Stability analysis of higher-order neutronics-depletion coupling schemes and Bateman operators

P. Cosgrove, E. Shwageraus

Department of Engineering, University of Cambridge, Trumpington Street, Cambridge, CB2 1PZ, United Kingdom

Abstract

Previous work has introduced the stability analysis of coupled neutronics-depletion solvers for the standard explicit Euler and predictor-corrector methods. The present work is an extension of this analysis to higher-order schemes that are commonly used, including the LE, LE/LI, LE/QI, and their implicit versions where such a method exists. Substepping, extrapolation, and linear and quadratic interpolation are investigated, and their effects on numerical stability are discussed. A realistic, numerically-stiff depletion system is considered by applying automatic differentiation to the Chebyshev rational approximation method; accounting for initial nonlinear behaviour, the predictions from the stability analysis match the outcomes of simulation.

Keywords: Depletion, Neutronics, Burn-up, Stability

1. Introduction

The simulation of nuclear reactors often entails a burn-up analysis, wherein the results from a neutron transport/diffusion calculation are used to evolve the isotopic compositions across a reactor core. The accuracy of these calculations is important for evaluating a reactor's excess reactivity, spent fuel composition, decay heat, vessel fluence, and many other quantities of interest to a reactor designer or operator.

The most standard means of coupling neutronics and depletion are either the explicit Euler or the predictor-corrector scheme. However, in the presence of strong absorbers, such as gadolinium, these often require taking impractically small time-steps to ensure accurate results, particularly when computationally expensive Monte Carlo neutron transport is used. To overcome this, new time-stepping schemes were introduced which allow reaction rates to vary during time-steps, thus achieving a higher-order and permitting longer time-steps (Isotalo and Aarnio, 2011a,b; Isotalo and Sahlberg, 2015; Josey, 2017).

Accurately treating fast-burning nuclides and changes in spectrum are not the only numerical difficulties that follow from coupling neutronics and depletion. In spatially-large

Email address: pmc55@cam.ac.uk (P. Cosgrove)

thermal reactor problems, numerical instabilities have been observed (Dufek and Hoogenboom, 2009; Kotlyar and Shwageraus, 2013; Cosgrove et al., 2020a,b). The occurrence of the instability is usually noticed as a severe asymmetry developing in a problem which should be uniform or symmetric. Although one might be inclined to attribute this phenomenon to stochastic artefacts of the Monte Carlo solver (which is used by all of the above studies), it has been demonstrated that similar behaviour can occur even in quite simple deterministic systems. This was shown by Densmore et al. (2013) for the explicit Euler coupling scheme and extended by Cosgrove et al. (2020c) for variants of the predictor-corrector scheme.

For more realistic problems which are not symmetric or uniform, these instabilities may evade the detection of a reactor analyst. Hence, there are presently two means of hopefully avoiding non-physical behaviour. The first is enforcing xenon equilibrium (Griesheimer, 2010; Isotalo et al., 2013) wherein the dynamics of xenon and iodine are removed from the depletion system. While this appears to substantially improve stability, instabilities may potentially occur due to the presence of other isotopes (Isotalo et al., 2013). The other common alternative is using an implicit depletion scheme. Although computationally expensive, this approach should, in principle, be agnostic to the cause of instability (Dufek et al., 2013; Kotlyar and Shwageraus, 2014, 2016; Josey, 2017; Valtavirta and Leppänen, 2018; Cosgrove et al., 2020a,c).

A reactor analyst would prefer to minimise the expense associated either with experimenting to find a stable time-step length for their simulations or by applying implicit methods where they are unnecessary. Ideally, as is the case in computational fluid dynamics where Courant-type instabilities can be avoided with aid from the CFL-condition, some form of stability diagnostic might be devised for burn-up calculations. Such a diagnostic might inform the time-step taken, below which instabilities cannot arise, or might be used to activate and deactivate implicit time-stepping.

The present paper hopes to bring this goal closer to fruition by extending previously performed stability analyses to dealing with more accurate methods in depletion. Namely, the paper will analyse higher-order time-stepping schemes commonly deployed in high-fidelity burn-up calculations and propose a means of treating Bateman operators which can handle realistic decay and transmutation chains. This latter extension was one of the more substantial limitations of the previous works in depletion stability analysis: both Densmore et al. (2013) and Cosgrove et al. (2020c) considered only a two-nuclide system without decay due to coarsely discretising the Bateman equations. This discretisation was necessary to obtain analytic results but fails when dealing with realistic, numerically-stiff systems of isotopes where the different transmutation and decay rates vary by several orders of magnitude (Pusa, 2013). Such a limitation is unacceptable if the analysis is to be successfully applied to realistic reactor problems.

This work begins by introducing the coupling schemes of interest before performing a Fourier stability analysis on a simplified neutronics-depletion system corresponding to each. Noticing that the resulting expressions contain the derivatives of a generic Bateman operator with respect to the neutron flux, the use of numerical differentiation is discussed. The expressions obtained are used to evaluate the stability of the coupling schemes applied to two different depletion systems: one idealised, dealt with by a coarse Bateman operator,

and one relatively realistic, with its analysis enabled by numerical differentiation.

2. Neutronics-depletion schemes

The equations of interest in burn-up should be stated before describing how they are solved numerically. Neutronics involves the solution of either the neutron transport equation or the neutron diffusion equation (Duderstadt and Hamilton, 1976). When performing burn-up calculations, these are solved in eigenvalue form, assuming that dynamical neutronic behaviour is unimportant on the time-scales of depletion. The eigenvalue form of the neutron transport equation is:

$$\begin{aligned}\boldsymbol{\Omega}\cdot\nabla\psi(\boldsymbol{r},E,\boldsymbol{\Omega})+\Sigma_{\mathrm{t}}(\boldsymbol{r},E)\psi(\boldsymbol{r},E,\boldsymbol{\Omega})=\frac{\chi(\boldsymbol{r},E)}{4\pi k}\int_0^\infty \mathrm{d}E'\nu\Sigma_{\mathrm{f}}(\boldsymbol{r},E')\int_{4\pi}\mathrm{d}\boldsymbol{\Omega}'\psi(\boldsymbol{r},E',\boldsymbol{\Omega}')\\ +\int_0^\infty \mathrm{d}E'\int_{4\pi}\mathrm{d}\boldsymbol{\Omega}'\Sigma_{\mathrm{s}}(\boldsymbol{r},E'\rightarrow E,\boldsymbol{\Omega}'\rightarrow\boldsymbol{\Omega})\psi(\boldsymbol{r},E',\boldsymbol{\Omega}')\ .\end{aligned} \tag{1}$$

Here $\boldsymbol{\Omega}$ is the neutron direction of flight, ψ is the angular neutron flux which, integrated over all angles, gives the scalar neutron flux, ϕ, $\boldsymbol{r}$ is the spatial position variable, E is the neutron energy, Σ_r is a macroscopic cross section, with $r\in\{\mathrm{t},\mathrm{a},\mathrm{f},\mathrm{s}\}$ giving the total, absorption, fission, or scattering cross sections, χ is the fission neutron energy distribution, k is the criticality eigenvalue, and ν is the average number of neutrons produced during fission. This equation will be greatly simplified for the analysis to follow in subsequent sections.

Throughout this paper, it is assumed that any neutronics solution obtained is exact, and not subject to truncation errors. This is true if either a sufficiently fine discretisation is applied to the neutronics problem or if a Monte Carlo neutron transport solver is used. Of course, Monte Carlo solvers will introduce stochastic errors instead, but their effects are not considered in the current work – in other words it is assumed that an infinite number of neutrons are simulated to obtain the solution of the transport equation, as was done in the stability analysis by Keady and Larsen (2016) when considering Monte Carlo neutronics accelerated with coarse-mesh finite-differencing.

Nuclides in a nuclear reactor, subject to a flux of neutrons, undergo transmutation, fission and spontaneous decay such that the density of a nuclide with index i, N_i, is governed by the following differential equation:

$$\begin{aligned}\frac{\partial N_i}{\partial t}=\sum_j[\langle\gamma_{ij}\sigma_{\mathrm{f},j}(E),\phi(\boldsymbol{r},E)\rangle+\langle\sigma_{\mathrm{c},ij}(E),\phi(\boldsymbol{r},E)\rangle+\lambda_{ij}\\ -\delta_{ij}\left(\lambda_j+\langle\sigma_{\mathrm{a},j}(E),\phi(\boldsymbol{r},E)\rangle\right)]N_j\ .\end{aligned} \tag{2}$$

Here t is the time variable, the angular bracket notation denotes a volume-averaged integration over space and neutron energy, γ_{ij} is the fission yield of nuclide i from the fission of nuclide j, $\sigma_{\mathrm{f},j}$ is the microscopic fission cross section of nuclide j, ϕ is the scalar flux, $\sigma_{\mathrm{c},ij}$ is

the microscopic capture cross section of nuclide j which transmutes directly to nuclide i, λ_{ij} is the decay constant of nuclide j for the decay channel leading directly to nuclide i, δ_{ij} is the Kronecker delta, λ_j is the decay constant of nuclide j for all decay channels, $\sigma_{\mathrm{a},j}$ is the microscopic absorption cross section of nuclide j, and N_j is the nuclide density of nuclide j. More succinctly, for a vector of nuclides transmuting, fissioning, and decaying into each other, this equation (known as the Bateman equation) can be written for a vector of nuclide densities as (Bell and Glasstone, 1970):

$$\frac{\partial N}{\partial t} = \mathbf{A}N \ . \tag{3}$$

N is now the nuclide density vector and $\mathbf{A}$ is the burn-up matrix which describes all pathways for fission, transmutation and decay from one nuclide to another. If the neutronics is mono-energetic (as is the case in the analysis to follow), the integration over energy is no longer performed in Eq. (2). Thus, the burn-up matrix can be written simply as:

$$\mathbf{A} = \mathbf{\Sigma}\phi + \mathbf{\Lambda} \ . \tag{4}$$

Here $\mathbf{\Sigma}$ is the matrix containing the cross sections and fission yields governing transmutation and fission from one nuclide to another, ϕ is the spatially-averaged local scalar neutron flux at time t, and $\mathbf{\Lambda}$ is the matrix of nuclide decay constants governing the rates at which nuclides are lost and gained through decay.

During burn-up calculations, the Bateman equation is numerically solved for every nuclide density across the spatial domain – for reactor calculations this is often only solved in the fuel, although it may also be done in structural materials, control rods, and the coolant as well. To deplete separate nuclide densities, a burn-up discretisation is usually imposed on the problem, dividing space into regions of uniform nuclide density which can then be depleted independently of each other. As with neutronics, this paper will treat the burn-up discretisation as sufficiently fine that it may be neglected. That said, its effects are considered in a separate publication (Cosgrove and Adamowicz, 2021).

Solving Eq. (3) numerically also necessitates a time discretisation, given that $\mathbf{A}$ is a function of the neutron flux solution, ϕ, which evolves during burn-up through changes in neutron energy spectrum and local reactions rates. As a result, as well as evaluating corresponding neutronics solutions, all burn-up time-stepping schemes involve performing one or more operations of the following form each time-step:

$$N_{n+1} = \mathbf{B}N_n \ . \tag{5}$$

Here N_{n+1} is the nuclide density vector at time-point $n+1$, $\mathbf{B}$ is a Bateman operator, and N_n is the nuclide density vector at time-point n. The term 'Bateman operator' is non-standard, but it simply refers to some operator which evolves the nuclide density vector from one time-point to the next. In the most naïve case this might simply be the matrix resulting from an explicit time discretisation of Eq. (3) ($\mathbf{B} = \mathbf{I} + \Delta t\mathbf{A}$, where $\mathbf{I}$ is the identity matrix and Δt is the time-step length). However, in practical, high-fidelity calculations, this treatment fails

to accurately resolve the often numerically stiff system described by $\mathbf{A}$. Therefore, practical Bateman operators tend to take the form of an ODE solver (Griesheimer et al., 2017), a matrix exponential (Pusa and Leppänen, 2010), or a semi-analytic treatment of the depletion equations where some approximations are made to the structure of the burn-up matrix (Cetnar, 2006). For fully explicit burn-up algorithms, only a single $\mathbf{B}$ is required, which will depend on the time-step length, Δt, and the neutronics solution at time-point n, ϕ_n. That is, for fully explicit methods, there will be a single Bateman operator given by $\mathbf{B} = \mathbf{B}(\Delta t, \phi_n)$. For predictor-corrector or implicit burn-up algorithms, $\mathbf{B}$ will also depend on the neutronics solution at time-point $n+1$, ϕ_{n+1}. Hence, $\mathbf{B} = \mathbf{B}(\Delta t, \phi_n, \phi_{n+1})$. It should also be added that explicit, predictor-corrector, and implicit methods can also use information from previous time-points (usually only that immediately previous in common algorithms), such that there may also be a dependence on ϕ_{n-1}. This is discussed further in due course. In the standard predictor-corrector methods which are used in burn-up, two different evaluations of Eq. (5) are performed: one using a Bateman operator with only a dependence on the n-th flux solution for the predictor step, and one using a Bateman operator with dependence on the $(n+1)$-th flux solution as well for the corrector step, requiring two flux solutions, at time-points n and $n+1$. Meanwhile, for common implicit methods, several evaluations of Eq. (5) will take place, depending on the specified number of iterations, again requiring as many neutronics solutions as Bateman equation evaluations. Aside from the first evaluation, the Bateman operators in implicit methods will all have a dependence on the $(n+1)$-th flux solution. This will also be elaborated on further throughout the paper.

The matrix exponential Bateman operator is arguably the most common in current reactor physics tools. This arises because, for a constant burn-up matrix evaluated at time-point n, $\mathbf{A}_n = \mathbf{A}(\phi_n)$, Eq. (3) has the solution (Bell and Glasstone, 1970):

$$N_{n+1} = \exp\left[\mathbf{A}_n \Delta t\right] N_n \ . \tag{6}$$

Here exp[·] is the matrix exponential (Moler and Van Loan, 2003). This equation can be efficiently and accurately solved for N_{n+1} for realistic burn-up matrices using the Chebyshev rational approximation method (CRAM) (Pusa and Leppänen, 2010; Pusa, 2011, 2016).

The use of Eq. (6) alongside a neutronics solution to evaluate the burn-up matrix at time-point n is known as the explicit Euler method (not to be confused with an explicit Euler time discretisation of Eq. (3)). Here only a single Bateman operator is used which is simply:

$$\mathbf{B}(\Delta t, \phi_n) = \exp\left[\mathbf{A}_n \Delta t\right] . \tag{7}$$

Bateman operators which use a single, constant burn-up matrix evaluated at time-point n are known as Constant Extrapolation (or CE) Bateman operators.

While relatively computationally efficient – requiring only a single neutronics solution per time-step – unfortunately, the explicit Euler method is only accurate for relatively short time-steps. Accuracy further deteriorates in the presence of strong burnable absorbers.

