcm]	Simulations
A	1.06801 ± 0.00017	-	12
B	1.06844 ± 0.00015	43 ± 32	7
C	1.06857 ± 0.00016	56 ± 33	5
D	1.07034 ± 0.00016	233 ± 33	2
E	1.06846 ± 0.00016	45 ± 33	7
F	1.07071 ± 0.00017	270 ± 34	1

For the result of the effective multiplication factor, the difference from Case A was taken for the rest of the cases as case A is assumed to be the most realistic. These results show that Cases B, C, and E are all very close to Case A. Cases D and E are the outliers, which is also reflected in the number of simulations run. Case D uses a short depletion time step initially followed by a single long step, which should have allowed for the simulation data to update based on the presence of fission products which were not initially present. However, this impact is likely offset by the continuous reprocessing, resulting in more impact purely due to the general changes in the system. It seems that multiple steps are needed during the first 360 days of operation in order to properly account for the changes within the reactor, such as Case E, which leads with a depletion step size of 180 days but then proceeds to follow it with six 30 day depletion steps.

The variable depletion step size could be considered to have an impact if the reactor were at steady-state, but after only 360 days of operation, there are still changes occurring [37]. This is the reason why having more simulations yielded better results than the cases which had fewer simulations.

4.1.4 Computational Cost Analysis

Computational cost of each method is presented in Table 4.3. This table represents the compuational cost of forming the depletion step mesh refinement study in the units of "number of simulations required," as this is the largest impact on run time. This unit must include the depletion matrix solve, but it may also include the transport solve time, data loads, and other overhead costs. To convert to units of time, the average time per simulation can be calculated and then multiplied by these values. In this case, the time cost per simulation will be represented using τ, which is larger for batchwise, τ_b, than continuous, τ_c. The depletion step size used by the batchwise methods is 3 days, while the continuous method uses 30 day depletion steps.

For unoptimized batchwise, the number of simulations is roughly double what is actually required, as the pre-reprocessed simulation is also run. An optimized batchwise method could remove the double running, which would significantly reduce computational cost. Additionally, as can be seen in the formula for calculating the cost, reduction can be achieved by maximizing the depletion step size or reducing the net depletion time, or T_{net}.

Table 4.3: Computational Cost Using Constant Depletion Steps

Method	Cost Formula	30 days	1 year	10 years
Unoptimized Batchwise	$2T_{net}\Delta t^{-1}\tau_b + \tau_b$	21 τ_b	245 τ_b	2436 τ_b
Optimized Batchwise	$T_{net}\Delta t^{-1}\tau_b + \tau_b$	11 τ_b	123 τ_b	1219 τ_b
Continuous (All)	$T_{net}\Delta t^{-1}\tau_c + \tau_c$	2 τ_c	14 τ_c	123 τ_c

Because continuous reprocessing enables the use of depletion time steps on the order of 10x larger while retaining a similar level of precision, the computational cost is significantly lower than the batchwise computational cost.

4.2 Advanced Protactinium Decay Model

The original method used by Rykhlevskii to model the decay of protactinium-233 into uranium-233 was to feed a mass of uranium-233 into the core equivalent to the mass of protactinium-233 pumped out every 3 days [37]. With

the implementation of continuous reprocessing methods, the breeding can be more accurately modeled. In the new method, protactinium-233 is continuously removed and allowed to decay. Once it decays, the uranium-233 is then continuously added back into the core. These two methods can be directly compared. I will refer to them here as "SaltProc Average" for the average feed into the MSBR calculated using SaltProc and "Decay Tank" for the model that continuously removes protactinium and adds the uranium generated via decay back into the core.

It is expected for the largest impact to be at beginning of cycle, while at steady state this improved method will likely have only a minor effect on the results. The results of this analysis can be seen in Figures 4.11 through 4.15, where the continuous method implemented for both was the direct linear method, and 305 day depletion steps were used over a period of 3660 days, or roughly ten years. Although this depletion step size is larger than those previously used, it is expected to still give reasonable accurate results. Additionally, the discard and salt discard processes are not considered here, as only the differences between the different feed rate methods are being investigated.

The effective multiplication factor for the realistic uranium-233 feed approach can be seen to drop rapidly, and then begin leveling off after around 3 years. The reason for this difference in behaviour is because the feed which uses the SaltProc averaged value makes a steady state assumption, which allows for a full supply of uranium-233 available for feeding the reactor at all times. In can be seen that the slopes of the two different methods are beginning to converge slowly after a few year, and it is expected that they would eventually match. However, if the batch discard processes are included, a true steady state equilibrium would not be reached, resulting in at least some minor differences between the methods. The largest stochastic error in the effective neutron multiplication factor is 19 pcm, and so is too small to appear in Figure 4.11.

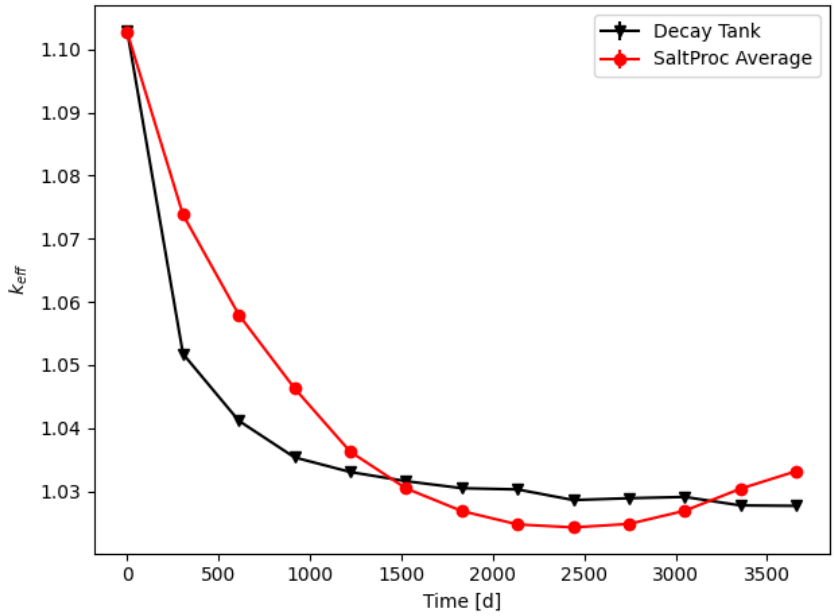


Figure 4.11: k_{eff} over time using various depletion step sizes with direct linear continuous reprocessing comparing the decay tank and SaltProc averaged feed rates.

The thorium-232 masses have a fairly constant difference from each other over the entire run. The reason for this is likely due to the fact that there is less uranium-233 being added to the system with the realistic approach, meaning more interactions with thorium-232 occur instead. This results in a lower effective multiplication factor, at least initially, which provides increased uranium-233 production, and lower thorium-232 mass.

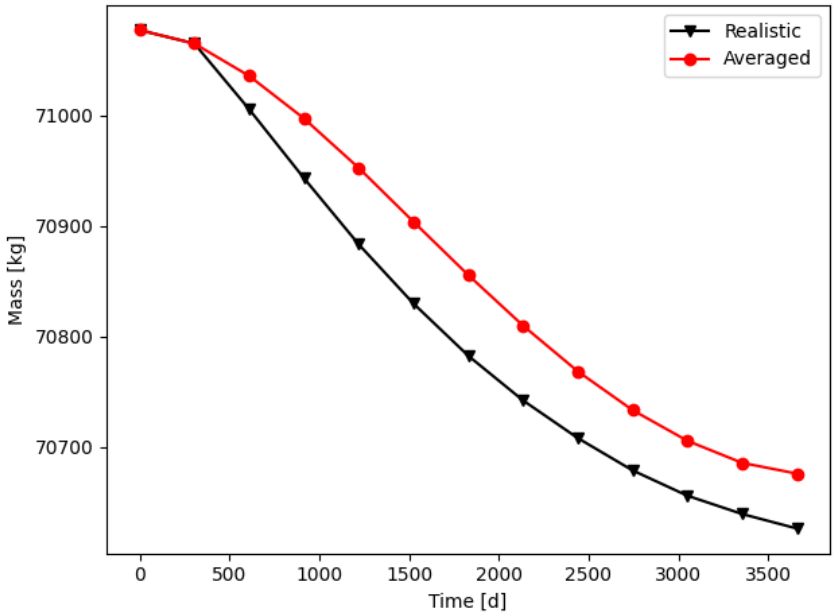


Figure 4.12: Thorium-232 mass over time using various depletion step sizes with direct linear continuous reprocessing comparing the decay tank and SaltProc averaged feed rates.

The uranium-233 for the realistic method follows a trend which is expected, which is a sharp initial drop followed by a decrease in the rate of change. After the sharp drop, the protactinium decay tank fills, and fresh uranium-233 is supplied to the system in an increasing quantity. For the averaged method, because the uranium-233 was burned at a higher rate, this leads to a decrease in the total amount. It is expected that there will be a few oscillations in the uranium-233 mass for the averaged system until it finally achieves a steady state value, though this is only true if the system only has continuous reprocessing processes. This oscillation is expected because initially, there is more uranium-233 which fissions, rapidly decreasing the quantity while maintaining a higher effective multiplication factor. This causes the uranium-233 to burn quickly, dropping the net mass until it reaches a point that the effective multiplication factor decreases while the uranium-233 mass is also lowered, reducing probability of fission as well as the number of neutrons. This allows the feed rate to build up more uranium-233 in the system, repeating the process.