Overcoming this limit on time-step lengths has inspired the development of 'substep' schemes, wherein the burn-up matrix is permitted to evolve over the course of a time-step

by splitting the time-step into several substeps and extrapolating/interpolating between two or more previously evaluated burn-up matrices (Isotalo and Aarnio, 2011a,b). The simplest of these schemes is known as the LE scheme (standing for linear extrapolation)[1]. Having evaluated the burn-up matrix at the current time-point and that immediately previous, one can perform a Linear Extrapolation. This is shown in Algorithm 1. In the algorithm as written, N_0 is the initial nuclide density vector, n_{steps} is the number of time-steps, S is the number of substeps per time-step, $w_n^{(s)}$ and $w_{n-1}^{(s)}$ are the extrapolation weights during a given substep, s, and $\phi(N_n)$ denotes the calculation of a neutronics solution given a nuclide density vector field. The extrapolation weights shown here are given by:

$$w_n^{(s)} = 1 + \frac{s}{S}\frac{\Delta t_n}{\Delta t_{n-1}} \ , \tag{8}$$

$$w_{n-1}^{(s)} = -\frac{s}{S}\frac{\Delta t_n}{\Delta t_{n-1}} \ , \tag{9}$$

with Δt_n the time-step between the n-th and $(n+1)$-th time-points, and Δt_{n-1} the time-step between the $(n-1)$-th and n-th time-points. The possible difference in time-steps is included here for generality, but will not be explicitly considered throughout the remainder of the paper, with all time-steps presumed identical.

Note that this scheme is not self-starting in that it must be initialised by a simpler method, namely the explicit Euler scheme. Taking only a single substep, the LE scheme reduces to the explicit Euler scheme. The basic difference between the LE and explicit Euler scheme is the numerical solution of the Bateman equation, which, for the LE scheme, is given by:

$$N_{n+1} = \prod_{s=0}^{S-1} \exp\left[\left(w_n^{(s)}\mathbf{A}_n + w_{n-1}^{(s)}\mathbf{A}_{n-1}\right)\frac{\Delta t}{S}\right] N_n \ , \tag{10}$$

where the product is defined in this paper such that, as s increases, subsequent matrix exponentials left-multiply the previous. Therefore the Bateman operator for the LE scheme is:

$$\mathbf{B}_S^{\text{LE}}(\Delta t, \phi_n, \phi_{n-1}) = \prod_{s=0}^{S-1} \exp\left[\left(w_n^{(s)}\mathbf{A}_n + w_{n-1}^{(s)}\mathbf{A}_{n-1}\right)\frac{\Delta t}{S}\right] \ . \tag{11}$$

The LE superscript is used henceforth to highlight that an LE Bateman operator is being used with the subscript S emphasising the number of substeps. If $S = 1$, it becomes a CE Bateman operator.

The LE scheme is advantageous in that it is relatively accurate while requiring only a single transport solution per time-step, albeit with an increased memory cost compared to explicit Euler due to storing the burn-up matrix from the previous time-step. However, it

[1]Although not considered here, it has also been shown that performing substepping even for a constant burn-up matrix can improve the accuracy of CRAM matrix exponential evaluations (Isotalo and Pusa, 2016)

has been reported to be apparently less stable than other methods, even while enforcing xenon equilibrium (Isotalo et al., 2013).

Input: $N_0, n_{\text{steps}}, \Delta t, S$
for $n = 0, ..., n_{\text{steps}} - 1$ **do**
  $\mathbf{A}_n \leftarrow \phi(N_n)$
  if $n > 0$ **then**
    $N \leftarrow N_n$
    for $s = 0, ..., S - 1$ **do**
      $\mathbf{A} \leftarrow w_n^{(s)} \mathbf{A}_n + w_{n-1}^{(s)} \mathbf{A}_{n-1}$
      $N \leftarrow \exp\left[\mathbf{A} \frac{\Delta t}{S}\right] N$
    end
    $N_{n+1} \leftarrow N$
  else
    $N_{n+1} \leftarrow \exp\left[\mathbf{A}_n \Delta t\right] N_n$
  end
end

Algorithm 1: LE substep method.

The simplest predictor-corrector-type substep method is the CE/LI scheme (standing for Constant Extrapolation/Linear Interpolation). Using the CE/LI, the predictor step is a standard explicit Euler extrapolation without any variation in the burn-up matrix across the time-step. The corrector step is a linear interpolation between the known beginning-of-step (time-point n) burn-up matrix and the predicted end-of-step (time-point $n + 1$) burn-up matrix, updating the weighting of each matrix as the substeps proceed. This is shown in Algorithm 2. The weights shown here now correspond to interpolation, rather than extrapolation previously. These are given by:

$$w_n^{(s)} = 1 - \frac{s + \frac{1}{2}}{S} \ , \tag{12}$$

$$w_{n+1}^{(s)} = \frac{s + \frac{1}{2}}{S} \ . \tag{13}$$

Taking only a single substep, this becomes one of the standard predictor-corrector schemes with an equal weighting between the initial and final burn-up matrices.

As with all predictor-corrector schemes, the CE/LI requires two neutronics solutions per time-step. First, ϕ_n is evaluated given the existence of N_n, followed by predictor depletion given by Eq. (6):

$$N_{n+1} = \exp\left[\mathbf{A}_n \Delta t\right] N_n \ ,$$

followed by the evaluation of a neutronics solution at time-point $n + 1$, ϕ_{n+1}, and ending with the corrector depletion:

$$N_{n+1} = \prod_{s=0}^{S-1} \exp\left[\left(w_n^{(s)}\mathbf{A}_n + w_{n+1}^{(s)}\mathbf{A}_{n+1}\right)\frac{\Delta t}{S}\right] N_n \ . \tag{14}$$

Hence, the predictor and corrector Bateman operators are, respectively:

$$\mathbf{B}_1^{\mathrm{LE}}(\Delta t, \phi_n) = \exp\left[\mathbf{A}_n \Delta t\right] \ , \tag{15}$$

and:

$$\mathbf{B}_S^{\mathrm{LI}}(\Delta t, \phi_n, \phi_{n+1}) = \prod_{s=0}^{S-1} \exp\left[\left(w_n^{(s)}\mathbf{A}_n + w_{n+1}^{(s)}\mathbf{A}_{n+1}\right)\frac{\Delta t}{S}\right] \ . \tag{16}$$

As with the LE Bateman operator, the superscript LI emphasises that an LI Bateman operator is used.

Input: $N_0, n_{\mathrm{steps}}, \Delta t, S$
for $n = 0, ..., n_{\mathrm{steps}} - 1$ **do**
  $\mathbf{A}_n \leftarrow \phi(N_n)$
  $N_{n+1} \leftarrow \exp\left[\mathbf{A}_n \Delta t\right] N_n$
  $\mathbf{A}_{n+1} \leftarrow \phi(N_{n+1})$
  $N \leftarrow N_n$
  for $s = 0, ..., S - 1$ **do**
    $\mathbf{A} \leftarrow w_n^{(s)}\mathbf{A}_n + w_{n+1}^{(s)}\mathbf{A}_{n+1}$
    $N \leftarrow \exp\left[\mathbf{A}\frac{\Delta t}{S}\right] N$
  end
  $N_{n+1} \leftarrow N$
end

Algorithm 2: CE/LI substep method.

The next permutation of these methods which is commonly applied is the LE/LI scheme – it is simply a combination of Algorithms 1 and 2 with linear extrapolation and interpolation on the predictor and corrector, respectively. It is shown in Algorithm 3. Here the former number of substeps, S, is replaced by the number of substeps on the predictor, S_p, and the number of substeps on the corrector, S_c, as these can potentially differ in the general case although, to the authors' knowledge, there is no previously stated significant reason for them to do so. Therefore, the predictor and corrector Bateman operators are, respectively:

$$\mathbf{B}_{S_p}^{\mathrm{LE}}(\Delta t, \phi_n, \phi_{n-1}) = \prod_{s=0}^{S_p-1} \exp\left[\left(w_n^{(s)}\mathbf{A}_n + w_{n-1}^{(s)}\mathbf{A}_{n-1}\right)\frac{\Delta t}{S_p}\right] \ , \tag{17}$$

and:

$$\mathbf{B}_{S_c}^{\mathrm{LI}}(\Delta t, \phi_n, \phi_{n+1}) = \prod_{s=0}^{S_c-1} \exp\left[\left(w_n^{(s)}\mathbf{A}_n + w_{n+1}^{(s)}\mathbf{A}_{n+1}\right)\frac{\Delta t}{S_c}\right] \ . \tag{18}$$

Input: $N_0, n_{\text{steps}}, \Delta t, S_p, S_c$
for $n = 0, ..., n_{\text{steps}} - 1$ **do**
  $\mathbf{A}_n \leftarrow \phi(N_n)$
  if $n > 0$ **then**
    $N \leftarrow N_n$
    for $s = 0, ..., S + p - 1$ **do**
      $\mathbf{A} \leftarrow w_n^{(s)} \mathbf{A}_n + w_{n-1}^{(s)} \mathbf{A}_{n-1}$
      $N \leftarrow \exp\left[\mathbf{A} \frac{\Delta t}{S_p}\right] N$
    end
    $N_{n+1} \leftarrow N$
  else
    $N_{n+1} \leftarrow \exp\left[\mathbf{A}_n \Delta t\right] N_n$
  end
  $\mathbf{A}_{n+1} \leftarrow \phi(N_{n+1})$
  $N \leftarrow N_n$
  for $s = 0, ..., S_c - 1$ **do**
    $\mathbf{A} \leftarrow w_n^{(s)} \mathbf{A}_n + w_{n+1}^{(s)} \mathbf{A}_{n+1}$
    $N \leftarrow \exp\left[\mathbf{A} \frac{\Delta t}{S_c}\right] N$
  end
  $N_{n+1} \leftarrow N$
end

Algorithm 3: LE/LI substep method.

Finally, there is the LE/QI scheme – the QI stands for Quadratic Interpolation, performed using the burn-up matrix from the previous time-point, as well as those at the beginning-(n) and end-of-step ($n + 1$). This is shown in Algorithm 4. The quadratic interpolation weights are (again, allowing the time-step lengths to vary for the sake of generality):

$$w_n^{(s)} = \left(1 + \frac{\Delta t_n}{\Delta t_{n-1}} \frac{s + \frac{1}{2}}{S}\right) \left(1 - \frac{s + \frac{1}{2}}{S}\right) , \tag{19}$$

$$w_{n+1}^{(s)} = \left(\frac{s + \frac{1}{2}}{S}\right) \left(\frac{1 + \frac{\Delta t_n}{\Delta t_{n-1}} \frac{s+\frac{1}{2}}{S}}{1 + \frac{\Delta t_n}{\Delta t_{n-1}}}\right) , \tag{20}$$

$$w_{n-1}^{(s)} = -\left(\frac{\Delta t_n}{\Delta t_{n-1}}\right)^2 \left(\frac{s + \frac{1}{2}}{S}\right) \left(\frac{1 - \frac{s+\frac{1}{2}}{S}}{1 + \frac{\Delta t_n}{\Delta t_{n-1}}}\right) . \tag{21}$$

The numerical solution of the Bateman equation using quadratic interpolation on the corrector is:

$$N_{n+1} = \prod_{s=0}^{S_c - 1} \exp\left[\left(w_n^{(s)} \mathbf{A}_n + w_{n+1}^{(s)} \mathbf{A}_{n+1} + w_{n-1}^{(s)} \mathbf{A}_{n-1}\right) \frac{\Delta t}{S_c}\right] N_n . \tag{22}$$

Hence, the Bateman operators used by the LE/QI scheme are:

$$\mathbf{B}_{S_p}^{\mathrm{LE}}(\Delta t, \phi_n, \phi_{n-1}) = \prod_{s=0}^{S_p-1} \exp\left[\left(w_n^{(s)}\mathbf{A}_n + w_{n-1}^{(s)}\mathbf{A}_{n-1}\right)\frac{\Delta t}{S_p}\right] , \tag{23}$$

and:

$$\mathbf{B}_{S_c}^{\mathrm{QI}}(\Delta t, \phi_n, \phi_{n+1}, \phi_{n-1}) = \prod_{s=0}^{S_c-1} \exp\left[\left(w_n^{(s)}\mathbf{A}_n + w_{n+1}^{(s)}\mathbf{A}_{n+1} + w_{n-1}^{(s)}\mathbf{A}_{n-1}\right)\frac{\Delta t}{S_c}\right] . \tag{24}$$

One practical difficulty with the LE/QI scheme is that it cannot be applied in cases where the system changes discontinuously across the two time-steps considered during the corrector, e.g., where there has been a control rod movement in between. The scheme also needs to hold three sets of burn-up matrices in memory during the corrector step – this may be a substantial memory burden when depleting large, finely-discretised reactor cores (Griesheimer et al., 2017).

For each of the predictor-corrector methods described, one may formulate an implicit version where the corrector step is iterated alongside a relaxation on the nuclide density vector (or the flux solution, although that variant is not discussed here). Applied to the CE/LI scheme, this is shown in Algorithm 5, but the procedure is analogous for LE/LI and LE/QI. Here $0 < \alpha < 1$ is a relaxation factor, and J is the number of corrector iterations to perform. Most commonly this has been implemented using the stochastic approximation (Robbins and Monro, 1951) where α is iteration-dependent and given by:

$$\alpha_j = \frac{1}{1+j} . \tag{25}$$

Here j is the iteration index, beginning at 0. This has been applied to most of the previously discussed depletion schemes (Dufek et al., 2013; Kotlyar and Shwageraus, 2014, 2016; Josey, 2017; Valtavirta and Leppänen, 2018). Alternatively, one can apply a fixed relaxation factor which, if well chosen, appears to be more efficient in achieving a stable solution (Cosgrove et al., 2020a,c). These implicit schemes are (with the exception of the first-order accurate SIE which is not discussed here) identical to predictor-corrector schemes, except that the corrector step is iterated and a relaxation is applied to the nuclide density vector. Any appropriate choice of predictor and corrector Bateman operator can be made. For the j-th iteration of the corrector step, the Implicit CE/LI substep method will obtain a flux solution $\phi_{n+1}^{(j)}$ from the nuclide density vector $N_{n+1}^{(j)}$ and update the nuclide density vector as:

$$N_{n+1}^{(j+1)} = \alpha \prod_{s=0}^{S-1} \exp\left[\left(w_n^{(s)}\mathbf{A}_n + w_{n+1}^{(s)}\mathbf{A}_{n+1}^{(j)}\right)\frac{\Delta t}{S}\right] N_n + (1-\alpha)N_{n+1}^{(j)} . \tag{26}$$

In terms of generic Bateman operators, the depletion update during the j-th corrector step of these implicit methods are described by a slightly different equation than Eq. (5), namely:

Input: $N_0, n_{\text{steps}}, \Delta t, S_p, S_c$
for $n = 0, ..., n_{\text{steps}} - 1$ **do**
  $\mathbf{A}_n \leftarrow \phi(N_n)$
  if $n > 0$ **then**
    $N \leftarrow N_n$
    for $s = 0, ..., S_p - 1$ **do**
      $\mathbf{A} \leftarrow w_n^{(s)} \mathbf{A}_n + w_{n-1}^{(s)} \mathbf{A}_{n-1}$
      $N \leftarrow \exp\left[\mathbf{A} \frac{\Delta t}{S_p}\right] N$
    end
    $N_{n+1} \leftarrow N$
  else
    $N_{n+1} \leftarrow \exp\left[\mathbf{A}_n \Delta t\right] N_n$
  end
  $\mathbf{A}_{n+1} \leftarrow \phi(N_{n+1})$
  $N \leftarrow N_n$
  if $n > 0$ **then**
    for $s = 0, ..., S_c - 1$ **do**
      $\mathbf{A} \leftarrow w_n^{(s)} \mathbf{A}_n + w_{n+1}^{(s)} \mathbf{A}_{n+1} + w_{n-1}^{(s)} \mathbf{A}_{n-1}$
      $N \leftarrow \exp\left[\mathbf{A} \frac{\Delta t}{S_c}\right] N$
    end
  else
    for $s = 0, ..., S_c - 1$ **do**
      $\mathbf{A} \leftarrow w_n^{(s)} \mathbf{A}_n + w_{n+1}^{(s)} \mathbf{A}_{n+1}$
      $N \leftarrow \exp\left[\mathbf{A} \frac{\Delta t}{S_c}\right] N$
    end
  end
  $N_{n+1} \leftarrow N$
end

Algorithm 4: LE/QI substep method.

$$N_{n+1}^{(j+1)} = \alpha \mathbf{B}(\phi_n, \phi_{n+1}^{(j)}) N_n + (1-\alpha) N_{n+1}^{(j)} \, . \tag{27}$$

In principle, one could conceive of other implicit schemes, e.g., where a Newton solve is performed to converge on the nuclide density vector at time-point $n+1$. However, this has not been pursued (especially in the Monte Carlo context) due to the computational cost of obtaining a Jacobian matrix and numerical difficulties associated with Jacobian-free methods.

Fully implicit algorithms across computational physics often iterate until some convergence criterion is met – although this might be done for implicit burn-up algorithms, it is not currently done (to the authors' knowledge) in popular burn-up codes. Both Serpent (Leppänen et al., 2015) and OpenMC (Romano et al., 2015, 2021) implement the stochastic implicit scheme to perform a specified number of iterations, rather than to meet a convergence criterion. There are instances in the literature where stochastic implicit schemes are used alongside a convergence criterion, although seemingly only when coupling with additional physics, like thermal-hydraulics (Kotlyar and Shwageraus, 2014; Tuominen, 2015). Throughout this paper, where implicit methods feature, they are performed for a specified number of iterations instead of following a convergence criterion.

Input: $N_0, n_{\text{steps}}, \Delta t, S, J, \alpha$
for $n = 0, ..., n_{\text{steps}} - 1$ **do**
  $\mathbf{A}_n \leftarrow \phi(N_n)$
  $N_{n+1}^{(0)} \leftarrow \exp\left[\mathbf{A}_n \Delta t\right] N_n$
  for $j = 0, ..., J-1$ **do**
    $\mathbf{A}_{n+1} \leftarrow \phi(N_{n+1}^{(j)})$
    $N \leftarrow N_n$
    for $s = 0, ..., S-1$ **do**
      $\mathbf{A} \leftarrow w_n^{(s)} \mathbf{A}_n + w_{n+1}^{(s)} \mathbf{A}_{n+1}$
      $N \leftarrow \exp\left[\mathbf{A} \frac{\Delta t}{S}\right] N$
    end
    $N_{n+1}^{(j+1)} \leftarrow \alpha N + (1-\alpha) N_{n+1}^{(j)}$
  end
  $N_{n+1} \leftarrow N_{n+1}^{(J)}$
end

Algorithm 5: Implicit CE/LI substep method.

While discussing methods to stabilise burn-up simulations, it would be remiss not to also highlight enforcing xenon equilibrium, mentioned previously in this paper. Enforcing xenon equilibrium means numerically converging the neutronics solution simultaneously with the xenon-135 and iodine-135 (xenon's main decay precursor) densities across the spatial domain – note that xenon strongly affects the absorption cross section in a given system while the amount of xenon and iodine evolved in a region is determined by the magnitude of the local

fission rate. At equilibrium, the local xenon and iodine densities are given by:

$$N_{\mathrm{Xe}} = \frac{(\gamma_{\mathrm{Xe}} + \gamma_{\mathrm{I}})\Sigma_{\mathrm{f}}\phi}{\lambda_{\mathrm{Xe}} + \sigma_{\mathrm{a,Xe}}\phi} \,, \tag{28a}$$

$$N_{\mathrm{I}} = \frac{\gamma_{\mathrm{I}}\Sigma_{\mathrm{f}}\phi}{\lambda_{\mathrm{I}}} \,. \tag{28b}$$

Here N_{Xe} and N_{I} correspond to the atomic densities of xenon and iodine, respectively, γ_{Xe} and γ_{I} are the corresponding average yields of the isotope from fission, λ_{Xe} and λ_{I} are the corresponding decay constants, Σ_{f} is the local macroscopic fission cross section, ϕ is the local scalar flux (obtained from solving the transport or diffusion equation, or from Monte Carlo), and $\sigma_{\mathrm{a,Xe}}$ is the microscopic absorption cross section of xenon.

Enforcing xenon equilibrium can be done either by fixed-point iteration between neutronics and the density equations or by periodically updating the xenon and iodine densities while neutronics converges during source iteration (Griesheimer, 2010). By enforcing xenon equilibrium and simultaneously removing xenon and iodine from the Bateman equations solved during burn-up, Isotalo et al. (2013) found that burn-up instabilities become substantially less prominent. This is not entirely surprising given that xenon can cause stability problems in physical reactors, not just simulated ones. That said, as mentioned above, Isotalo et al. (2013) describes the emergence of instabilities when taking larger steps for particular burn-up algorithms and few further assurances exist regarding stability even while enforcing xenon equilibrium.

3. Analysis

Each of the algorithms previously defined will be analysed here, based on the paradigm introduced by Densmore et al. (2013) and developed by Cosgrove et al. (2020c).

To perform the analysis, the neutronics system is assumed to be well-represented by a 1D slab-geometry, mono-energetic diffusion equation with isotropic scattering in eigenvalue form. This can be justified given numerical instabilities tend to be observed in essentially 1D, axially homogeneous PWR problems where neutrons are thermalised, have short mean-free paths, and the diffusion approximation is reasonable. Furthermore, the most problematic frequencies where burn-up instabilities are concerned tend to be low – in these cases diffusion and transport agree well. Finally, work has been performed to extend previous stability analyses to handle transport instead of diffusion as the neutronics treatment: the final differences in both the equations obtained and the numerical results are small for problems and frequencies of interest (Adamowicz and Cosgrove, 2021). However, this neglects the effect of a changing neutron energy spectrum during burn-up, changing the one-group cross sections which partly constitute the burn-up matrix. The diffusion equation is given as:

$$-\frac{\partial}{\partial x}\left(\frac{1}{3\Sigma_{\mathrm{t}}}\frac{\partial\phi}{\partial x}\right) + \Sigma_{\mathrm{a}}\phi = \frac{1}{k}\nu\Sigma_{\mathrm{f}}\phi \,. \tag{29}$$

Here ϕ is the scalar neutron flux, x is the spatial variable, Σ_{t} is the total macroscopic cross section, Σ_{a} is the macroscopic absorption cross section, k is the criticality eigenvalue, ν is

the average neutron production per fission, and Σ_{f} is the macroscopic fission cross section. Note that macroscopic cross sections depend on the nuclide density field as:

$$\Sigma_r = \sigma_r^T N \ . \tag{30}$$

Here $r \in \{\mathrm{t}, \mathrm{a}, \mathrm{f}, \mathrm{s}\}$ as described above, and σ_r^T is the transposed vector of microscopic cross sections of type r. This provides the coupling from depletion to neutronics, and so the flux solution will vary with time. Therefore, given the time discretisation, Eq. (29) would be more precisely written for a given time-point as:

$$-\frac{\partial}{\partial x}\left(\frac{1}{3\Sigma_{\mathrm{t},n}}\frac{\partial \phi_n}{\partial x}\right) + \Sigma_{\mathrm{a},n}\phi_n = \frac{1}{k_n}\nu\Sigma_{\mathrm{f},n}\phi_n \ . \tag{31}$$

Only for simplicity, it is assumed that ν is independent of the nuclide composition in the analysis that follows – this can be easily remedied without substantial consequence to the expressions ultimately obtained. As specified previously, the spatial discretisation of the neutronics is not considered as the numerical comparisons to follow are finely discretised such that the continuous diffusion equation should be representative of the problem. This is also the case when Monte Carlo is used to obtain the neutronics solution, as it does not have spatial truncation error.

As described above, the Bateman operator in realistic reactor physics problems often takes the form of a matrix exponential of the burn-up matrix. However, the stability analysis to follow necessitates linearising the Bateman operator in terms of the flux – the analysis cannot readily be performed for the matrix exponential while giving a closed-form expression. During the analysis to follow, the Bateman operators will remain generic to allow the use of matrix exponentials subsequently, but analytic expressions will be stated for the simplified Bateman operators which have been used in previous works. The simplification that has previously been made involves taking the one-term Taylor expansion of the matrix exponential (equivalent to the explicit Euler discretisation of Eq. (3)). This is a relatively coarse approximation given the numerical stiffness of realistic depletion systems (Pusa, 2013), but it will be seen that the equations obtained are amenable to more accurate operators using numerical differentiation. Hence, the CE matrix exponential Bateman operator given by Eq. (7), which, given a flux solution at time-point n, evolves a nuclide density at time-point n, N_n, to that at time-point $n+1$, N_{n+1}, would be truncated to:

$$\mathbf{B} = \mathbf{I} + \Delta t \mathbf{A} = \mathbf{I} + \Delta t \left(\mathbf{\Sigma}\phi_n + \mathbf{\Lambda}\right) \ . \tag{32}$$

Here $\mathbf{I}$ is the identity matrix, Δt is the time-step length, (as described following Eq. (4)) $\mathbf{\Sigma}$ is the matrix containing microscopic cross sections and fission yields to account for neutron-induced reactions, ϕ_n is the scalar neutron flux at time-point n in the particular burnable region, and $\mathbf{\Lambda}$ is the matrix containing decay constants, accounting for nuclide loss and gain through decay. Performing substepping, the Bateman operator will differ from the simple variant given in Eq. (32). When extrapolating for S substeps, truncating the Bateman operator in Eq. (11) to first order in Δt gives:

$$\mathbf{B}_S^{\mathrm{LE}} = \prod_{s=0}^{S-1}\left[\mathbf{I} + \frac{\Delta t}{S}\left(\mathbf{\Sigma}\left(w_n^{(s)}\phi_n + w_{n-1}^{(s)}\phi_{n-1}\right) + \mathbf{\Lambda}\right)\right] \ . \tag{33}$$

Similarly, when performing a linear interpolation during the corrector step, the Bateman operator is:

$$\mathbf{B}_S^{\mathrm{LI}} = \prod_{s=0}^{S-1} \left[\mathbf{I} + \frac{\Delta t}{S} \left(\mathbf{\Sigma} \left(w_n^{(s)} \phi_n + w_{n+1}^{(s)} \phi_{n+1} \right) + \mathbf{\Lambda} \right) \right] . \tag{34}$$

Finally, when performing a quadratic interpolation during the corrector, one has:

$$\mathbf{B}_S^{\mathrm{QI}} = \prod_{s=0}^{S-1} \left[\mathbf{I} + \frac{\Delta t}{S} \left(\mathbf{\Sigma} \left(w_n^{(s)} \phi_n + w_{n+1}^{(s)} \phi_{n+1} + w_{n-1}^{(s)} \phi_{n-1} \right) + \mathbf{\Lambda} \right) \right] . \tag{35}$$

Having defined these various Bateman operators, a single step in a predictor-corrector scheme is given by two neutronic solutions and two appropriate Bateman operators, one for the predictor step and one for the corrector. The LE scheme can be described using only Eqs. (31) and (33).

For future convenience, two more items of notation around Bateman operators are introduced. First, as will become clear from the stability analysis, it is often desirable to evaluate a Bateman operator with all fluxes (beginning-of-step (n), end-of-step $(n+1)$, previous-step $(n-1)$) set to $\phi = \phi_0$. This will be denoted by a 0 superscript such that:

$$\mathbf{B}_S^{\mathrm{LI},0} = \mathbf{B}_S^{\mathrm{QI},0} = \mathbf{B}_S^{\mathrm{LE},0} = \mathbf{B}_S^0 . \tag{36}$$

This expression is identical for all substepping Bateman operators due to the flux being constant across time-steps and the interpolation/extrapolation weights summing to 1. For Bateman operators that have been truncated to first order in Δt this gives:

$$\mathbf{B}_S^0 = \prod_{s=0}^{S-1} \left[\mathbf{I} + \frac{\Delta t}{S} \left(\mathbf{\Sigma} \phi_0 + \mathbf{\Lambda} \right) \right] , \tag{37}$$

while for the truncated CE operator this would be:

$$\mathbf{B}_1^0 = \mathbf{B}^0 = \mathbf{I} + \Delta t \left(\mathbf{\Sigma} \phi_0 + \mathbf{\Lambda} \right) . \tag{38}$$

Finally, the expression for evaluating depletion over only one of S substeps with a fixed flux at ϕ_0 using a truncated Bateman operator will be written as:

$$\mathbf{b}_S = \mathbf{I} + \frac{\Delta t}{S} \left(\mathbf{\Sigma} \phi_0 + \mathbf{\Lambda} \right) . \tag{39}$$

3.1. Stability analysis

Fourier or Von Neumann stability analysis consists of several steps once the system of equations has been defined. The first involves expressing solutions of the system of equations in terms of an initial spatially-flat solution plus perturbations taking the form of different Fourier modes of the domain of interest. This necessitates that the domain is quite simple such that solutions may be written as a Fourier series. In this paper (as with Densmore

et al. (2013)), the spatial domain is taken to be $x \in [0, L]$ and the boundary conditions on the flux are reflective:

$$\left.\frac{\partial \phi}{\partial x}\right|_{x=0} = \left.\frac{\partial \phi}{\partial x}\right|_{x=L} = 0 \ . \tag{40}$$

The nuclide density field also has the initial condition:

$$N_0(x) = N_0 \ . \tag{41}$$

The necessity of specifying a normalisation for solving the eigenvalue problem sets ϕ_0, the initial, uniform flux. This normalisation is usually to power or power density. Taking P as the fixed power per unit area, this gives:

$$P = \int_0^L \kappa_{\mathrm{f}} \Sigma_{\mathrm{f},n} \phi_n dx \ . \tag{42}$$

Here κ_{f} is the energy release per fission, also assumed independent of composition for convenience. For $n = 0$ this gives the uniform initial flux as:

$$\phi_0 = \frac{P}{\kappa_{\mathrm{f}} \Sigma_{\mathrm{f},0} L} \ . \tag{43}$$

This uniformity also gives a simple expression for the initial eigenvalue:

$$k_0 = \frac{\nu \Sigma_{\mathrm{f},0}}{\Sigma_{\mathrm{a},0}} \ . \tag{44}$$

Given this, the nuclide density and neutronic solutions (flux and eigenvalue) may be expressed as:

$$N_n = N_0 + \delta N_n \ , \tag{45}$$

$$\phi_n = \phi_0 + \delta \phi_n \ , \tag{46}$$

$$k_n = k_0 + \delta k_n \ . \tag{47}$$

The flux and nuclide density perturbation terms can then be expressed as:

$$\delta N_n = \sum_{m=0}^{\infty} \delta N_{n,m} \cos\left(\frac{m\pi x}{L}\right) \ , \tag{48}$$

$$\delta \phi_n = \sum_{m=0}^{\infty} \delta \phi_{n,m} \cos\left(\frac{m\pi x}{L}\right) \ . \tag{49}$$

The eigenvalue is always uniform across the problem and so δk_n is described by only a zero-th Fourier mode.