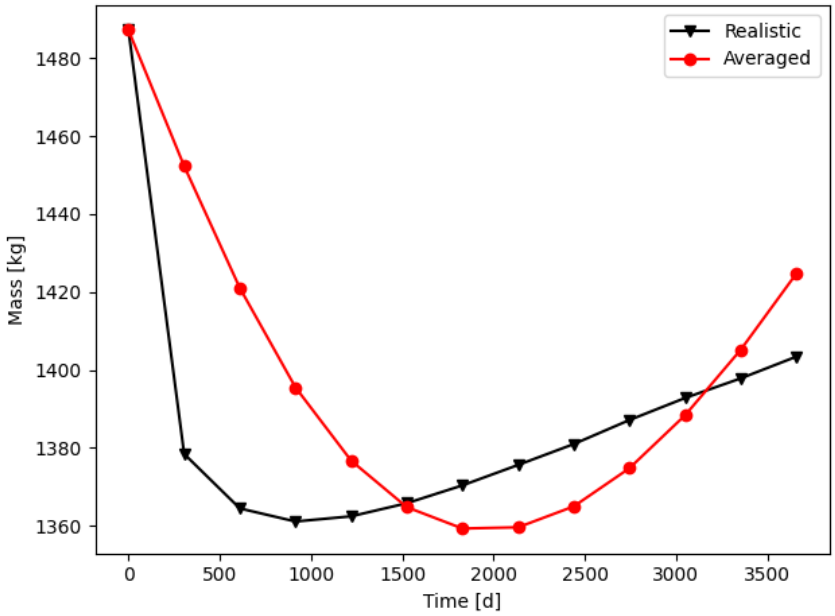


Figure 4.13: Uranium-233 mass over time using various depletion step sizes with direct linear continuous reprocessing comparing the decay tank and SaltProc averaged feed rates.

For the xenon-135 and xenon-136, the masses are slightly smaller than the averaged method initially, which is due to the decreased uranium-233, resulting in fewer fission products being generated. However, after more time passes, the values match. This is due to the steady state value of xenon-135 not significantly changing when the flux is already large.

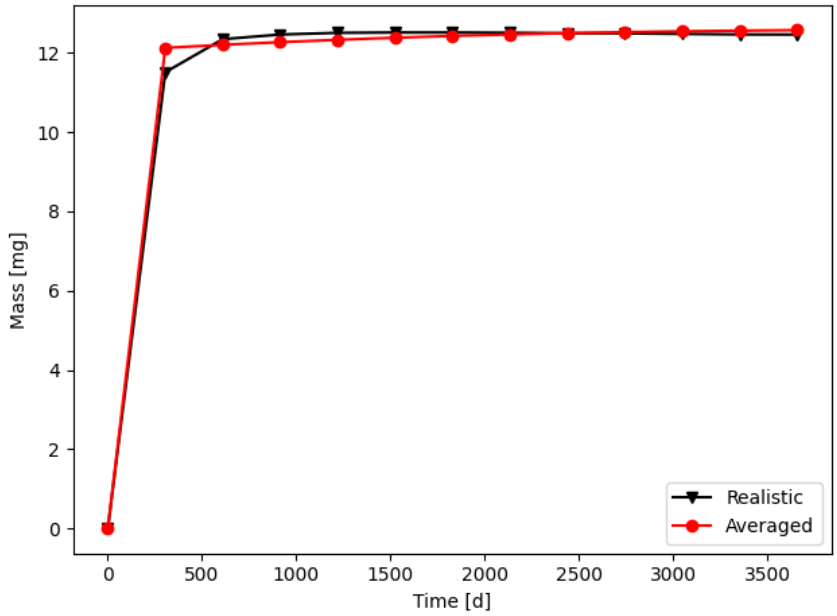


Figure 4.14: Xenon-135 mass over time using various depletion step sizes with direct linear continuous reprocessing comparing the decay tank and SaltProc averaged feed rates.

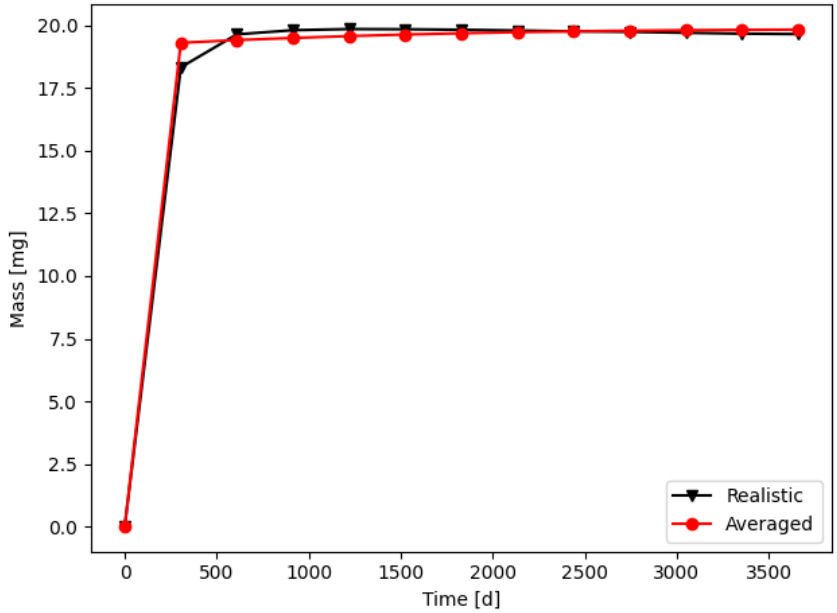


Figure 4.15: Xenon-136 mass over time using various depletion step sizes with direct linear continuous reprocessing comparing the decay tank and SaltProc averaged feed rates.

4.3 Comparisons of Methods

There are many different works which have implemented batchwise reprocessing computational methods to simulate a continuous, online reprocessing scheme, which are presented in Section 2.5.2. It follows that the accuracy of each method should be resolved. From previous work by Rykhlevskii, as well as through the depletion step mesh refinement provided in this work, it has been shown that increasing the depletion step size causes batchwise methods to become less effective at simulating the physical, continuous process of online reprocessing [38]. This can be seen in Section 3.2.3.1 where the theoretical impact is discussed and Section 4.1.1.2 where the impacts are shown in the MSBR.

4.3.1 Continuous Methods

Sections 4.1 and 4.2 have exclusively used the direct linear continuous reprocessing method due to similarities between the continuous methods. However, it is useful to determine how the different continuous methods compare when applied to an MSBR model. For this comparison, the steady SaltProc-based averaged feed rates and modified cycle times are used, shown in Section 3.2.2. A 30 day depletion step size is also used, the reason for which is provided in Section 4.1.2.

The effective multiplication factor, over the course of 360 days, has a difference on the order of 500pcm between the cycle time decay, CTD, method and the direct linear and cycle rate, DL and CR, methods respectively. These methods are described in detail in Sections 3.3.2.1, 3.3.2.5, and 3.3.2.3, respectively. The direct linear and cycle rate methods are very close to each other, which is expected since the reprocessing constants are also very similar. The control method, CTRL, shows the effects of no reprocessing and no feed. The maximum stochastic uncertainty in these values is 17 pcm.

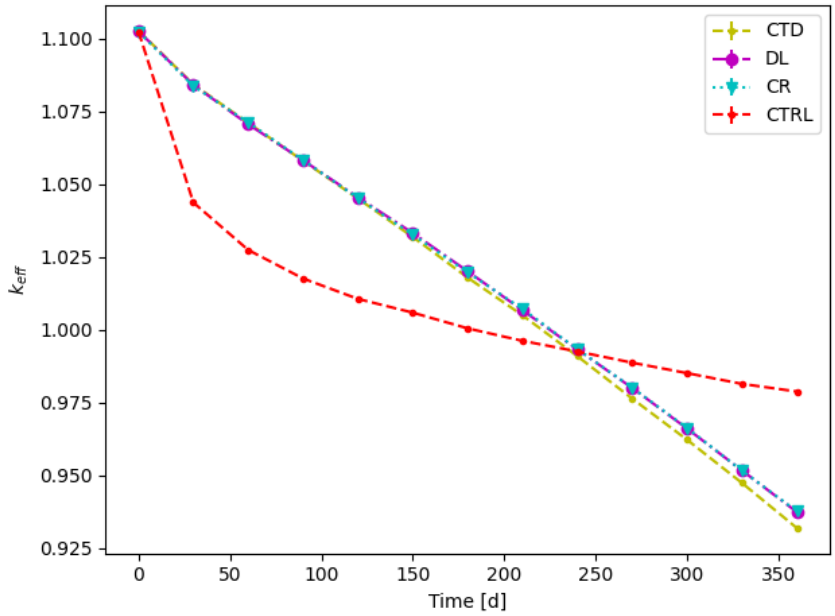


Figure 4.16: k_{eff} over time comparing different continuous reprocessing methods.

The reason the continuous methods shown here continue decreasing and are eventually overtaken by the control reprocessing is due to the protactinium reprocessing. Because the protactinium in these methods is continuously removed and the uranium-233 feed is averaged only over the first 90 days of the steady batchwise method, it is expected that the continuous model will become inaccurate some time after 90 days. This accuracy can be improved by implementing the advanced protactinium decay model which removes the dependency on an average uranium-233 feed rate.

However, the main purpose of this figure is to demonstrate how the different continuous methods compare to each other over a long period of time. Additionally, this figure shows how the effective multiplication factor without reprocessing drops below critical after roughly 300 days, and that having either a good estimate for the uranium-233 feed rate or a continuous method to generate the uranium-233 feed rate is very important for depletion modeling of the MSBR.

4.3.2 Continuous and Batchwise Methods

The continuous method selected for the comparison is the direct linear reprocessing method. For the batchwise methods, the bulk and steady batchwise reprocessing schemes from SaltProc early and current versions are used. For the bulk batchwise reprocessing, the cycle times from the Robertson report are followed [34]. Additionally, the uranium-233 and thorium-232 feed rates in the continuous model use the values from the bulk batchwise results [37].

For the steady batchwise comparison, the modified cycle times are used which include an efficiency term on the xenon, krypton, and protactinium reprocessing constants. For the continuous comparison, the average thorium-232 and uranium-233 feed rates are generated from the 90 day runs and are implemented.

For all of these comparisons, the direct linear reprocessing method uses the average feed rate of the given batchwise method and the cycle times from the Robertson report [34]. The SaltProc direct linear reprocessing method, which is intended to be a continuous approximation of the batchwise reprocessing method, will match the reprocessing constants used in the given batchwise method.

If the results are sufficiently close, then it may be reasonable for a continous method to be implemented in place of a batchwise method when the reprocessing scheme calls for a physically batchwise processes. This would allow for reduced computational cost and potentially large depletion step sizes to be implemented. If the results are not close, then these results will provide information on how the difference in reprocessing methods can affect results.

4.3.2.1 Continuous and Bulk Batchwise Methods

In this comparison, there are three different methods considered. These methods are the continuous direct linear, DL, method, the continuous SaltProc direct linear, SPDL, method; and the bulk batchwise SaltProc method. Figure 4.17 shows how the effective multiplication factor varies over time for these methods, with stochastic error less than 20 pcm. It can be seen that there is a jump at around 50 days for the bulk SaltProc data, which is caused by removal of the rare earths, as shown in the MSBR reprocessing scheme [34]. However, it is clear to see here that the continuous methods are not able to match the bulk batchwise reprocessing results very closely at all, even using the SaltProc direct linear method which attempts to replicate the same reprocessing constants.

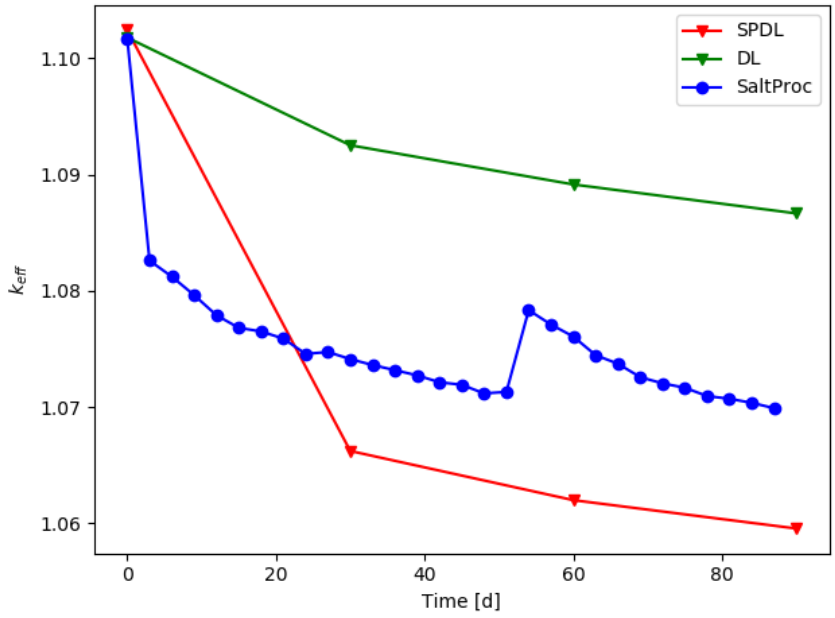


Figure 4.17: k_{eff} over time comparing bulk batchwise results to continuous methods.

Figures 4.18 and 4.19 show the mass of thorium-232 and uranium-233 over time, respectively. Overall, the masses of both isotopes are within a few kilograms, and do not have a large impact on the differences between the methods.

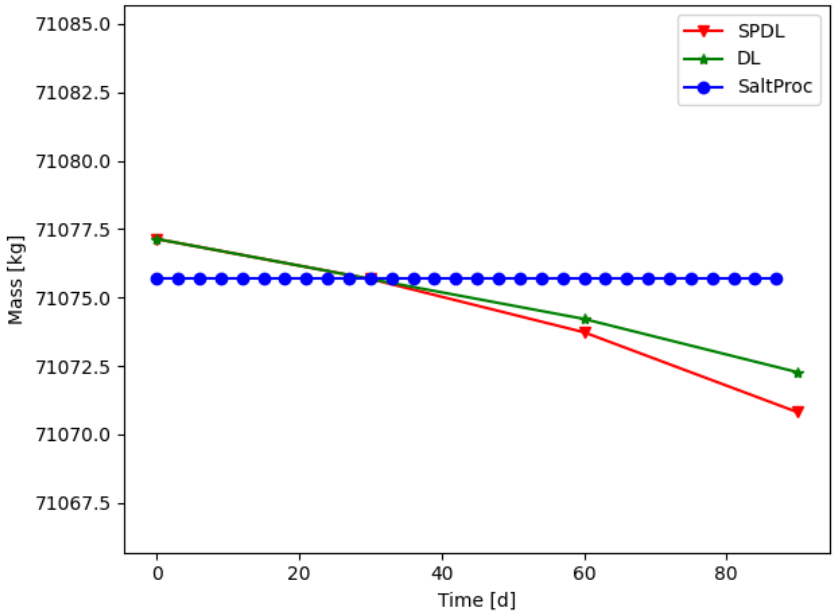


Figure 4.18: Thorium-232 mass over time comparing bulk batchwise results to continuous methods.

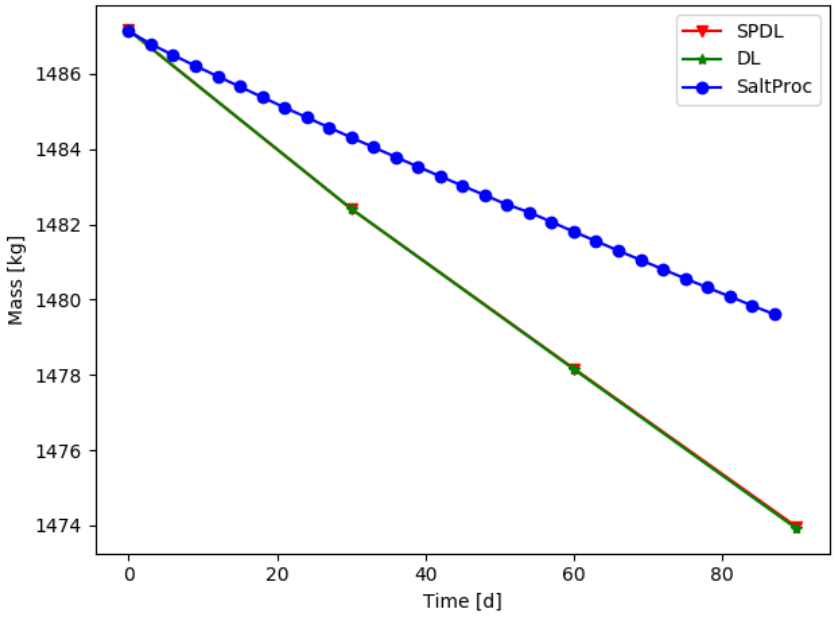


Figure 4.19: Uranium-233 mass over time comparing bulk batchwise results to continuous methods.

Figures 4.20 and 4.21 show how the xenon-135 and xenon-136 masses, respectively, change over time for the different methods. However, since this version of SaltProc removes 100% each depletion time step, resulting in a flat line of zero grams, the mass before reprocessing is shown instead. This gives a better idea of the amount of xenon which is in the system most of the time. From these figures, it can be seen that the SPDL continuous method does a fairly good job of matching the bulk batchwise reprocessing method's masses. However, the SPDL method allows for slightly more fission product poisons to exist, such as xenon-135, which allows for more parasitic absorption, into isotopes such as xenon-136. This is the reason that the SPDL effective multiplication factor is lower than that of bulk SaltProc.

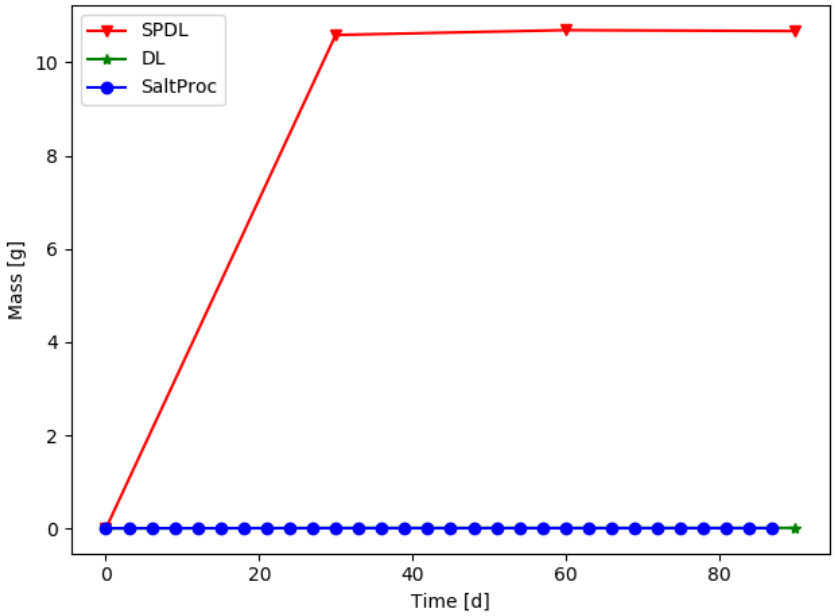


Figure 4.20: Xenon-135 mass over time comparing bulk batchwise results to continuous methods.

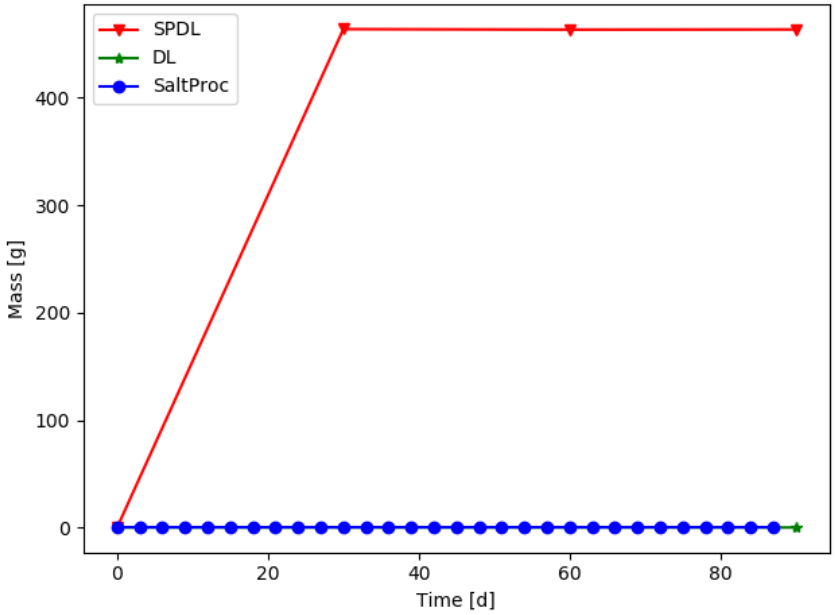


Figure 4.21: Xenon-136 mass over time comparing bulk batchwise results to continuous methods.