The second step of the stability analysis consists of inserting the perturbation expressions for each of the variables into the governing equations and linearising, discarding any second-or-higher-order perturbation terms. The orthogonality of each Fourier mode allows them to

be considered separately. This will be done here for the flux equation while each depletion scheme will be treated in its own section. Inserting the perturbed variables into Eq. (31) gives:

$$\begin{aligned} &-\frac{\partial}{\partial x}\left(\frac{1}{3(\Sigma_{\mathrm{t},0}+\delta\Sigma_{\mathrm{t},n})}\frac{\partial(\phi_0+\delta\phi_n)}{\partial x}\right) \\ &+\left[\Sigma_{\mathrm{a},0}+\delta\Sigma_{\mathrm{a},n}-\frac{\nu}{k_0+\delta k_n}\left(\Sigma_{\mathrm{f},0}+\delta\Sigma_{\mathrm{f},n}\right)\right](\phi_0+\delta\phi_n)=0\ . \end{aligned} \tag{50}$$

This equation is expanded, second-order terms are eliminated, and one notes that ϕ_0 is spatially flat and the eigenvalue is defined by Eq. (44). This results in:

$$-\frac{1}{3\Sigma_{\mathrm{t},0}}\frac{\partial^2\delta\phi_n}{\partial x^2}=\left(\frac{\delta\Sigma_{\mathrm{f},n}}{\Sigma_{\mathrm{f},0}}-\frac{\delta\Sigma_{\mathrm{a},n}}{\Sigma_{\mathrm{a},0}}-\frac{\delta k_n}{k_0}\right)\Sigma_{\mathrm{a},0}\phi_0\ . \tag{51}$$

To proceed further with this equation, it must be decomposed into its Fourier modes. For $m>0$, using the definition of the macroscopic cross section from Eq. (30) and inserting Eqs. (48) and (49), one obtains:

$$\frac{1}{3\Sigma_{\mathrm{t},0}}\delta\phi_{n,m}\left(\frac{m\pi}{L}\right)^2=\left(\frac{\sigma_{\mathrm{f}}^T}{\Sigma_{\mathrm{f},0}}\delta N_{n,m}-\frac{\sigma_a^T}{\Sigma_{a,0}}\delta N_{n,m}\right)\Sigma_{\mathrm{a},0}\phi_0\ . \tag{52}$$

Note the disappearance of the eigenvalue term as it only contributes to the zero-th mode. Rearranging this expression while introducing the diffusion length $L_D=\frac{1}{\sqrt{3\Sigma_{\mathrm{t},0}\Sigma_{\mathrm{a},0}}}$, the flux perturbation in terms of the nuclide density perturbation is:

$$\delta\phi_{n,m}=\phi_0\left(\frac{L}{m\pi L_D}\right)^2\left(\frac{\sigma_{\mathrm{f}}^T}{\Sigma_{\mathrm{f},0}}-\frac{\sigma_{\mathrm{a}}^T}{\Sigma_{\mathrm{a},0}}\right)\delta N_{n,m}\ . \tag{53}$$

As Densmore et al. (2013) highlight, Eq. (51) cannot be used to obtain the zero-th mode flux perturbation. This can instead be obtained from Eq. (42) as:

$$\delta\phi_{n,0}=-\frac{\phi_0\sigma_{\mathrm{f}}^T}{\Sigma_{\mathrm{f},0}}\delta N_{n,0}\ . \tag{54}$$

For brevity, throughout the rest of this paper, the expression for converting a nuclide density perturbation to a flux perturbation will be written as:

$$\Phi_m^T=\begin{cases}\phi_0\left(\frac{L}{m\pi L_D}\right)^2\left(\frac{\sigma_{\mathrm{f}}^T}{\Sigma_{\mathrm{f},0}}-\frac{\sigma_{\mathrm{a}}^T}{\Sigma_{\mathrm{a},0}}\right) & m>0\\ -\frac{\phi_0\sigma_{\mathrm{f}}^T}{\Sigma_{\mathrm{f},0}} & m=0\end{cases}\ . \tag{55}$$

This expression will reoccur when performing the same expansion and linearisation of each depletion scheme in terms of perturbations. When that is performed for each in the following sections, expressions will be obtained relating the nuclide density perturbation at time-point n to that at time-point $n+1$. The eigenvalues of these expressions will allow the

determination of whether higher spatial mode perturbations grow or decay in amplitude with time – if they do grow then the depletion scheme is unstable.

It should be highlighted that in the expressions that will be obtained, the zero frequency terms will remain. This is because in burn-up the true flux and nuclide density solutions are time-dependent and the linearisations performed here are about the initial conditions. Therefore, it is possible that the zero frequency terms will be amplified over time, which is not in conflict with the physics. However, this is not an indicator of the stability of the system and so, while the expressions governing the zero frequencies will be obtained, they will not be analysed subsequently.

3.2. Linear Extrapolation

Assuming the existence of a previous nuclide density and neutronics solution, the LE scheme is described by performing a neutronics solution at the current time-point, n, and using it to evolve the current nuclide density with a Bateman operator given by, for example, Eq. (11) or Eq. (33):

$$N_{n+1} = \mathbf{B}_S^{\mathrm{LE}}(\phi_n, \phi_{n-1}) N_n \ . \tag{56}$$

Expanding nuclide densities and fluxes using their perturbation expressions gives:

$$N_0 + \delta N_{n+1} = \mathbf{B}_S^{\mathrm{LE}}(\phi_0 + \delta\phi_n, \phi_0 + \delta\phi_{n-1}) \left(N_0 + \delta N_n\right) \ . \tag{57}$$

Progress requires linearising the LE Bateman operator about ϕ_0 in terms of $\delta\phi_n$ and $\delta\phi_{n-1}$. That is, the Bateman operator is Taylor-expanded as:

$$\mathbf{B}_S^{\mathrm{LE}}(\phi_0 + \delta\phi_n, \phi_0 + \delta\phi_{n-1}) \approx \mathbf{B}_S^{\mathrm{LE}}(\phi_0, \phi_0) + \partial_{\phi_n}\mathbf{B}_S^{\mathrm{LE}}\delta\phi_n + \partial_{\phi_{n-1}}\mathbf{B}_S^{\mathrm{LE}}\delta\phi_{n-1} \ , \tag{58}$$

where:

$$\partial_{\phi_n}\mathbf{B}_S^{\mathrm{LE}} = \frac{\partial}{\partial\phi_n}\mathbf{B}_S^{\mathrm{LE}}|_{\phi_n=\phi_0,\phi_{n-1}=\phi_0} \ , \tag{59}$$

and:

$$\partial_{\phi_{n-1}}\mathbf{B}_S^{\mathrm{LE}} = \frac{\partial}{\partial\phi_{n-1}}\mathbf{B}_S^{\mathrm{LE}}|_{\phi_n=\phi_0,\phi_{n-1}=\phi_0} \ . \tag{60}$$

In the case where the LE Bateman operator is given by Eq. (33), these can be written analytically as:

$$\partial_{\phi_n}\mathbf{B}_S^{\mathrm{LE}} = \sum_{r=0}^{S-1}\left(\prod_{s=0}^{S-1}\left[(1-\delta_{rs})\mathbf{b}_S + \delta_{rs}\frac{\Delta t}{S}\mathbf{\Sigma}\right]\right) w_n^{(r)} \ , \tag{61}$$

and:

$$\partial_{\phi_{n-1}}\mathbf{B}_S^{\mathrm{LE}} = \sum_{r=0}^{S-1}\left(\prod_{s=0}^{S-1}\left[(1-\delta_{rs})\mathbf{b}_S + \delta_{rs}\frac{\Delta t}{S}\mathbf{\Sigma}\right]\right) w_{n-1}^{(r)} \ . \tag{62}$$

Here, δ_{rs} is the Kronecker delta. Assuming that these Bateman derivative expressions can be evaluated by some means for a given Bateman operator, inserting Eq. (58) into Eq. (57) and neglecting higher-order terms and cancelling the N_0 terms on the left- and right-hand sides gives:

$$\delta N_{n+1} = \mathbf{B}_S^0 \delta N_n + \left[\mathbf{B}_S^0 - \mathbf{I}\right] N_0 + \partial_{\phi_n} \mathbf{B}_S^{\mathrm{LE}} N_0 \delta\phi_n + \partial_{\phi_n} \mathbf{B}_S^{\mathrm{LE}} N_0 \delta\phi_{n-1} \ . \tag{63}$$

Decomposing Eq. (63) into Fourier modes, both the $\delta\phi$ terms are given by the application of the nuclide-to-flux conversion factor from Eq. (55). This gives:

$$\delta N_{n+1,m} = \left[\partial_{\phi_n} \mathbf{B}^{\mathrm{LE}} N_0 \Phi_m^T + \mathbf{B}_S^0\right] \delta N_{n,m} + \delta_{m0} \left[\mathbf{B}_S^0 - \mathbf{I}\right] N_0 + \partial_{\phi_{n-1}} \mathbf{B}_S^{\mathrm{LE}} N_0 \Phi_m^T \delta N_{n-1,m} \ . \tag{64}$$

In order to obtain a single matrix acting on the nuclide density perturbation, a change of variables is necessary. A compound nuclide density vector is introduced which concatenates the current and previous nuclide densities:

$$X_n = \begin{bmatrix} N_n \\ N_{n-1} \end{bmatrix} \ , \tag{65}$$

and, correspondingly:

$$\delta X_n = \begin{bmatrix} \delta N_n \\ \delta N_{n-1} \end{bmatrix} \ . \tag{66}$$

As well, for convenience:

$$X_0 = \begin{bmatrix} N_0 \\ 0 \end{bmatrix} \ . \tag{67}$$

Hence, Eq. (64) can be rewritten as:

$$\delta X_{n+1,m} = \mathbf{C}_m \delta X_{n,m} + \delta_{m0} \mathbf{D}_0 X_0 \ , \tag{68}$$

where:

$$\mathbf{C}_m = \begin{bmatrix} \partial_{\phi_n} \mathbf{B}^{\mathrm{LE}} N_0 \Phi_m^T + \mathbf{B}_S^0 & \partial_{\phi_{n-1}} \mathbf{B}^{\mathrm{LE}} N_0 \Phi_m^T \\ \mathbf{I} & \mathbf{0} \end{bmatrix} \ , \tag{69}$$

and:

$$\mathbf{D}_0 = \begin{bmatrix} \mathbf{B}_S^0 - \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \ . \tag{70}$$

3.3. Constant Extrapolation/Linear Interpolation

The CE/LI scheme with substeps is relatively simple compared to the others examined here in that it ultimately does not differ substantially from the CE/LI scheme without substeps described in Cosgrove et al. (2020c). Other than including substeps, the primary difference here is that the Bateman operators are allowed to remain generic. Given a nuclide density at time-point n, the scheme is described by obtaining a neutronics solution and using it to apply a CE Bateman operator (such as those given by Eq. (7) or Eq. (32)) to obtain a predicted nuclide density at time-point $n+1$, denoted $N_{n+1}^{(0)}$. Explicitly:

$$N_{n+1}^{(0)} = \mathbf{B}(\phi_n) N_n \ . \tag{71}$$

A neutronics solution is obtained from $N_{n+1}^{(0)}$, $\phi_{n+1}^{(0)}$, and used to calculate the final N_{n+1} with an LI Bateman operator (from the likes of Eq. (16) or Eq. (34)):

$$N_{n+1} = \mathbf{B}_S^{\mathrm{LI}}(\phi_n, \phi_{n+1}^{(0)}) N_n \ . \tag{72}$$

Hence, one must propagate the nuclide density perturbation through Eqs. (71) and (72).

Expressing Eq. (71) in terms of the density perturbation was shown originally in Densmore et al. (2013) for the simple Bateman operator given by Eq. (32). In the generic Bateman operator case, as was done for the LE scheme, this necessitates linearising the Bateman operator in terms of the flux – here only ϕ_n for the CE operator. The end result for each Fourier mode is:

$$\delta N_{n+1,m}^{(0)} = \mathbf{B}^0 \delta N_{n,m} + \delta_{m0} \left[\mathbf{B}^0 - \mathbf{I}\right] N_0 + \partial_{\phi_n} \mathbf{B} N_0 \Phi_m^T \delta N_n \ . \tag{73}$$

Here, $\partial_{\phi_n} \mathbf{B}$ is the derivative of the CE Bateman operator evaluated at $\phi_n = \phi_0$, specifically:

$$\partial_{\phi_n} \mathbf{B} = \frac{\partial}{\partial \phi_n} \mathbf{B}|_{\phi_n = \phi_0} \ . \tag{74}$$

In the particular case of the CE Bateman operator being given by Eq. (32) one has:

$$\partial_{\phi_n} \mathbf{B} = \Delta t \boldsymbol{\Sigma} \ . \tag{75}$$

Linearising and simplifying Eq. (72), one ends up with:

$$\delta N_{n+1} = \mathbf{B}_S^0 \delta N_n + \left[\mathbf{B}_S^0 - \mathbf{I}\right] N_0 + \partial_{\phi_n} \mathbf{B}_S^{\mathrm{LI}} N_0 \delta \phi_n + \partial_{\phi_{n+1}} \mathbf{B}_S^{\mathrm{LI}} N_0 \delta \phi_{n+1}^{(0)} \ . \tag{76}$$

Again, the final two terms here are merely the derivatives of the LI Bateman operator with respect to each flux and multiplied by the appropriate flux perturbations. Formally, these are:

$$\partial_{\phi_n} \mathbf{B}_S^{\mathrm{LI}} = \frac{\partial}{\partial \phi_n} \mathbf{B}_S^{\mathrm{LI}}|_{\phi_n = \phi_0, \phi_{n+1} = \phi_0} \ , \tag{77}$$

and:

$$\partial_{\phi_{n+1}} \mathbf{B}_S^{\mathrm{LI}} = \frac{\partial}{\partial \phi_{n+1}} \mathbf{B}_S^{\mathrm{LI}}|_{\phi_n = \phi_0, \phi_{n+1} = \phi_0} \ . \tag{78}$$

For the LI Bateman operator given by Eq. (34), these derivatives can be written as:

$$\partial_{\phi_n} \mathbf{B}^{\mathrm{LI}} = \sum_{r=0}^{S-1} \left(\prod_{s=0}^{S-1} \left[(1 - \delta_{rs}) \mathbf{b}_S + \delta_{rs} \frac{\Delta t}{S} \boldsymbol{\Sigma} \right] \right) w_n^{(r)} \ , \tag{79}$$

and:

$$\partial_{\phi_{n+1}}\mathbf{B}^{\mathrm{LI}} = \sum_{r=0}^{S-1}\left(\prod_{s=0}^{S-1}\left[(1-\delta_{rs})\mathbf{b}_S + \delta_{rs}\frac{\Delta t}{S}\mathbf{\Sigma}\right]\right) w_{n+1}^{(r)} \ . \tag{80}$$