From these results, it can be concluded that using continuous reprocessing to perform a bulk reprocessing scheme will not yield same results. This also means that attempting to simulate a continuous reprocessing scheme using a bulk batchwise method will not yield results that are the same as a continuous reprocessing method.

4.3.2.2 Continuous and Steady Batchwise Methods

The next set of comparisons are between the steady batchwise results from SaltProc and two continuous methods, direct linear and SaltProc direct linear. The effective multiplication factor over time for each of the three methods can be seen in Figure 4.22, and it has stochastic uncertainty less than 20 pcm. It can be seen that the direct linear continuous k_{eff} is closer to the steady batchwise k_{eff} than the SaltProc direct linear result. Among figures 4.23 through 4.26, the plot which demonstrates the reason behind this is the xenon-135 mass over time.

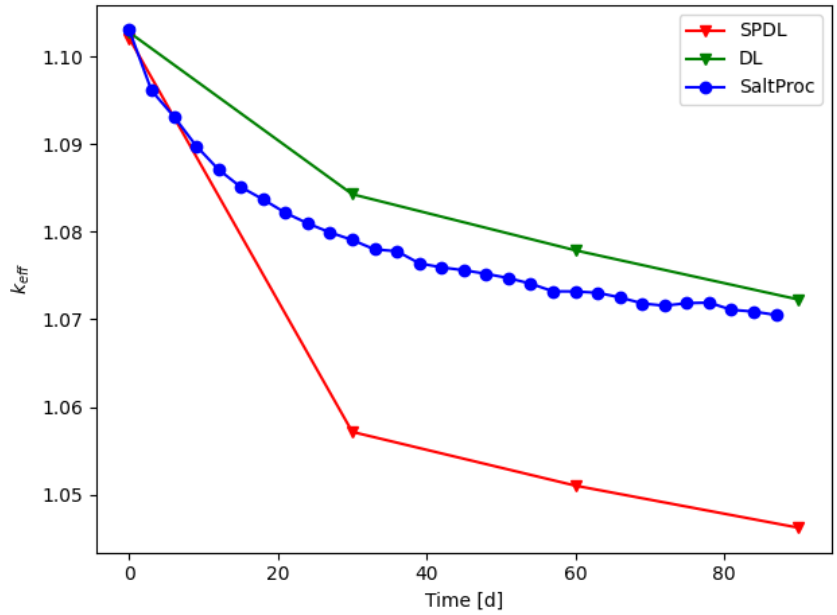


Figure 4.22: k$_{eff}$ over time comparing steady batchwise results to continuous methods.

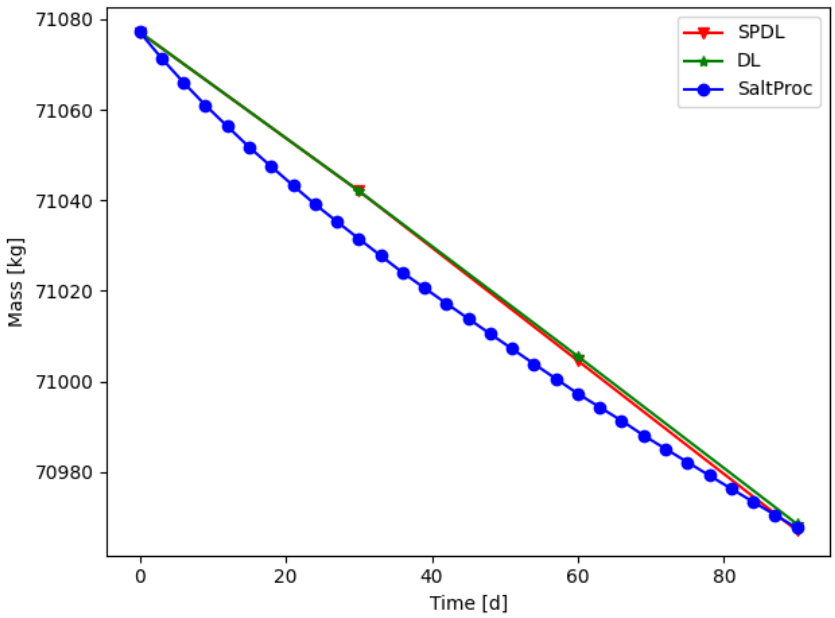


Figure 4.23: Thorium-232 mass over time comparing steady batchwise results to continuous methods.

The uranium-233 mass over time for the different methods reflects very closely with the effective multiplication factors. The uranium-233 comes from beta decay of protactinium-233, where protactinium is one of the elements designated to be removed in the MSBR reprocessing scheme. For direct linear, the removal of protactinium is approximated using 100% removal over three days. For SaltProc direct linear, the removal is instead approximated to 9.5% over 3 days, which is the value used in steady SaltProc. However, because the SPDL method is continuously removing the protactinium, this means that there is less protactinium-233 which decays into uranium-233 when compared to the steady batchwise method in SaltProc. Due to the short net depletion time, the uranium-233 feed rate plays a much larger role in dominating the uranium-233 concentration.

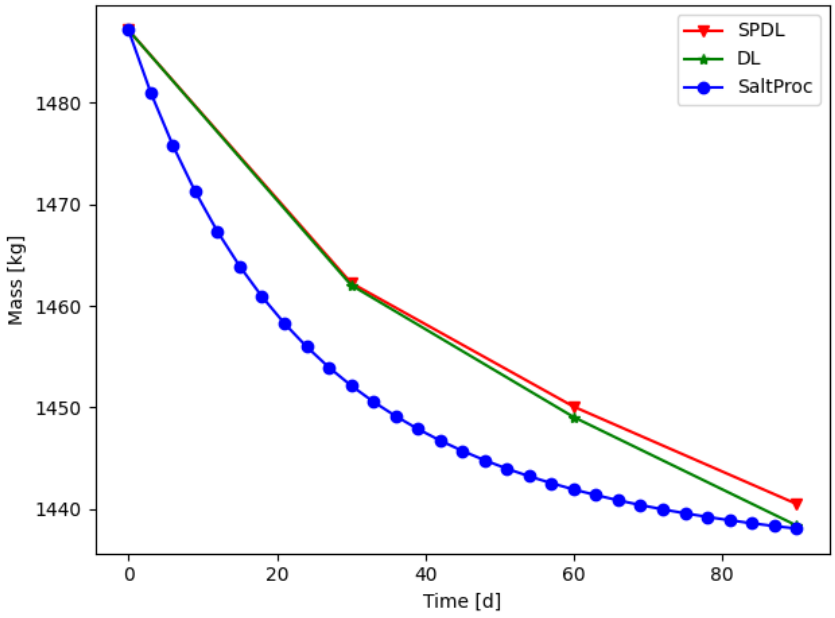


Figure 4.24: Uranium-233 mass over time comparing steady batchwise results to continuous methods.

A similar effect with the protactinium can be seen in Figures 4.25 and 4.26. Although the SPDL method is using a reprocessing constant which should allow for matching SaltProc, the continuous removal results in fairly large discrepancies. The xenon-135 difference is only on the order of tenths of a gram, but because the quantity is constantly suppressed in the continuous SPDL method, the overall capture is reduced drastically, resulting in a difference of roughly 40 grams in the xenon-136 mass.

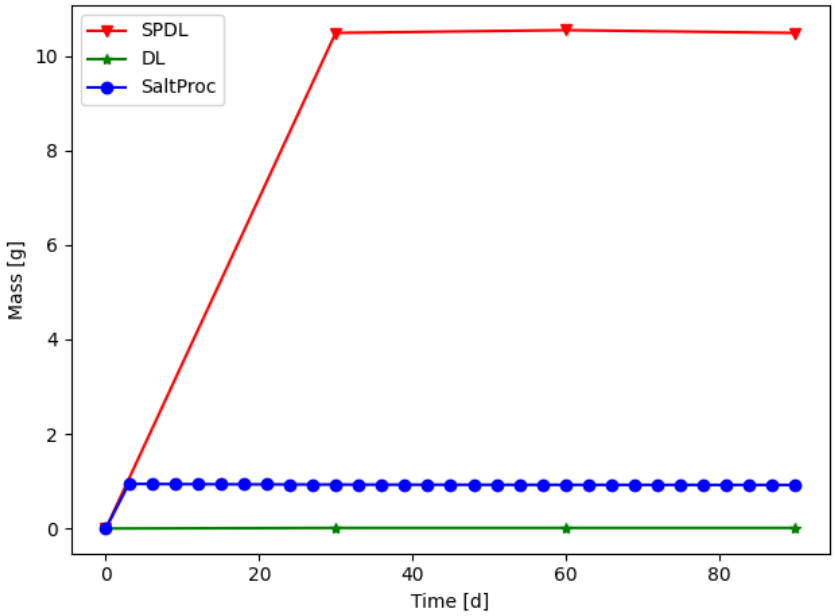


Figure 4.25: Xenon-135 mass over time comparing steady batchwise results to continuous methods.

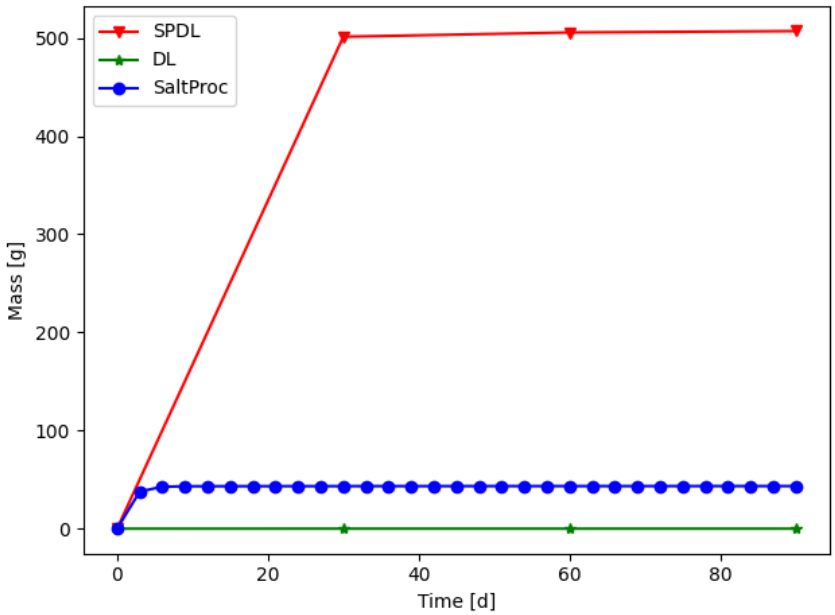


Figure 4.26: Xenon-136 mass over time comparing steady batchwise results to continuous methods.

Overall, using a continuous method to approximate a steady batchwise method does not seem generally viable. This is because isotopes which are continuously removed have a very different behaviour from isotopes which are removed after interactions occur over an entire depletion step. If the reprocessing scheme calls for a batchwise process, it may be important to reactor behaviour to allow for those capture or decay processes to occur without

influence from reprocessing.

4.3.3 Overall Differences

Figures 4.27 through 4.31 show how the steady batchwise reprocessing method with the average uranium-233 feed and the direct linear continuous method with the realistic protactinium decay compare with each other. The batchwise method uses the long term bulk average uranium-233 feed rate, and implements altered cycle rates for protactinium, xenon, and krypton. The continuous method uses only the cycle time data from the Robertson report to generate the reprocessing constants [34].

The effective multiplication factor for the two different approaches, shown in Figure 4.27, initially has the steady batchwise effective neutron multiplication factor falling at a greater rate. This is due to the initial buildup of fission product poisons, while the continuous reprocessing allows for these poisons to be removed more efficiently. However, as time goes on, the lacking uranium-233 feed due to the empty protactinium decay tank causes the current work to dip below the previous work. Over the course of three months, the longer lived behaviour cannot be seen. However, Section 4.2 covers the advanced protactinium decay model over a longer time period. That section shows the results over the course of several years and reveals that the reactivity difference due to the different refueling methods eventually decreases.

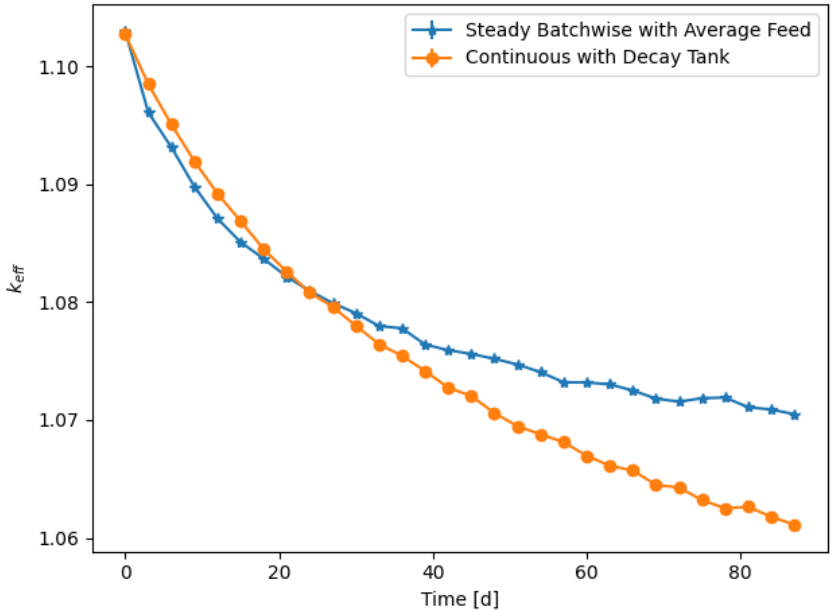


Figure 4.27: k_{eff} over time with three day depletion steps for steady batchwise and direct linear continuous reprocessing using different uranium-233 feed schemes.

The thorium-232 feed rate for both approaches is the same, yet the steady batchwise method burns it slightly

faster than the continuous method. This effect is likely caused by the presence of more uranium-233 in the previous work, which allows for a faster burn rate of the thorium-232 due to a higher net flux. The total amount of thorium-232 over time can be seen in Figure 4.28.

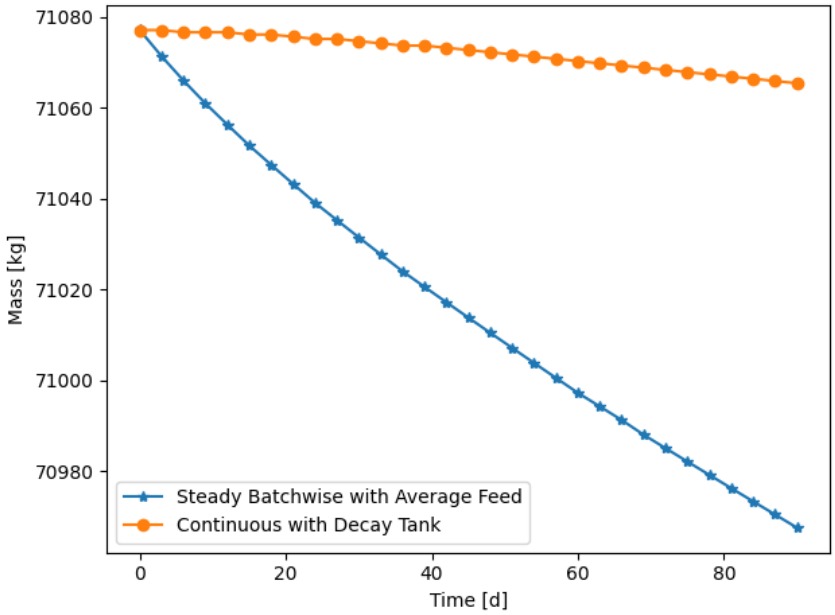


Figure 4.28: ^{232}Th over time with three day depletion steps for steady batchwise and direct linear continuous reprocessing using different uranium-233 feed schemes.

This difference in the uranium-233 masses, shown in Figure 4.29, is primarily due to the effects of the different uranium-233 feed rate approaches implemented in each of the different works.

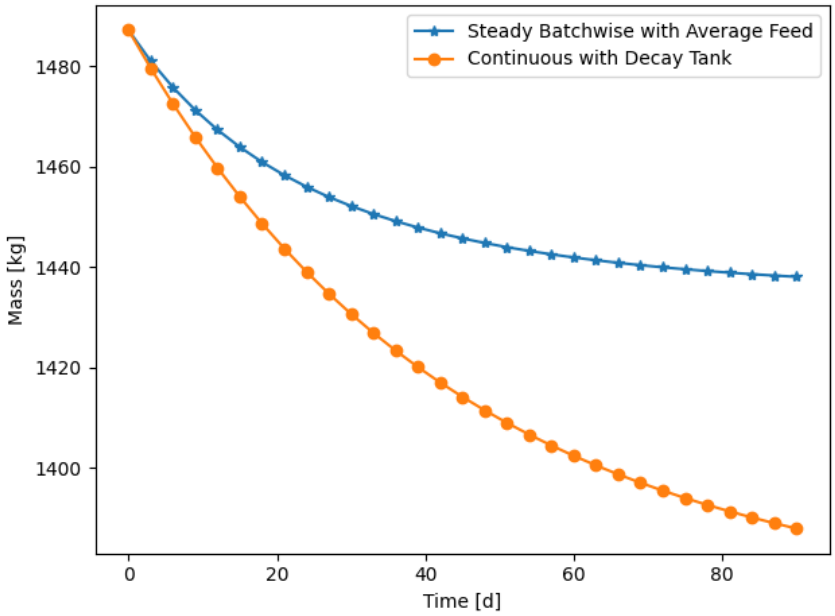


Figure 4.29: ^{233}U over time with three day depletion steps for steady batchwise and direct linear continuous reprocessing using different uranium-233 feed schemes.

For the xenon-135 and xenon-136 masses, shown in Figures 4.30 and 4.31, respectively, these results are primarily impacted due to the continuous and reprocessing methods implemented.

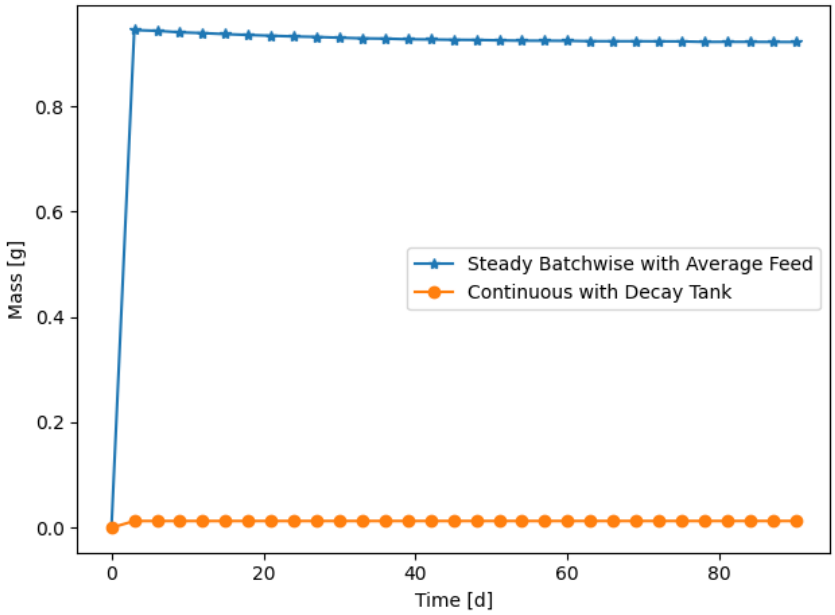


Figure 4.30: ^{135}Xe over time three day depletion steps for steady batchwise and direct linear continuous reprocessing using different uranium-233 feed schemes.