Completing the expression for the CE/LI scheme only requires separating Eq. (76) into Fourier modes and inserting Eq. (73) for the $\delta\phi_{n+1}^{(0)}$ term. This gives:

$$\delta N_{n+1,m} = \mathbf{C}_m \delta N_{n,m} + \delta_{m0}\mathbf{D}N_0 \ , \tag{81}$$

where:

$$\mathbf{C}_m = \mathbf{B}_S^0 + \partial_{\phi_n}\mathbf{B}^{\mathrm{LI}}N_0\Phi_m^T + \partial_{\phi_{n+1}}\mathbf{B}^{\mathrm{LI}}N_0\Phi_m^T\left[\mathbf{B}^0 + \partial_{\phi_n}\mathbf{B}N_0\Phi_m^T\right] \ , \tag{82}$$

and:

$$\mathbf{D} = \left[\mathbf{B}_S^0 - \mathbf{I}\right] + \partial_{\phi_{n+1}}\mathbf{B}^{\mathrm{LI}}N_0\Phi_m^T\left[\mathbf{B}^0 - \mathbf{I}\right] \ . \tag{83}$$

3.4. Constant Extrapolation/Linear Interpolation while relaxing the corrector step

When performing a relaxation on the corrector step, the $(j+1)$-th iteration of the corrector would be written as:

$$N_{n+1}^{(j+1)} = \alpha\mathbf{B}_S^{\mathrm{LI}}\left(\phi_n, \phi_{n+1}^{(j)}\right)N_n + (1-\alpha)\,N_{n+1}^{(j)} \ . \tag{84}$$

Expanding this equation using the perturbation expressions, neglecting higher-order terms and simplifying gives:

$$\begin{aligned}\delta N_{n+1}^{(j+1)} &= \alpha\mathbf{B}_S^0\delta N_n + \alpha\partial_{\phi_n}\mathbf{B}^{\mathrm{LI}}N_0\delta\phi_n \\ +\alpha\partial_{\phi_{n+1}}\mathbf{B}^{\mathrm{LI}}N_0\delta\phi_{n+1}^{(j)} &+ \alpha\left[\mathbf{B}_S^0 - \mathbf{I}\right]N_0 + (1-\alpha)\delta N_{n+1}^{(j)} \ .\end{aligned} \tag{85}$$

It remains to express $\delta N_{n+1}^{(j)}$ in terms of δN_n and N_0. This requires a recurrence relation, making use of Eqs. (73) and (81). After some algebra, one ultimately obtains the following expression:

$$\delta N_{n+1,m}^{(j+1)} = \mathbf{C}_m^{(j)}\delta N_{n,m} + \delta_{m0}\mathbf{D}_0^{(j)}N_0 \ , \tag{86}$$

where:

$$\mathbf{C}_m^{(j)} = \alpha\left[\mathbf{B}_S^0 + \partial_{\phi_n}\mathbf{B}^{\mathrm{LI}}N_0\Phi_m^T\right] + \left[\alpha\partial_{\phi_{n+1}}\mathbf{B}^{\mathrm{LI}}N_0\Phi_m^T + (1-\alpha)\mathbf{I}\right]\mathbf{C}_m^{(j-1)} \ , \tag{87}$$

and:

$$\mathbf{D}_0^{(j)} = \alpha\left[\mathbf{B}_S^0 - \mathbf{I}\right] + \left[\alpha\partial_{\phi_{n+1}}\mathbf{B}^{\mathrm{LI}}N_0\Phi_m^T + (1-\alpha)\mathbf{I}\right]\mathbf{D}_0^{(j-1)} \ . \tag{88}$$

These are initialised by:

$$\mathbf{C}_m^{(0)} = \alpha\left[\mathbf{B}_S^0 + \partial_{\phi_n}\mathbf{B}^{\mathrm{LI}}N_0\Phi_m^T\right] + \left[\alpha\partial_{\phi_{n+1}}\mathbf{B}^{\mathrm{LI}}N_0\Phi_m^T + (1-\alpha)\mathbf{I}\right]\left[\mathbf{B}^0 + \partial\mathbf{B}_{\phi_n}N_0\Phi_m^T\right] \ , \tag{89}$$

and:

$$\mathbf{D}_0^{(0)} = \alpha\left[\mathbf{B}_S^0 - \mathbf{I}\right] + \left[\alpha\partial_{\phi_{n+1}}\mathbf{B}^{\mathrm{LI}}N_0\Phi_m^T + (1-\alpha)\mathbf{I}\right]\left[\mathbf{B}^0 - \mathbf{I}\right] \ . \tag{90}$$

3.5. Linear Extrapolation/Linear Interpolation

Analysis of the LE/LI scheme follows straightforwardly from the LE and CE/LI: its predictor equation is given by Eq. (56) and its corrector by Eq. (72). Emphasising that the predictor and corrector can generally use different numbers of substeps, S_p will be used to denote the number of substeps during the predictor while S_c will denote the number of substeps during the corrector. The main algebraic manipulation to perform is expressing the flux perturbation, $\delta\phi^{(0)}_{n+1}$, in Eq. (76) in terms of the predictor nuclide density perturbation from Eq. (64). Ultimately this gives:

$$\begin{aligned}\delta N_{n+1,m} = &\left[\mathbf{B}^0_{S_c} + \partial_{\phi_n}\mathbf{B}^{\mathrm{LI}}_{S_c} N_0 \Phi^T_m + \partial_{\phi_{n+1}}\mathbf{B}^{\mathrm{LI}}_{S_c} N_0 \Phi^T_m \left[\mathbf{B}^0_{S_p} + \partial_{\phi_n}\mathbf{B}^{\mathrm{LE}}_{S_p} N_0 \Phi^T_m\right]\right] \delta N_{n,m} \\ &+\partial_{\phi_{n+1}}\mathbf{B}^{\mathrm{LI}}_{S_c} N_0 \Phi^T_m \partial_{\phi_{n-1}}\mathbf{B}^{\mathrm{LE}}_{S_p} N_0 \Phi^T_m \delta N_{n-1,m} + \delta_{m0}\left[\left[\mathbf{B}^0_{S_c} - \mathbf{I}\right] + \partial_{\phi_{n+1}}\mathbf{B}^{\mathrm{LI}}_{S_c} N_0 \Phi^T_m \left[\mathbf{B}^0_{S_p} - \mathbf{I}\right]\right] N_0 \ .\end{aligned} \tag{91}$$

Making use, again, of the concatenated nuclide density vector defined by Eq. (65), this can be written more conveniently as:

$$\delta X_{n+1,m} = \mathbf{C}_m \delta X_{n,m} + \delta_{m0}\mathbf{D}_0 X_0 \ , \tag{92}$$

where the elements of $\mathbf{C}_m$ are:

$$(\mathbf{C}_m)_{1,1} = \mathbf{B}^0_{S_c} + \partial_{\phi_n}\mathbf{B}^{\mathrm{LI}}_{S_c} N_0 \Phi^T_m + \partial_{\phi_{n+1}}\mathbf{B}^{\mathrm{LI}}_{S_c} N_0 \Phi^T_m \left[\mathbf{B}^0_{S_p} + \partial_{\phi_n}\mathbf{B}^{\mathrm{LE}}_{S_p} N_0 \Phi^T_m\right] \ , \tag{93a}$$

$$(\mathbf{C}_m)_{1,2} = \partial_{\phi_{n+1}}\mathbf{B}^{\mathrm{LI}}_{S_c} N_0 \Phi^T_m \partial_{\phi_{n-1}}\mathbf{B}^{\mathrm{LE}}_{S_p} N_0 \Phi^T_m \ , \tag{93b}$$

$$(\mathbf{C}_m)_{2,1} = \mathbf{I} \ , \tag{93c}$$

$$(\mathbf{C}_m)_{2,2} = \mathbf{0} \ , \tag{93d}$$

And $\mathbf{D}_0$ is:

$$\mathbf{D}_0 = \begin{bmatrix} \left[\mathbf{B}^0_{S_c} - \mathbf{I}\right] + \partial_{\phi_{n+1}}\mathbf{B}^{\mathrm{LI}}_{S_c} N_0 \Phi^T_m \left[\mathbf{B}^0_{S_p} - \mathbf{I}\right] & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \ . \tag{94}$$

3.6. Linear Extrapolation/Linear Interpolation while relaxing the corrector step

Obtaining expressions for the implicit LE/LI scheme follows much the same logic as in Section 3.4. Indeed, the equation describing the corrector step is Eq. (84) with the corresponding perturbation expansion given by Eq. (85). The only additional complication arises from including the flux perturbation during the $(n-1)$-th time-point which propagates to the $\delta N^{(j)}_{n+1}$ and $\delta\phi^{(j)}_{n+1}$ terms in Eq. (85) due to the LE predictor. This is done using Eq. (68). The final expression obtained is:

$$\delta X^{(j+1)}_{n+1,m} = \mathbf{C}^{(j)}_m \delta X_{n,m} + \delta_{m0}\mathbf{D}^{(j)}_0 X_0 \ , \tag{95}$$

where the element of $\mathbf{C}_m^{(j)}$ are:

$$\left(\mathbf{C}_m^{(j)}\right)_{1,1} = \alpha\left[\mathbf{B}_{S_c}^0 + \partial_{\phi_n}\mathbf{B}_{S_c}^{\mathrm{LI}} N_0 \Phi_m^T\right] + \left[\alpha\partial_{\phi_{n+1}}\mathbf{B}_{S_c}^{\mathrm{LI}} N_0 \Phi_m^T + (1-\alpha)\mathbf{I}\right]\left(\mathbf{C}_m^{(j-1)}\right)_{1,1} , \tag{96a}$$

$$\left(\mathbf{C}_m^{(j)}\right)_{1,2} = \left[\alpha\partial_{\phi_{n+1}}\mathbf{B}_{S_c}^{\mathrm{LI}} N_0 \Phi_m^T + (1-\alpha)\mathbf{I}\right]^{j+1} \partial_{\phi_{n-1}}\mathbf{B}_{S_p}^{\mathrm{LE}} N_0 \Phi_m^T , \tag{96b}$$

$$\left(\mathbf{C}_m\right)_{2,1} = \mathbf{I} , \tag{96c}$$

$$\left(\mathbf{C}_m\right)_{2,2} = \mathbf{0} , \tag{96d}$$

and $\left(\mathbf{C}_m^{(j)}\right)_{1,1}$ is initialised by:

$$\left(\mathbf{C}_m^{(0)}\right)_{1,1} = \alpha\left[\mathbf{B}_{S_c}^0 + \partial_{\phi_n}\mathbf{B}_{S_c}^{\mathrm{LI}} N_0 \Phi_m^T\right] + \left[\alpha\partial_{\phi_{n+1}}\mathbf{B}_{S_c}^{\mathrm{LI}} N_0 \Phi_m^T + (1-\alpha)\mathbf{I}\right]\left[\partial_{\phi_n}\mathbf{B}_{S_p}^{\mathrm{LE}} N_0 \Phi_m^T + \mathbf{B}_{S_p}^0\right] . \tag{97}$$

The only non-zero matrix element of $\mathbf{D}_0^{(j)}$ is:

$$\left(\mathbf{D}_0^{(j)}\right)_{1,1} = \alpha\left[\mathbf{B}_{S_c}^0 - \mathbf{I}\right] + \left[\alpha\partial_{\phi_{n+1}}\mathbf{B}_{S_c}^{\mathrm{LI}} N_0 \Phi_m^T + (1-\alpha)\mathbf{I}\right]\left(\mathbf{D}_0^{(j-1)}\right)_{1,1} . \tag{98}$$

This is initialised by:

$$\left(\mathbf{D}_0^{(0)}\right)_{1,1} = \alpha\left[\mathbf{B}_{S_c}^0 - \mathbf{I}\right] + \left[\alpha\partial_{\phi_{n+1}}\mathbf{B}_{S_c}^{\mathrm{LI}} N_0 \Phi_m^T + (1-\alpha)\mathbf{I}\right]\left[\mathbf{B}_{S_p}^0 - \mathbf{I}\right] . \tag{99}$$

3.7. Linear Extrapolation/Quadratic Interpolation

The LE/QI scheme differs from LE/QI by having a QI corrector Bateman operator, such as Eq. (24) or Eq. (35). Expanding the corrector equation in its perturbation terms gives:

$$N_0 + \delta N_{n+1} = \mathbf{B}_{S_c}^{\mathrm{QI}}\left(\phi_0 + \delta\phi_n, \phi_0 + \delta\phi_{n+1}, \phi_0 + \delta\phi_{n-1}\right)\left(N_0 + \delta N_n\right) . \tag{100}$$

Simplifying and neglecting second-order terms gives:

$$\begin{aligned}\delta N_{n+1} = \mathbf{B}_{S_c}^0 \delta N_n + \partial_{\phi_n}\mathbf{B}_{S_c}^{\mathrm{QI}} N_0 \delta\phi_n + \partial_{\phi_{n+1}}\mathbf{B}_{S_c}^{\mathrm{QI}} N_0 \delta\phi_{n+1} + \\ \partial_{\phi_{n-1}}\mathbf{B}_{S_c}^{\mathrm{QI}} N_0 \delta\phi_{n-1} + \left[\mathbf{B}_{S_c}^0 - \mathbf{I}\right] N_0 .\end{aligned} \tag{101}$$

As with the other Bateman operators, the derivatives terms are defined as:

$$\partial_{\phi_n}\mathbf{B}_{S_c}^{\mathrm{QI}} = \frac{\partial}{\partial\phi_n}\mathbf{B}_{S_c}^{\mathrm{QI}}|_{\phi_n=\phi_0,\, \phi_{n+1}=\phi_0,\, \phi_{n-1}=\phi_0} , \tag{102}$$

$$\partial_{\phi_{n+1}}\mathbf{B}_{S_c}^{\mathrm{QI}} = \frac{\partial}{\partial\phi_{n+1}}\mathbf{B}_{S_c}^{\mathrm{LI}}|_{\phi_n=\phi_0,\, \phi_{n+1}=\phi_0,\, \phi_{n-1}=\phi_0} , \tag{103}$$

and:

$$\partial_{\phi_{n-1}}\mathbf{B}_{S_c}^{\mathrm{QI}} = \frac{\partial}{\partial\phi_{n-1}}\mathbf{B}_{S_c}^{\mathrm{LI}}|_{\phi_n=\phi_0,\, \phi_{n+1}=\phi_0,\, \phi_{n-1}=\phi_0} . \tag{104}$$

For the QI Bateman operator given by Eq. (35), the Bateman derivative terms are:

$$\partial_{\phi_n}\mathbf{B}_{S_c}^{\mathrm{QI}} = \sum_{r=0}^{S_c-1}\left(\prod_{s=0}^{S_c-1}\left[(1-\delta_{rs})\mathbf{b}_{S_c}+\delta_{rs}\frac{\Delta t}{S_c}\mathbf{\Sigma}\right]\right)w_n^{(r)}\ , \tag{105}$$

$$\partial_{\phi_{n+1}}\mathbf{B}_{S_c}^{\mathrm{QI}} = \sum_{r=0}^{S_c-1}\left(\prod_{s=0}^{S_c-1}\left[(1-\delta_{rs})\mathbf{b}_{S_c}+\delta_{rs}\frac{\Delta t}{S_c}\mathbf{\Sigma}\right]\right)w_{n+1}^{(r)}\ , \tag{106}$$

$$\partial_{\phi_{n-1}}\mathbf{B}_{S_c}^{\mathrm{QI}} = \sum_{r=0}^{S_c-1}\left(\prod_{s=0}^{S_c-1}\left[(1-\delta_{rs})\mathbf{b}_{S_c}+\delta_{rs}\frac{\Delta t}{S_c}\mathbf{\Sigma}\right]\right)w_{n-1}^{(r)}\ . \tag{107}$$

Recall that the interpolation weights here are quadratic, rather than linear.

Both the $\delta\phi_n$ and $\delta\phi_{n-1}$ terms are obtained by applying the nuclide-to-flux perturbation conversion to the appropriate density perturbations, while the $\delta\phi_{n+1}$ term, as before, is the flux perturbation corresponding to the nuclide density perturbation from the LE predictor (Eq. (68)). Carrying out this algebra, one obtains:

$$\delta X_{n+1,m} = \mathbf{C}_m\delta X_{n,m} + \delta_{m0}\mathbf{D}_0 X_0\ . \tag{108}$$

Here the elements of $\mathbf{C}_m$ are:

$$(\mathbf{C}_m)_{1,1} = \mathbf{B}_{S_c}^0 + \partial_{\phi_n}\mathbf{B}_{S_c}^{\mathrm{QI}}N_0\Phi_m^T + \partial_{\phi_{n+1}}\mathbf{B}_{S_c}^{\mathrm{QI}}N_0\Phi_m^T\left[\mathbf{B}_{S_p}^0 + \partial_{\phi_n}\mathbf{B}_{S_p}^{\mathrm{LE}}N_0\Phi_m^T\right]\ , \tag{109a}$$

$$(\mathbf{C}_m)_{1,2} = \partial_{\phi_{n-1}}\mathbf{B}_{S_c}^{\mathrm{QI}}N_0\Phi_m^T + \partial_{\phi_{n+1}}\mathbf{B}_{S_c}^{\mathrm{QI}}N_0\Phi_m^T\partial_{\phi_{n-1}}\mathbf{B}_{S_p}^{\mathrm{LE}}N_0\Phi_m^T\ , \tag{109b}$$

$$(\mathbf{C}_m)_{2,1} = \mathbf{I}\ , \tag{109c}$$

$$(\mathbf{C}_m)_{2,2} = \mathbf{0}\ , \tag{109d}$$

And $\mathbf{D}_0$ is:

$$\mathbf{D}_0 = \begin{bmatrix}\left[\mathbf{B}_{S_c}^0 - \mathbf{I}\right] + \partial_{\phi_{n+1}}\mathbf{B}_{S_c}^{\mathrm{QI}}N_0\Phi_m^T\left[\mathbf{B}_{S_p}^0 - \mathbf{I}\right] & \mathbf{0}\\ \mathbf{0} & \mathbf{0}\end{bmatrix}\ . \tag{110}$$

3.8. Linear Extrapolation/Quadratic Interpolation while relaxing the corrector step

The implicit LE/QI scheme is the last to be investigated and follows straightforwardly from combining Sections 3.6 and 3.7. The corrector when iterating and relaxing the QI Bateman operator is:

$$N_{n+1}^{(j+1)} = \alpha\mathbf{B}_{S_c}^{\mathrm{QI}}\left(\phi_n, \phi_{n+1}^{(j)}, \phi_{n-1}\right)N_n + (1-\alpha)\,N_{n+1}^{(j)}\ . \tag{111}$$

Following the same logic as for the implicit LE/LI, one obtains:

$$\delta X_{n+1,m}^{(j+1)} = \mathbf{C}_m^{(j)}\delta X_{n,m} + \delta_{m0}\mathbf{D}_0^{(j)}X_0\ . \tag{112}$$

The elements of $\mathbf{C}_m^{(j)}$ are:

$$\left(\mathbf{C}_m^{(j)}\right)_{1,1} = \alpha\left[\mathbf{B}_{S_c}^0 + \partial_{\phi_n}\mathbf{B}_{S_c}^{\mathrm{QI}} N_0 \Phi_m^T\right] + \left[\alpha\partial_{\phi_{n+1}}\mathbf{B}_{S_c}^{\mathrm{QI}} N_0 \Phi_m^T + (1-\alpha)\mathbf{I}\right]\left(\mathbf{C}_m^{(j-1)}\right)_{1,1} , \quad (113a)$$

$$\left(\mathbf{C}_m^{(j)}\right)_{1,2} = \alpha\partial_{\phi_{n-1}}\mathbf{B}_{S_c}^{\mathrm{QI}} N_0 \Phi_m^T + \left[\alpha\partial_{\phi_{n+1}}\mathbf{B}_{S_c}^{\mathrm{QI}} N_0 \Phi_m^T + (1-\alpha)\mathbf{I}\right]\left(\mathbf{C}_m^{(j-1)}\right)_{1,2} , \quad (113b)$$

$$\left(\mathbf{C}_m\right)_{2,1} = \mathbf{I} , \quad (113c)$$

$$\left(\mathbf{C}_m\right)_{2,2} = \mathbf{0} . \quad (113d)$$

Here $\left(\mathbf{C}_m^{(j)}\right)_{1,1}$ is initialised by:

$$\left(\mathbf{C}_m^{(0)}\right)_{1,1} = \alpha\left[\mathbf{B}_{S_c}^0 + \partial_{\phi_n}\mathbf{B}_{S_c}^{\mathrm{QI}} N_0 \Phi_m^T\right] + \left[\alpha\partial_{\phi_{n+1}}\mathbf{B}_{S_c}^{\mathrm{QI}} N_0 \Phi_m^T + (1-\alpha)\mathbf{I}\right]\left[\partial_{\phi_n}\mathbf{B}_{S_p}^{\mathrm{LE}} N_0 \Phi_m^T + \mathbf{B}_{S_p}^0\right] , \quad (114)$$

and $\left(\mathbf{C}_m^{(j)}\right)_{1,2}$ is initialised by:

$$\left(\mathbf{C}_m^{(0)}\right)_{1,2} = \alpha\partial_{\phi_{n-1}}\mathbf{B}_{S_c}^{\mathrm{QI}} N_0 \Phi_m^T + \left[\alpha\partial_{\phi_{n+1}}\mathbf{B}_{S_c}^{\mathrm{QI}} N_0 \Phi_m^T + (1-\alpha)\mathbf{I}\right]\partial_{\phi_{n-1}}\mathbf{B}_{S_p}^{\mathrm{LE}} N_0 \Phi_m^T . \quad (115)$$

The only non-zero matrix element of $\mathbf{D}_0^{(j)}$ is:

$$\left(\mathbf{D}_0^{(j)}\right)_{1,1} = \alpha\left[\mathbf{B}_{S_c}^0 - \mathbf{I}\right] + \left[\alpha\partial_{\phi_{n+1}}\mathbf{B}_{S_c}^{\mathrm{QI}} N_0 \Phi_m^T + (1-\alpha)\mathbf{I}\right]\left(\mathbf{D}_0^{(j-1)}\right)_{1,1} . \quad (116)$$

This is initialised by:

$$\left(\mathbf{D}_0^{(0)}\right)_{1,1} = \alpha\left[\mathbf{B}_{S_c}^0 - \mathbf{I}\right] + \left[\alpha\partial_{\phi_{n+1}}\mathbf{B}_{S_c}^{\mathrm{QI}} N_0 \Phi_m^T + (1-\alpha)\mathbf{I}\right]\left[\mathbf{B}_{S_p}^0 - \mathbf{I}\right] . \quad (117)$$

3.9. On the use of more complex Bateman operators

This paper and previous works have used simple Bateman operators in order to produce analytic expressions for the matrices governing the stability of the burn-up system. To be precise, the CE Bateman operator used in many reactor physics tools is the matrix exponential:

$$\mathbf{B} = \exp\left[\left(\mathbf{\Sigma}\phi + \mathbf{\Lambda}\right)\Delta t\right] . \quad (118)$$

This work has so far used the Taylor expansion of the matrix exponential, truncated to first order (Eq. (32)) to allow analytic results to be obtained. However, this simple operator is inadequate when dealing with realistic, numerically stiff depletion systems, as is a higher-order Taylor expansion (Moler and Van Loan, 2003; Pusa and Leppänen, 2010). This must be resolved if this stability analysis is to be put to practical purpose, e.g., providing an on-line estimate of the maximum allowable time-step or giving insight into the behaviour of realistic problems.

This work has used a derivative notation for Bateman operators, partly for notational convenience and clarity to the expressions obtained, e.g., $\partial_{\phi_n}\mathbf{B}$. Each of the expressions are exact and, for a time-point parametrised by $\eta \in \{n-1, n, n+1\}$, are equivalent to:

$$\partial_{\phi_\eta}\mathbf{B} = \frac{\partial}{\partial\phi_\eta}\mathbf{B}(\phi_\eta)|_{\phi_\eta=\phi_0} . \quad (119)$$

Where more than one flux term features in the Bateman operator, they are also set to $\phi = \phi_0$. In principle, the $\mathbf{B}$ and derivative notation used in this work could be replaced with the use of any form of Bateman operator and an exact derivative with respect to the flux used in the burn-up matrix. However, difficulty arises when using the common matrix exponential due to the burn-up matrix. One could easily obtain an analytic expression for the matrix exponential's derivative with respect to flux provided there was no decay matrix, $\mathbf{\Lambda}$; this is because parameter differentiation of the matrix exponential is straightforward provided the exponentiated matrix commutes with its derivative. If so, for a generic matrix $\mathbf{M}$, differentiated by the scalar β to give a matrix $\partial_\beta \mathbf{M}$, the corresponding derivative of the matrix exponential is:

$$\frac{\partial}{\partial \beta} \exp\left[\mathbf{M}\right] = \partial_\beta \mathbf{M} \exp\left[\mathbf{M}\right] = \exp\left[\mathbf{M}\right] \partial_\beta \mathbf{M} \, . \tag{120}$$

Otherwise, one obtains the less convenient expression (Najfeld and Havel, 1995; Fung, 2004):

$$\frac{\partial}{\partial \beta} \exp\left[\mathbf{M}\right] = \int_0^1 \exp\left[(1-\tau)\mathbf{M}\right] \partial_\beta \mathbf{M} \exp\left[\tau \mathbf{M}\right] \mathrm{d}\tau \, . \tag{121}$$

If $\mathbf{M} = \mathbf{A}\Delta t = (\mathbf{\Sigma}\phi + \mathbf{\Lambda})\,\Delta t$, then obviously $\mathbf{M}$ does not generally commute with its derivative with respect to flux, $\partial_\phi \mathbf{M} = \mathbf{\Sigma}\Delta t$, forcing the use of the more complicated expression.

Alternatively, one need not obtain analytic expressions for the derivative of the matrix exponential (or a product of matrix exponentials when substepping). Instead, the derivative expressions may be evaluated numerically. There are several methods that one might use for this evaluation. The most simple and familiar is central differencing, although this would require some judgement or experimentation to ensure an appropriate step size is selected without inducing a substantial loss of precision.

Another method may be the complex-step derivative approximation (Martins et al., 2003): this method is much less sensitive to step-size selection, but, unfortunately, cannot be used where complex arithmetic is involved in the computation, as is the case with the common CRAM matrix exponential evaluation.

Another method is automatic or algorithmic differentiation. Given that the numerical evaluation consists of a series of simple operations and elementary function evaluations, it is possible to differentiate this evaluation using only the chain rule and knowledge of simple operations and elementary function derivatives. The resulting derivative may be inconvenient to write or program by hand, but a compiler may generate such a derivative function relatively easily given an input function (Gebremedhin and Walther, 2020). While such an approach is most attractive when evaluating many derivatives (or a Jacobian), automatic differentiation is also advantageous for maintaining numerical precision for a single derivative, even if at a slightly higher computational cost than central differencing or the complex-step approximation. Using an automatic differentiation tool, it is feasible to reliably obtain stability estimates for more complex depletion systems than examined in previous work.

4. Numerical investigations

Having obtained several expressions for the different burn-up schemes, their stability will be evaluated against two numerical cases. First, using the simple system described in Densmore et al. (2013). Second, using a more complicated system of nuclides with its analysis enabled by automatic differentiation. Both sections will examine the eigenvalues produced by the expressions derived in the previous section and compare them against simulations.

Where simulations are performed, these are done using a MATLAB script which numerically solves the diffusion equation in 150 discrete meshes (which appears sufficiently fine to justify treating the diffusion equation continuously in stability estimates) and performs all depletion operations. Likewise, all stability estimates are evaluated using a MATLAB script. In all cases the eigenvalues are obtained by computing the appropriate $\mathbf{C}_m$ matrices obtained in the previous section for a given burn-up scheme and problem and inserting them into MATLAB's eig function.

4.1. Two nuclide system

This case is comparatively simple: a system featuring two nuclides, one fissile, one fission product, neither decaying. The fission product is produced with a yield of one. The cross sections (in barns) of the nuclides are given by the following vectors:

$$\sigma_\mathrm{a} = \begin{bmatrix} 3000 \\ 150 \end{bmatrix}, \ \sigma_\mathrm{f} = \begin{bmatrix} 3000 \\ 0 \end{bmatrix}, \ \sigma_\mathrm{s} = \begin{bmatrix} 1000 \\ 0 \end{bmatrix},$$

with $\sigma_\mathrm{t} = \sigma_\mathrm{a} + \sigma_\mathrm{s}$. The neutron-induced reaction matrix (also in barns) is:

$$\mathbf{\Sigma} = \begin{bmatrix} -3000 & 0 \\ 3000 & -150 \end{bmatrix}.$$

The initial nuclide density (in atoms/b/cm) is:

$$N_0 = \begin{bmatrix} 0.00025 \\ 0 \end{bmatrix}.$$

ν is fixed at 2.3, as is the fission energy release, κ_f, at 200 MeV. The power per unit area is 10 kW/cm^2. The geometry is 3 m long.

The linear stability properties of the CE/LI with substeps are unextraordinary in this case: they are identical to that of the basic predictor-corrector scheme shown previously (Cosgrove et al., 2020c), i.e., they are independent of the number of substeps. However, this is not true generally, only for the present simple system which does not feature decay.

For schemes using an LE predictor, the formulation of the matrices describing the spatial modes necessitates a doubling of the number of eigenpairs – there are twice as many eigenpairs as nuclides in the depletion system. Obtained from Eq. (68), the first few non-trivial eigenvalues of the LE scheme for this problem are shown in Fig. 1 – the $m = 0$ modes are included for completeness even though they are indeed trivial. Here 5 substeps are used.