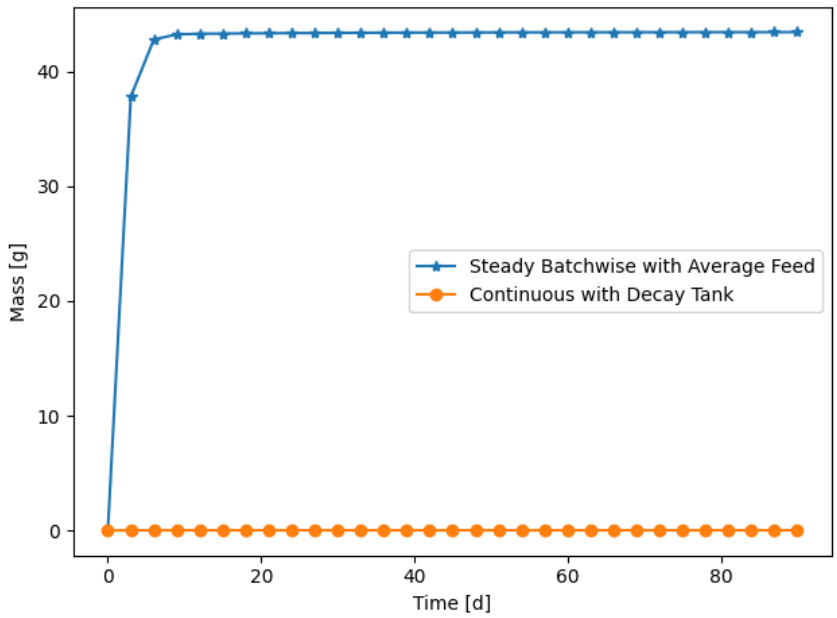


Figure 4.31: ^{136}Xe over time three day depletion steps for steady batchwise and direct linear continuous reprocessing using different uranium-233 feed schemes.

Overall, these results show the combined effects of implementing continuous reprocessing methods with a continuously updated uranium-233 refeed tank which is initially empty. This is compared with batchwise reprocessing methods and an averaged uranium-233 feed rate.

4.4 Mass Balancing

The continuous reprocessing method allows for larger depletion steps to be used without losing out on the impact of reprocessing while also allowing for fewer simulations to be run when compared to batchwise methods. However, one potential issue with this method is mass balancing.

The effects of mass balancing are of particular note in the Serpent2 code because an increase in mass causes a corresponding increase in density due to constant volumes. A simple workaround to this is to either remove some fraction of mass to return the net mass to equilibrium or adjust the volumes so the density remains constant, thus maintaining physicality.

However, if the issue of mass balancing is not considered, then the results could be fairly inaccurate. This is because changes in isotopic composition and densities will occur, where material is added to a fixed volume space. This has the potential to raise the density more than is physically possible, thus generating non-physical results.

4.4.1 Impact Minimization Using Modified Feeds

One aspect investigated is minimizing the change in the net mass by only altering the average feed rates implemented. It is determined through repeated simulations that increasing the thorium-232 feed rate drives the second derivative of the net mass negative. The uranium-233 feed increasing drives the second derivative to be positive. The reason for these effects are due to the change in the effective multiplication factor, mass of the added feed, and removal of fission products by reprocessing. Using these relationships, a process to determine the feed rates to minimize the change in the net mass is determined. The process is as follows: hold one feed rate constant while adjusting the other; if the final net mass is higher than initial net mass, then reduce the feed rates and vice versa; once the final net mass is same as initial net mass, make adjustment based on second derivative; if the second derivative is positive, increase thorium-232 or decrease uranium-233 feed; if the second derivative is negative, increase uranium-233 or decrease thorium-232 feed; iterate until second derivative is zero and net mass remains roughly constant.

After iterating through this processes several times using a depletion time step of 300 days over 10 steps, the modified average feed rates of 1.9 kg/day of thorium-232 and 2.16 kg/day of uranium-233 were determined to result in a very small change to the net mass on the order of 0.02%.

4.4.2 Impact of Mass Balancing in the MSBR

Previously in Section 3.4, I calculated the maximum impact of improper mass balancing and and showed there could be on the order of a 21% difference impact on the density of thorium-232 when only considering the feed rate and neglecting loss terms. Realistically, that percent difference in the density should be far above the maximum in this case, since loss terms are very important in determining the amount of thorium-232 present in the system. However, it is expected for the thorium-232 mass to change over time, so only viewing the mass of thorium is not very useful. Instead, viewing the net mass of the system will allow for insight into how the overall system behaviour could be expected to change due to density variations.

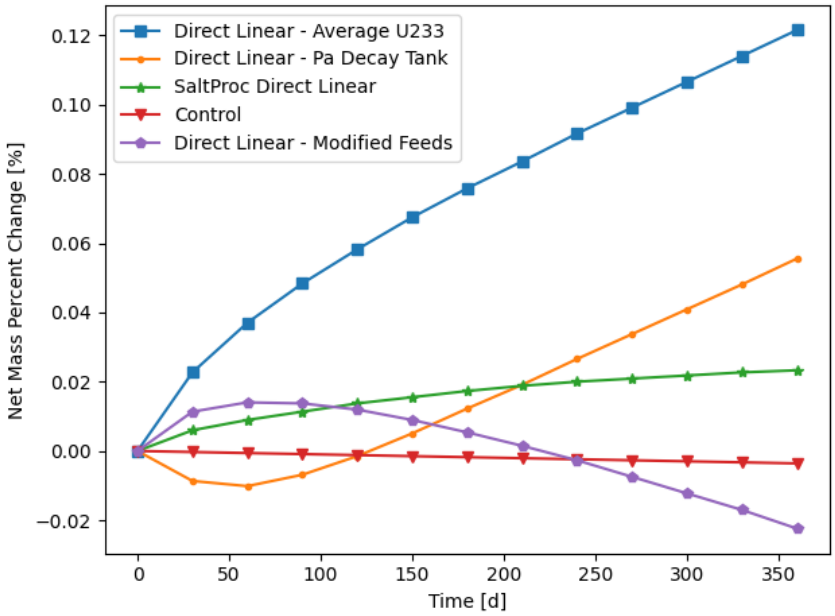


Figure 4.32: Change in net mass of the fuel salt for various methods.

Figure 4.32 shows the evolution of the net mass of the fuel salt over time while using a 30 day depletion time step over 360 days. The "Control" data points show how the net mass changes without any reprocessing feeds or removals. The method which has the largest change in the net mass is the direct linear method with the averaged uranium-233 feed rate. The net mass difference is roughly 198 kilograms, which appears to be fairly substantial. However, the approximate impact upon the simulation accuracy is dependent on the density and the total volume of the system. Equation (4.1) shows how the final density is impacted by the added mass, where the initial fuel salt density, ρ_0, is 3.35 grams per cubic centimeter.

$$\rho_f = \frac{m_0 + \Delta m}{V} = \rho_0 + \Delta\rho \tag{4.1}$$

Continuing with the direct linear case using the averaged uranium-233 feed, the change in the density is on the order of four milligrams per cubic centimeter, yielding a final density percent difference of 0.12%. This density change due to adding material to a constant volume should be kept in mind if no mass balancing measures are taken, as it will have a non-physical impact on the results.

Chapter 5

Conclusions

5.1 Comparison of Continuous and Batchwise Reprocessing Methods

In this work, I compared how continuous and batchwise reprocessing methods are used to model continuous online reprocessing by analyzing the various approaches in the MSBR. Overall, I showed that based on variable changes in step size, computational cost, and steady state relative error, the continuous reprocessing method yields better results when handling continuous online reprocessing of an MSR.

For the time step sizes, I showed mathematically how implementing larger depletion step sizes with a batchwise reprocessing method causes the relative steady state error to become larger. I also showed that this error is related to the cycle time of the elemental target, meaning that the maximum depletion time step which will yield valid results is dependent upon the reprocessing scheme of the given reactor. For continuous reprocessing, I showed that the depletion time step is instead limited by updating the simulation data and does not depend on the reprocessing scheme, allowing for greater flexibility. I also considered variable step sizes for continuous reprocessing. For the time frame of one year in the MSBR, I showed that using depletion time steps larger than 30 days led to increased error. While investigating these variable step sizes, I showed that changing the ordering of the steps did not have significant impact, but changing the largest depletion step size did directly impact the accuracy of the results.

For computational cost, I generated generic equations which showed the approximate costs of continuous and batchwise methods. The main differences in the costs came from the smaller depletion step size required of batchwise reprocessing to generate accurate results and from the doubling up of runs at the same time step. Although the doubling up of runs could be removed, the continuous reprocessing method still retains a lower computational cost due to the larger depletion step sizes it allows.

For the steady state relative error, I generated an equation under steady state assumptions. This equation I generated allowed for understanding how the difference in nuclide concentrations when using the different reprocessing methods comes about. Specifically, it perfectly isolates the differences, allowing for understanding the level of error without accounting for more complex, but realistic scenarios.

The realistic scenarios for this work were demonstrated using the MSBR. This allowed for non-steady state

analysis, where the difference in reprocessing methods have many impacts. For a given nuclide, its rate of generation from fission products may be impacted by the refueling rate, the reprocessing removal of its parent, and the change in the fission rate due to parasitic absorption which depends on the reprocessing rates. These complexities are shown within the results, which demonstrate how the batchwise and continuous reprocessing methods differ.

5.2 Continuous Reprocessing Investigations

Because the continuous reprocessing method provides many benefits when compared to the batchwise reprocessing method, investigating weaknesses in the method is important. One of these weaknesses I investigated in this work was the mass balancing of the fuel salt in the simulation. My analysis of the net mass showed that most of the methods caused an increase in mass of the MSBR after one year. Using an iterative process, I found the smallest net mass change after one year using the MSBR reprocessing scheme by adjusting the feed rates. This smallest net mass change was 0.02%, while the largest net mass change was 0.12%. This means that over the course of one year, the net mass in the MSBR has an insignificant impact on the results.