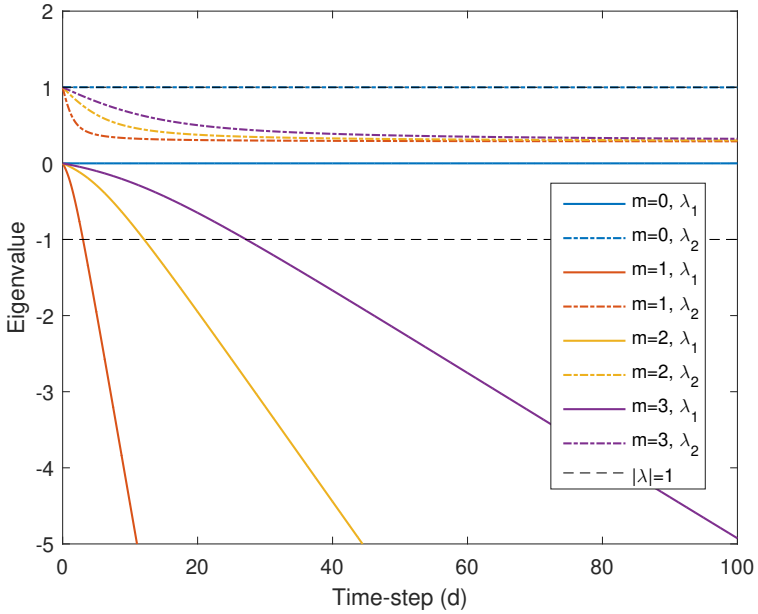


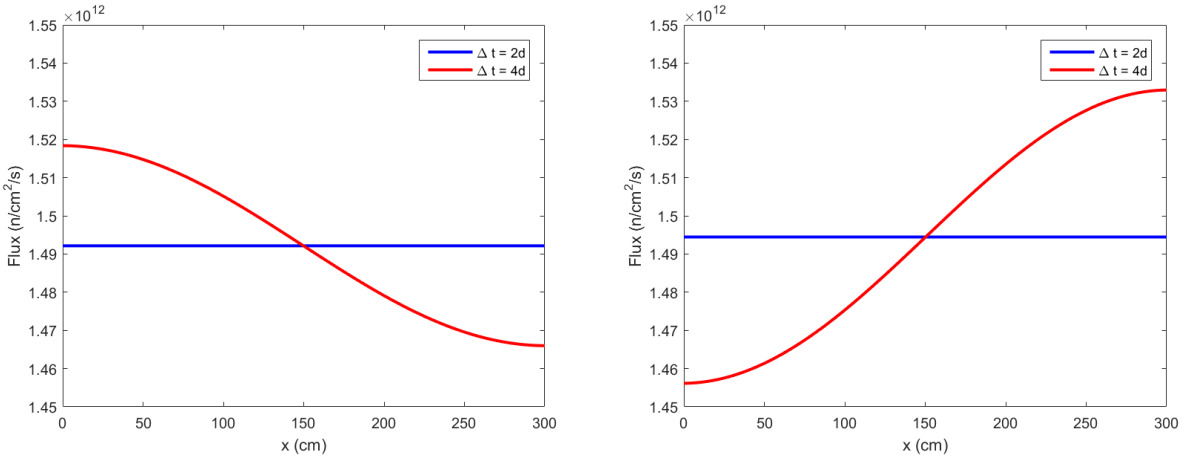
Fig. 1. Eigenvalues of the LE scheme with time using 5 substeps.

As seen in previous work, the first spatial mode is the most unstable. The LE scheme is noticeably less stable than both the explicit Euler and predictor-corrector scheme for this particular problem (Cosgrove et al., 2020c); although the others showed a maximum stable time-step of approximately 5.5 days (before relaxation), the LE scheme using 5 substeps has a stable time-step length of about 3 days. These results appear to be borne out by simulation, as shown in Fig. 2. This apparent relative instability of the LE scheme has been commented on previously by Isotalo et al. (2013) while enforcing xenon equilibrium.

Another interesting property of the LE scheme in this simple case is that, unlike the CE/LI scheme, its stability is sensitive to the number of substeps taken. Increasing the number of substeps decreases the maximum stable time-step length. This is shown in Fig. 3 where the most unstable eigenvalues are shown, each belonging to the first spatial mode. The stability penalty appears to reach a limit as the number of substeps are increased.

Using Eq. (93), the LE/LI scheme is also seen to display relatively complex behaviour on this simple problem: not only are there additional eigenpairs due to the LE predictor, but complex conjugate eigenpairs also occur. Here stability is determined by the magnitude of a complex eigenvalue. Nevertheless, unlike the LE scheme, neither the number of substeps during the predictor nor the corrector affect the maximum stable time-step length. Plots of the absolute eigenvalues of the LE/LI scheme are shown in Fig. 4 when using 2 and 20 substeps on both the predictor and corrector. Applying a relaxation and iteration, however, does change the stable time-step length of the LE/LI scheme – this will be seen in Section 4.2.

Using Eq. (109), the properties of the LE/QI scheme can be interrogated. LE/QI appears to possess a behaviour which had not been observed before: simply using a QI corrector appears to be more stable than an LI corrector. When taking a single substep on the QI



(a) 196 days **(b)** 200 days

Fig. 2. LE flux solutions at consecutive time-points.

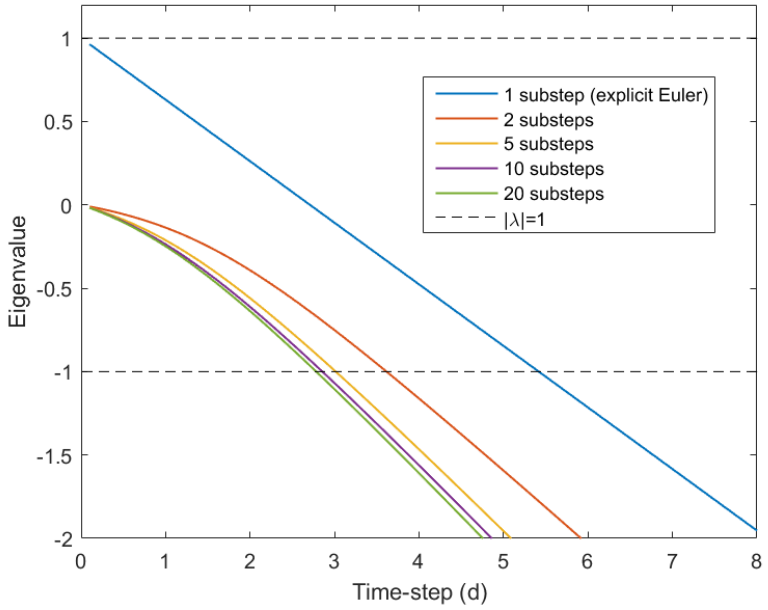
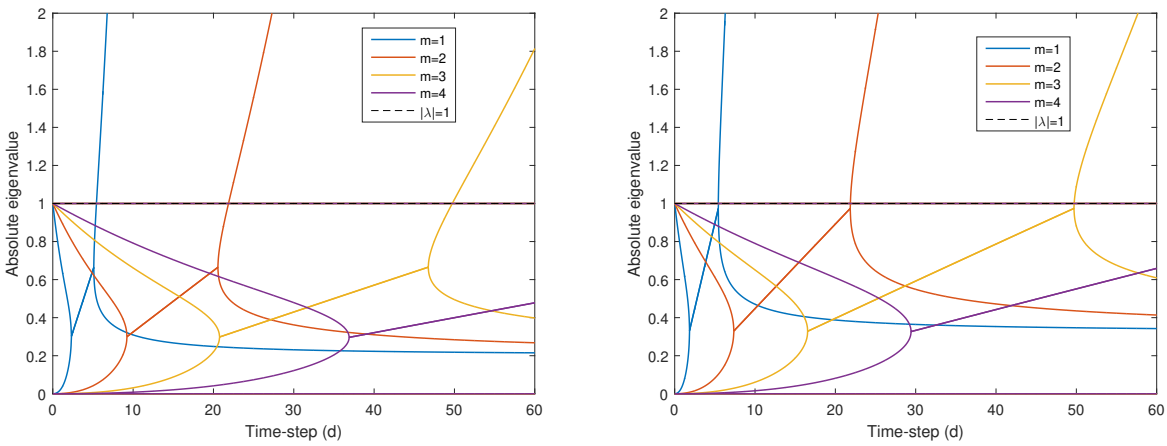


Fig. 3. Most unstable eigenvalues of the LE scheme with time using different numbers of substeps.



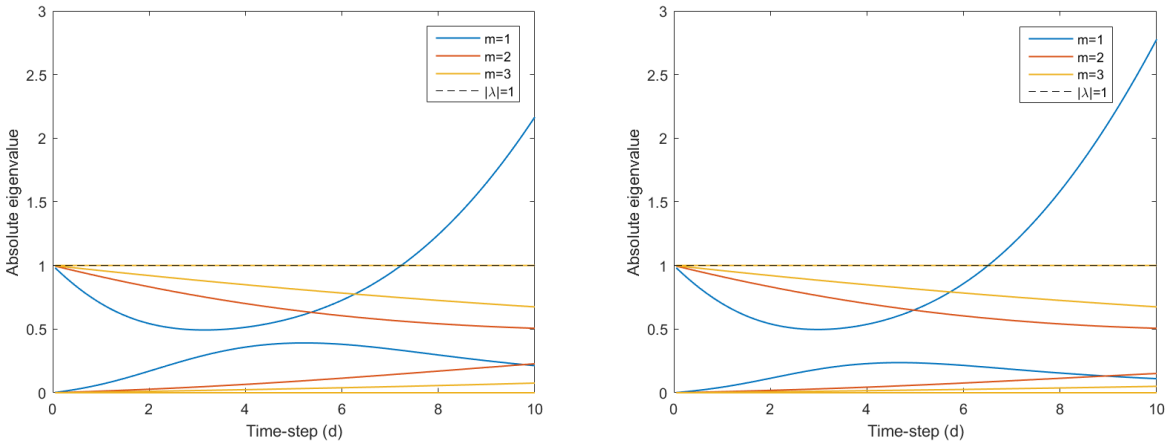
(a) 2 substeps **(b)** 20 substeps

Fig. 4. Absolute eigenvalues of the LE/LI scheme with time when using 2 and 20 substeps on both the predictor and corrector.

corrector, the maximum stable time-step is about 7.25 days for this problem, as opposed to the 5.5 days possessed by the CE/LI, LE/LI and single-substep LE schemes. Furthermore, this does not appear to be affected by the number of LE substeps taken, but it is deteriorated by taking more QI substeps, albeit still apparently limiting to a relatively long stable time-step of about 6.5 days. This is shown in Fig. 5 – as the LE/QI scheme possesses complex eigenvalues when using more than one LE substep, the absolute eigenvalues are plotted. These results appear to agree with simulation – Fig. 6 shows two LE/QI schemes taking time-steps of 7 days, with that using 20 substeps becoming unstable while that using a single substep remains uniform.

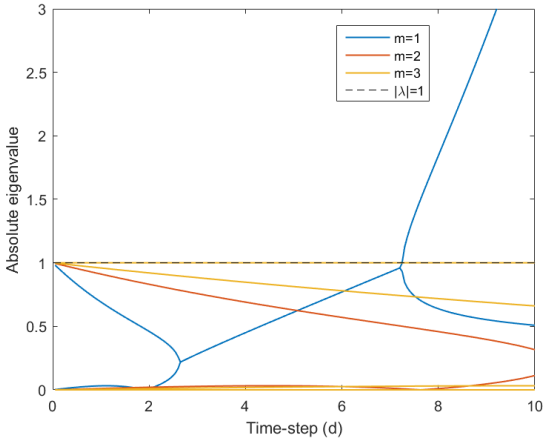
4.2. Eight nuclide system

This section aims to make assertions about the stability of different depletion schemes applied to more realistic problems. A more complicated nuclide system is considered here, aiming to emulate that in a nuclear reactor. The system features eight nuclides which are intended to be identified with uranium-235, uranium-238, neptunium-239, plutonium-239, a lumped fission product, iodine, xenon, and hydrogen. From a simple fresh PWR pin geometry with 5% enrichment, cross sections for each of these isotopes (except for the lumped fission product) were generated using Serpent (Leppänen et al., 2015). The neptunium- and iodine-like nuclides are treated as neutronically inert, existing only to enable the plutonium and xenon decay chains, respectively. Hydrogen is treated as purely scattering – the only scattering nuclide in the system. Uranium-235 capture is neglected. The cross sections of

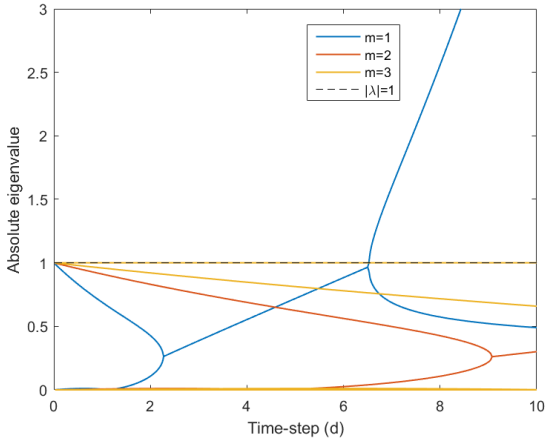


(a) 1 LE substep, 1 QI substep

(b) 1 LE substep, 20 QI substeps



(c) 20 LE substeps, 1 QI substep



(d) 20 LE substeps, 20 QI substeps

Fig. 5. Absolute eigenvalues of the LE/QI scheme with time when using different numbers of substeps on both the predictor and corrector

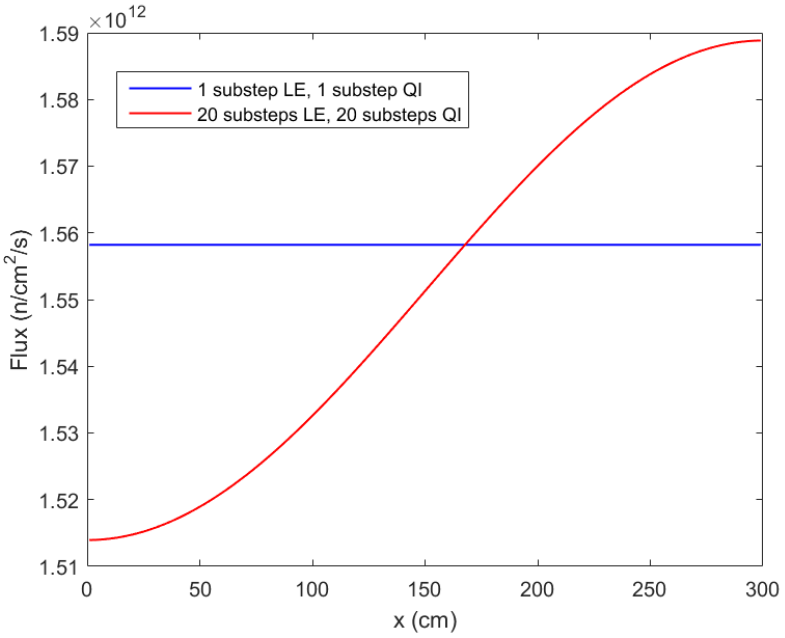


Fig. 6. Flux solutions of the LE/QI scheme after 140 days when taking 7 day time-steps and using differing numbers of substeps.