This work also included how continuous methods compared to each other and how continuous methods could approximate a batchwise method. Overall, I showed that, although there are several different methods to Decay reprocessing, they all yield similar results.

Finally, I showed how the inclusion of a decay tank allows for creation of a more physically accurate model of the MSBR. This method behaved differently from the method used by Rykhlevskii, with noticeable differences even after ten years of operation [37]. This method was then combined with the continuous reprocessing to compare with a batchwise reprocessing method that used an average feed rate instead. This led to combined differences due to the reprocessing methods and due to the refueling, which caused large differences after only 90 days of operation. This shows how the combination of different assumptions and approximations in a model can lead to differing results for an MSR model.

5.3 Future Work

There are areas that merit further investigation that will supplement this work. The first area of interest would be replication of works which have implemented batchwise reprocessing to simulate a continuous reprocessing scheme, such as those shown in Section 2.5.2. This would primarily consist of replicating the results using continuous reprocessing. Additional work would be calculating the error for different isotopes, specifically isotopes which are expected to have a large impact on reactor behavior, affected by the reprocessing scheme. This would lead nicely into an analysis of the difference in the reactor physics after depletion caused by those key isotopes.

Another useful analysis is looking into those same works, which have implemented batchwise reprocessing to simulate a continuous reprocessing scheme, and perform a depletion time step refinement study. This would be useful to demonstrate the difference in reactor physics if the batchwise reprocessing scheme implemented used depletion time steps twice as long or twice as short. Either would alter the concentrations of elements with short cycle times significantly. The concentrations could potentially vary by a factor of two, which would then lead to differences in the reactor physics. This would have an impact on the final results, and could be compared directly with the previous results of the work.

Another area of study is to replicate works which implement continuous reprocessing, such as those shown in Section 2.5.1. Particularly, investigating the development of the net mass would be of interest, as it could indicate non-physical results. Specifically, of interest is the magnitude of the change in the net mass and how it affects the densities, and thus the macroscopic cross sections. A change of a few percent in the macroscopic cross section due to a non-physical change in the mass would mean that the results presented in the work would reflect physics of a non-physical system. If the net mass change is large enough, determining feed rates which balance the mass, implementing batchwise reprocessing to balance the mass, or some other method of balancing the mass would be necessary to implement in these methods. This would allow for a comparison against the same results presented in the work with balanced mass to determine the magnitude of the effect on the reactor physics in the system.

Appendix A: Batchwise Steady State Mass Derivation

This appendix serves as a supplement to the discussion in Section 3.2.3. Here, I show the steady state mass of some arbitrary nuclide which is processed using Steady batchwise reprocessing.

The first step of this derivation is to separate the λ term into the continuous λ_c, which is the continuous losses in the Bateman equation, and the batchwise reprocessing λ_r. This is given by:

$$\lambda = \lambda_c + \lambda_r. \tag{5.1}$$

Next, set up and solve the differential equation for the continuous portions of the process. This is set as the rate of change of the mass over time is equivalent to the constant rate of gain, C, minus the continuous losses. This is given by:

$$\frac{dm}{dt} = C - \lambda_c m, \tag{5.2}$$

where

$$m(t) = \left(m_0 - \frac{C}{\lambda_c}\right) e^{-\lambda_c t} + \frac{C}{\lambda_c}. \tag{5.3}$$

Then, I modify the continuous equation to a process which updates at discrete time steps and is pre-reprocessing:

$$m_{n+1}^{pre} = \left(m_n - \frac{C}{\lambda_c}\right) e^{-\lambda_c \Delta t} + \frac{C}{\lambda_c}. \tag{5.4}$$

After each new discrete time step, I apply the batchwise reprocessing. Each time step is evaluated as a depletion time step. This is given by:

$$m_{n+1} = \left(\left(m_n - \frac{C}{\lambda_c}\right) e^{-\lambda_c \Delta t} + \frac{C}{\lambda_c}\right)(1 - \lambda_r \Delta t). \tag{5.5}$$

From here, I define some arbitrary variables for ease of use:

$$\alpha = \frac{C}{\lambda_c}, \tag{5.6}$$

$$\epsilon = e^{-\lambda_c \Delta t}, \tag{5.7}$$

$$\gamma = (1 - \lambda_r \Delta t). \tag{5.8}$$

Using these in Equation (5.5), the mass can be calculated for several subsequent steps. The more compact form of Equation (5.5) is given by:

$$m_{n+1} = ((m_n - \alpha)\epsilon + \alpha)\gamma. \tag{5.9}$$

The initial condition is defined as:

$$m(t = 0) = m_0. \tag{5.10}$$

Combining Equations (5.9) and (5.10), the first step is given by:

$$m_1 = m_0\epsilon\gamma - \alpha\epsilon\gamma + \alpha\gamma, \tag{5.11}$$

and the second step is given by:

$$m_2 = m_0\epsilon^2\gamma^2 - \alpha\epsilon^2\gamma^2 + \alpha\epsilon\gamma^2 - \alpha\epsilon\gamma + \alpha\gamma. \tag{5.12}$$

Finally, the nth step mass can be derived by following the pattern shown in the previous equations. This is given by:

$$m_n = m_0\epsilon^n\gamma^n + \alpha \sum_{j=1}^{n} \gamma^j(\epsilon^{j-1} - \epsilon^j). \tag{5.13}$$

As the time approaches infinity, the steady state mass will be reached:

$$m_{ss} = \lim_{t\to\infty} m(t). \tag{5.14}$$

Because this is an iterative process, this means n will be set to approach infinity as well to reach steady state. This is given by:

$$m_{ss} = m_0 \epsilon^{\infty} \gamma^{\infty} + \alpha \sum_{j=1}^{\infty} \gamma^j (\epsilon^{j-1} - \epsilon^j). \tag{5.15}$$

The values of γ and ϵ are between zero and one, so raising them to a large power makes them approach zero. The equation simplifies to:

$$m_{ss} = \alpha \sum_{j=1}^{\infty} \gamma^j (\epsilon^{j-1} - \epsilon^j). \tag{5.16}$$

The next step is to simplify the summation term by changing ϵ back to its original form. This is given by:

$$m_{ss} = \alpha \sum_{j=1}^{\infty} \gamma^j (e^{-\lambda_c \Delta t (j-1)} - e^{-\lambda_c \Delta t j}). \tag{5.17}$$

Next, I group the exponential terms together that have j in the exponent, given by:

$$m_{ss} = \alpha \sum_{j=1}^{\infty} \gamma^j e^{-\lambda_c \Delta t j} (e^{\lambda_c \Delta t} - 1). \tag{5.18}$$

The terms which multiply but do not contain j from the summation can be pulled outside of the summation, given by:

$$m_{ss} = \alpha (e^{\lambda_c \Delta t} - 1) \sum_{j=1}^{\infty} \gamma^j e^{-\lambda_c \Delta t j}. \tag{5.19}$$

This is of the form of a generic geometric series, which has a known convergence criteria and solution:

$$\sum_{n=1}^{\infty} x^n = \frac{x}{1-x} \ni |x| < 1 \tag{5.20}$$

This yields the equation for the steady state mass:

$$m_{ss} = \alpha (e^{\lambda_c \Delta t} - 1) \frac{\gamma^j e^{-\lambda_c \Delta t j}}{1 - \gamma^j e^{-\lambda_c \Delta t j}}. \tag{5.21}$$

Plugging in the variables α and γ yields the full equation for the steady state mass with batchwise reprocessing:

$$m_{ss} = \frac{C}{\lambda - \lambda_r} \left(e^{(\lambda - \lambda_r)\Delta t} - 1 \right) \sum_{n=1}^{\infty} \left((1 - \lambda_r \Delta t)\, e^{-(\lambda - \lambda_r)\Delta t} \right)^n. \tag{5.22}$$

References

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

[2] Andrei Rykhlevskii, Jin Whan Bae, K. H. (2018). arfc/saltproc: Code for online reprocessing simulation of molten salt reactor with external depletion solver SERPENT.

[3] Aufiero, M., Brovchenko, M., Cammi, A., Clifford, I., Geoffroy, O., Heuer, D., Laureau, A., Losa, M., Luzzi, L., Merle-Lucotte, E., Ricotti, M. E., and Rouch, H. (2014a). Calculating the effective delayed neutron fraction in the Molten Salt Fast Reactor: Analytical, deterministic and Monte Carlo approaches. *Annals of Nuclear Energy*, 65:78–90.

[4] Aufiero, M., Cammi, A., Fiorina, C., Leppänen, J., Luzzi, L., and Ricotti, M. (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(1-3):473–486.

[5] Aufiero, M., Cammi, A., Geoffroy, O., Losa, M., Luzzi, L., Ricotti, M. E., and Rouch, H. (2014b). Development of an OpenFOAM model for the Molten Salt Fast Reactor transient analysis. *Chemical Engineering Science*, 111:390–401.

[6] Betzler, B. (2021). Liquid-fueled Molten Salt Reactor Depletion Modeling. Technical report, Oak Ridge National Lab. (ORNL), Oak Ridge, TN (United States).

[7] Betzler, B. R., Powers, J. J., Brown, N. R., and Rearden, B. T. (2017a). Implementation of molten salt reactor tools in SCALE. In *M&C 2017: International Conference on Mathematics and Computational Methods Applied to Nuclear Science and Engineering*, Korea, Republic of. Korean Nuclear Society - KNS. INIS Reference Number: 53074549.

[8] Betzler, B. R., Powers, J. J., and Worrall, A. (2017b). Molten salt reactor neutronics and fuel cycle modeling and simulation with SCALE. *Annals of Nuclear Energy*, 101:489–503.

[9] Brovchenko, M., Kloosterman, J.-L., Luzzi, L., Merle, E., Heuer, D., Laureau, A., Feynberg, O., Ignatiev, V., Aufiero, M., Cammi, A., Fiorina, C., Alcaro, F., Dulla, S., Ravetto, P., Frima, L., Lathouwers, D., and Merk, B. (2019). Neutronic benchmark of the molten salt fast reactor in the frame of the EVOL and MARS collaborative projects. *EPJ Nuclear Sciences & Technologies*, 5:2.

[10] Cervi, E., Lorenzi, S., Cammi, A., and Luzzi, L. (2019). Development of a multiphysics model for the study of fuel compressibility effects in the Molten Salt Fast Reactor. *Chemical Engineering Science*, 193:379–393.

[11] Collette, A. (2013). *Python and HDF5*. O'Reilly.

[12] Cui, Y., Chen, J., Wu, J., Zou, C., Cui, L., He, F., and Cai, X. (2022). Development and verification of a three-dimensional spatial dynamics code for molten salt reactors. *Annals of Nuclear Energy*, 171:109040.

[13] Fei, T., Feng, B., and Heidet, F. (2020). Molten salt reactor core simulation with PROTEUS. *Annals of Nuclear Energy*, 140:107099.

[14] Fiorina, C., Cammi, A., Krepel, J., Mikityuk, K., and Ricotti, M. E. (2012). Preliminary Analysis of the MSFR Fuel Cycle Using Modified-EQL3D Procedure. In *Volume 4: Codes, Standards, Licensing, and Regulatory Issues; Fuel Cycle, Radioactive Waste Management and Decommissioning; Computational Fluid Dynamics (CFD) and Coupled Codes; Instrumentation and Co*, page 293, Anaheim, California, USA. ASME.

[15] Gehin, J. C. and Powers, J. J. (2016). Liquid Fuel Molten Salt Reactors for Thorium Utilization. *Nuclear Technology*, 194(2):152–161.

[16] Goluoglu, S. and McClure, J. (1998). SOFTWARE QUALIFICATION REPORT for MCNP Version 4B2.

[17] H. F. Bauman, G. W. Cunnningham III, J. L. Lucius, H. T. Kerr, and C. W. Craven, Jr. (1971). ROD: a nuclear and fuel-cycle analysis code for circulating-fuel reactors. Technical Report ORNL-TM-3359, ORNL.

[18] Hermann, O. and Westfall, R. (1984). ORIGEN-S: SCALE system module to calculate fuel depletion, actinide transmutation, fission product buildup and decay, and associated radiation source terms. Technical report, United States. NUREG/CR–0200-Vol2 INIS Reference Number: 17040291.

[19] Hombourger, B., Křepel, J., and Pautz, A. (2020). The EQL0D fuel cycle procedure and its application to the transition to equilibrium of selected molten salt reactor designs. *Annals of Nuclear Energy*, 144:107504.

[20] Ignatiev, V., Feynberg, O., Gnidoi, I., Merzlyakov, A., Smirnov, V., Surenkov, A., Tretiakov, I., Zakirov, R., Afonichkin, V., Bovet, A., Subbotin, V., Panov, A., Toropov, A., and Zherebtsov, A. (2007). Progress in Development of Li,Be,Na/F Molten Salt Actinide Recycler & Transmuter Concept. *Proceedings of ICAPP*, page 10.

[21] Jr. Vicente Valdez, P., Betzler, B., Wieselquist, W., and Fratoni, M. (2020). Modeling Molten Salt Reactor Fission Product Removal with SCALE. Technical Report ORNL/TM–2019/1418, 1608211, ORNL.

[22] Kelly, J. E. (2014). Generation IV International Forum: A decade of progress through international cooperation. *Progress in Nuclear Energy*, 77:240–246.

[23] Leppänen, J. (2007). Development of a New Monte Carlo Reactor Physics Code. *VTT Publications*, 640:241.

[24] Leppänen, J., Pusa, M., Viitanen, T., Valtavirta, V., and Kaltiaisenaho, T. (2015). The Serpent Monte Carlo code: Status, development and applications in 2013. *Annals of Nuclear Energy*, 82:142–150.

[25] Liaoyuan, H., Shaopeng, X., Jingen, C., Guimin, L., Jianhui, W., and Yang, Z. (2021). Th-U Breeding Performances in an Optimized Molten Chloride Salt Fast Reactor. *Nuclear Science and Engineering*, 195(2):185–202. Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/00295639.2020.1798728.

[26] Merle-Lucotte, E., Heuer, D., Allibert, M., Brovchenko, M., Capellan, N., and Ghetta, V. (2011). Launching the thorium fuel cycle with the Molten Salt Fast Reactor.

[27] Merle-Lucotte, E., Heuer, D., Allibert, M., Ghetta, V., Brun, C. L., Brissot, R., Liatard, E., and Mathieu, L. (2007). The thorium molten salt reactor: Launching the thorium cycle while closing the current fuel cycle. In *European Nuclear Conference*, pages 48–53, Brussels, Belgium.

[28] Nagy, K., Kloosterman, J. L., and Lathouwers, D. (2008). Parametric studies on the fuel salt composition in thermal molten salt breeder reactors. *International Conference on the Physics of Reactors*, page 8.

[29] Nuttin, A., Heuer, D., Billebaud, A., Brissot, R., Brun, C. L., Liatard, E., Meplan, O., Merle-Lucotte, E., Nifenecker, H., and Perdu, F. (2005). Potential of Thorium Molten Salt Reactors : Detailed Calculations and Concept Evolution with a View to Large Scale Energy Production. *Progress in Nuclear Energy*, 46(1):23.

[30] Park, J., Jeong, Y., Lee, H. C., and Lee, D. (2015). Whole core analysis of molten salt breeder reactor with online fuel reprocessing: Whole core analysis of MSBR with online fuel reprocessing. *International Journal of Energy Research*, pages 1673–1680.

[31] Petrie, L. M., Jordon, W., Edwards, A., Williams, P., Ryman, J., Hermann, O., Landers, N., Bucholz, J., Knight, J., Parks, C., Turner, J., Westfall, R., West, J., Emmett, M., and Greene, N. (2005). SCALE: A Modular Code System for Performing Standardized Computer Analyses for Licensing Evaluations. vols IIII. ORNL/TM-2005/39, version 5.

[32] Powers, J. J., Harrison, T. J., and Gehin, J. C. (2013). A new approach for modeling and analysis of molten salt reactors using SCALE. Technical report, American Nuclear Society - ANS; La Grange Park (United States).

[33] Ridley, G. and Chvala, O. (2017). A method for predicting fuel maintenance in once-through MSRs. *Annals of Nuclear Energy*, 110:265–281.

[34] Robertson, R. (1971). Conceptual Design Study of a Single-Fluid Molten-Salt Breeder Reactor. Technical Report ORNL–4541, 4030941, ORNL.

[35] Rodriguez-Vieitez, E., Lowenthal, M., Greenspan, E., and Ahn, J. (2002). Transmutation Capability of Once-Through Critical or Sub-Critical Molten-Salt Reactors. In *Seventh Information Exchange Meeting*, pages 161–174, Jeju, Korea.

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

[37] Rykhlevskii, A. (2018). Advanced online fuel reprocessing simulation for Thorium-fueled Molten Salt Breeder Reactor. Master's thesis, University of Illinois at Urbana-Champaign, Urbana, IL.

[38] Rykhlevskii, A. (2020). *Fuel Processing Simulation Tool for Liquid-fueled Nuclear Reactors*. Doctoral Dissertation, University of Illinois at Urbana-Champaign, Urbana, IL.

[39] Rykhlevskii, A., Bae, J. W., and Huff, K. D. (2019). Modeling and simulation of online reprocessing in the thorium-fueled molten salt breeder reactor. *Annals of Nuclear Energy*, 128:366–379.

[40] Seifert, L., Wheeler, A., and Chvala, O. (2020). Material Flows and Online Reprocessing in Serpent. In *10th Serpent User Group Meeting*, Garching, Germany.

[41] Serp, J., Allibert, M., Beneš, O., Delpech, S., Feynberg, O., Ghetta, V., Heuer, D., Holcomb, D., Ignatiev, V., Kloosterman, J. L., Luzzi, L., Merle-Lucotte, E., Uhlíř, J., Yoshioka, R., and Zhimin, D. (2014). The molten salt reactor (MSR) in generation IV: Overview and perspectives. *Progress in Nuclear Energy*, 77:308–319.

[42] Sheu, R., Chang, C., Chao, C., and Liu, Y.-W. (2013). 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*, 53:1–8.

[43] Shi, J. and Fratoni, M. (2021). Gen-foam Model and Benchmark of Delayed Neutron Precursor Drift in the Molten Salt Reactor Experiment. *EPJ Web of Conferences*, 247:06040. JC0007.

[44] Singh, V., Lish, M. R., Chvála, O., and Upadhyaya, B. R. (2017). Dynamics and control of molten-salt breeder reactor. *Nuclear Engineering and Technology*, 49(5):887–895.

[45] Singh, V., Wheeler, A. M., Upadhyaya, B. R., Chvála, O., and Greenwood, M. S. (2020). Plant-level dynamic modeling of a commercial-scale molten salt reactor system. *Nuclear Engineering and Design*, 360:110457.

[46] Wooten, D. and Powers, J. J. (2018). A Review of Molten Salt Reactor Kinetics Models. *Nuclear Science and Engineering*, 191(3):203–230.

[47] Xia, S., Chen, J., Guo, W., Cui, D., Han, J., Wu, J., and Cai, X. (2019). Development of a Molten Salt Reactor specific depletion code MODEC. *Annals of Nuclear Energy*, 124:88–97.

[48] Zhou, S., Yang, W. S., Park, T., and Wu, H. (2018). Fuel cycle analysis of molten salt reactors based on coupled neutronics and thermal-hydraulics calculations. *Annals of Nuclear Energy*, 114:369–383.

[49] Zhuang, K., Li, T., Zhang, Q., He, Q., and Zhang, T. (2020). Extended development of a Monte Carlo code OpenMC for fuel cycle simulation of molten salt reactor. *Progress in Nuclear Energy*, 118:103115.