each, ordered as listed above, are (in barns):

$$\sigma_{\mathrm{a}} = \begin{bmatrix} 31.4 \\ 0.835 \\ 0 \\ 86.5 \\ 50 \\ 0 \\ 128000 \\ 0 \end{bmatrix}, \quad \sigma_{\mathrm{f}} = \begin{bmatrix} 31.4 \\ 0.005 \\ 0 \\ 86.5 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad \sigma_{\mathrm{s}} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 13.5 \end{bmatrix}.$$

The fission yields from any fissile isotope are taken to be identical. The generic fission product has a yield of 90%, iodine has 5%, and xenon has 1%. This gives (in barns):

$$\boldsymbol{\Sigma} = \begin{bmatrix} -31.4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -0.835 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 8.3 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -86.5 & 0 & 0 & 0 & 0 \\ 28.26 & 0.0045 & 0 & 77.85 & -50 & 0 & 0 & 0 \\ 1.57 & 0.00025 & 0 & 4.325 & 0 & 0 & 0 & 0 \\ 0.314 & 0.0005 & 0 & 4.325 & 0 & 0 & -128000 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

Three nuclides undergo decay: the neptunium, iodine, and xenon. Their standard half-lives are used. This gives (in hr^{-1}):

$$\mathbf{\Lambda} = \ln 2 \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1/57.6 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1/57.6 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1/7 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1/7 & -1/9.1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

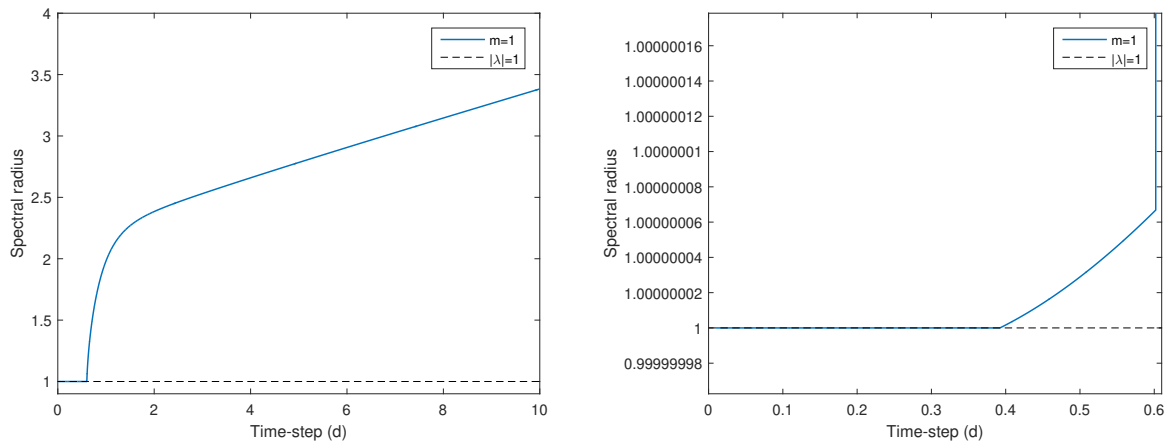
The initial nuclide density corresponds to that of a UO_2 pin cell with radius 0.4095 cm, a cladding outer radius of 0.475 cm, and a pitch of 1.26 cm. The UO_2 density is 10.97 $\mathrm{g/cm}^3$, while the water density is 0.7 $\mathrm{g/cm}^3$. Oxygen and cladding atoms are neglected. This gives (in atoms/b/cm):

$$N_0 = \begin{bmatrix} 0.000394 \\ 0.007389 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0.31296 \end{bmatrix}.$$

The average number of neutrons and energy release per fission are assumed constant across fissile nuclides, at 2.3 and 200 MeV, respectively. The length of the geometry remains the same as before with $L = 3$ m, while the power density is, like a PWR, 104 $\mathrm{W/cm}^3$, giving $P = 104L$ $\mathrm{W/cm}^2$.

As this system is relatively numerically stiff, the simple first-order Taylor expansion of the matrix exponential is inadequate for accurately describing its evolution or stability behaviour. Instead, CRAM is used as the Bateman operator and automatic differentiation is used to calculate the Bateman flux derivative terms described in Section 3. This is done with the ADiGator MATLAB package (Weinstein and Rao, 2016) – ADiGator conveniently generates differentiated Bateman operator functions which can be rapidly evaluated, the only constraints being the need to generate a new function when changing either the number of nuclides in the system or the number of substeps performed. It should also be made clear: CRAM is not usually implemented to construct the matrix exponential directly, but rather to obtain the action of a matrix exponential on a vector. Hence, where a Bateman operator matrix, $\mathbf{B}$, is required, this is generated by having CRAM act on the columns of the identity matrix and concatenating the results.

Due to there now being at least eight eigenvalues per mode, only the spectral radius of a given mode is presented for clarity. One may note that, even when a stable time-step is chosen in the figures to follow, the spectral radius is barely noticeably below a value of



(a) Plot from 0 to 10 days.

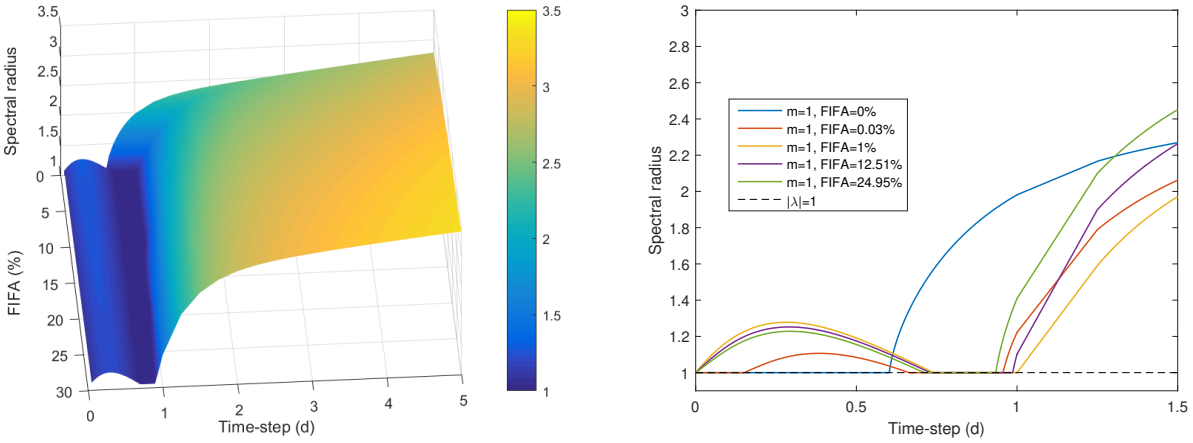
(b) Zoomed-in plot from 0 to 0.6 days.

Fig. 7. Spectral radius of the first mode for the explicit Euler scheme.

1. This is because these eigenvalues correspond to the depletion of fuel, which is a lengthy process in realistic systems, even for large time-steps. However, if one constructed a very stable system, one would notice these eigenvalues fall with longer time-steps, corresponding to greater fuel depletion per step.

First, with this realistic system, it can be immediately noticed that strong non-linearities are present, necessitating some caution in the use of a linear stability analysis. This can be seen by first considering the stability of the standard explicit Euler scheme – the stability spectral radii are obtained using a single substep applied to Eq. (68) and shown in Fig. 7. Higher modes are not shown here for visual clarity and because they are only significant when taking time-steps of about 60 days or more. This figure implies a short maximum time-step, as one might expect given previous reports of instability – in fact, the problem is even less stable than Fig 7(a) might suggest, requiring zooming in to demonstrate that the spectral radius exceeds one at about a time-step of 0.35 days, shown in Fig. 7(b). Below this value the spectral radius is indeed marginally less than one. However, if one simulates this problem with apparently stable time-steps of around 0.3 days or less, a spurious first mode ultimately develops in spite of the numerical prediction.

This predictive failure is attributable to the rapid build-up of strong absorbers, invalidating the linear stability analysis about the chosen value of N_0. To investigate this, the spectral radius of the first spatial mode against time-step was plotted along the additional dimension of burn-up, essentially showing how stability varies with the value of N_0. This was done by simulating the depletion of an identical system with short time-steps but where the composition is forced to be uniform by using only a single burnable/neutronic region, preventing any spatial instabilities. The resulting nuclide density vectors were used to obtain the spectral radius with time-step and the results are shown in Fig. 8. Here FIFA stands for Fissions per Initial Fissile Atom, a measure of burn-up. One can see that, with even a small degree of burn-up, the stability of the system differs substantially from that



(a) Spectral radius of the first mode for the explicit Euler scheme against time-step length and burn-up.

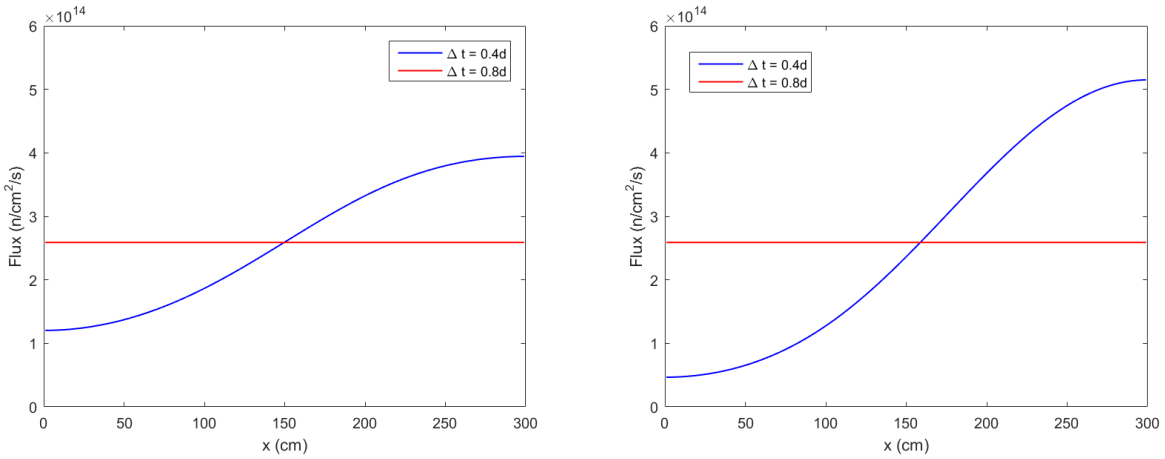
(b) Spectral radius of the first mode for the explicit Euler scheme for several chosen burn-up points.

Fig. 8. Evolving stability of the 8-nuclide problem.

of the un-burned composition in Fig. 7. Reassuringly, however, after this initial burn-up to accumulate fission products, stability properties only evolve slowly with additional burn-up.

To validate this numerically, the system is simulated using the explicit Euler scheme taking time-steps of 0.4 and 0.8 days, as shown in Fig. 9. From Fig. 8, one anticipates that a time-step of 0.4 days should be unstable, while 0.8 days should remain stable, which is observed. To the authors' knowledge, this region of stability when taking a time-step above a few hours and below about a day has not been reported – it may be a phenomenon unique to the constructed depletion system, although it would be intriguing to observe it in realistic systems. In fact, the two unstable regions have two quite different behaviours: that with a shorter time-step acts much like a 'physical' xenon oscillation, not necessarily undergoing a complete phase change each step, while that with a longer time-step does indeed change phase each step. This difference is attributable to the eigenvalues in each region: the shorter-step region has complex eigenvalues, inducing only a partial phase change per step, while the longer-step region possesses real eigenvalues. Slightly different stable zones also exist for different coupling schemes.

This nonlinearity implies that if, say, these stability estimates were to be used to select the time-step for realistic systems, they should be re-evaluated periodically to ensure they are still valid and the chosen time-step remains acceptable. This nonlinear behaviour affecting stability estimates can also be evaded practically by using a slightly different value for N_0 when estimating stability, namely, that corresponding to xenon equilibrium. The justification for this is that xenon is the most prominent absorber which is evolved rapidly at the start of burn-up, but will saturate quickly. Therefore, it is more appropriate to estimate stability based on this saturated value which the problem will quickly reach regardless. The standard equations to set the iodine and xenon densities are given by Eq. (28). These can be used alongside N_0 and the value of ϕ_0 from the power normalisation Eq. (43) to obtain equilibrium iodine and xenon density estimates which will then be inserted into N_0 for eval-



(a) 62.4 days **(b)** 63.2 days

Fig. 9. Explicit Euler flux solutions at consecutive time-points.

uating stability, even if that value of N_0 is not used directly in the simulation. Applying the equilibrium equations to this system gives:

$$N_0 = \begin{bmatrix} 0.000394 \\ 0.007389 \\ 0 \\ 0 \\ 0 \\ 5.8998 \times 10^{-9} \\ 3.5641 \times 10^{-9} \\ 0.31296 \end{bmatrix}.$$

This value is used for all subsequent stability estimates presented. The stability estimate for the first spurious mode of the explicit Euler scheme with time-step subject to xenon equilibrium nuclide densities is shown in Fig. 10. This estimate is substantially closer to the more useful estimates given by partially burned fuel than it is to the fresh fuel, seen in Fig. 8(b).

Finally, the behaviour of implicit schemes is briefly investigated. The authors' previous work implied substantial stability differences between implicit predictor-corrector schemes in a realistic system with and without substeps (Cosgrove et al., 2020a) – this is examined by comparing the stability estimates for the standard predictor-corrector scheme and the LE/LI scheme with 10 substeps on the predictor and corrector when performing multiple iterations, all with a relaxation factor of $\alpha = 0.3$. Their stability is shown in Fig. 11. From the spectral radii, one can see that, as previously reported, the implicit LE/LI scheme requires more aggressive relaxation to improve its stability as compared to the predictor-corrector scheme without substeps – not only is the first mode instability more persistent, but higher modes also. The comparative stability is shown numerically in Fig. 12.

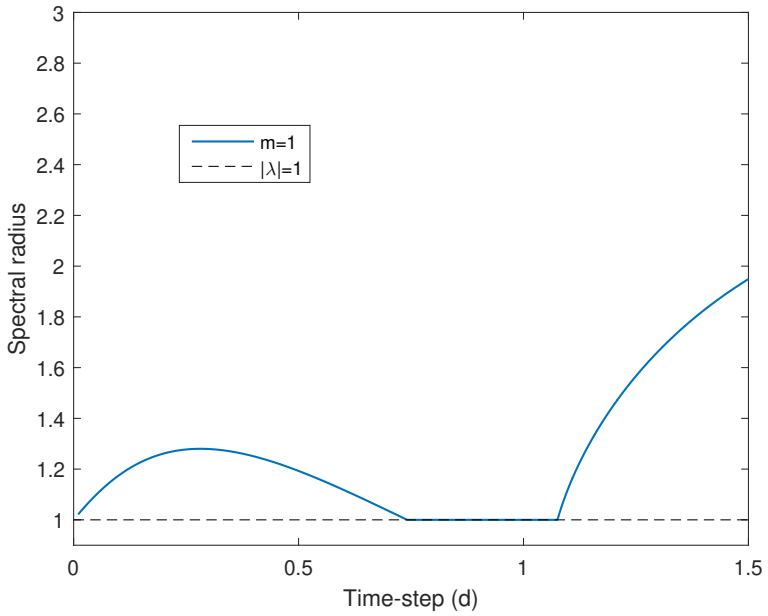
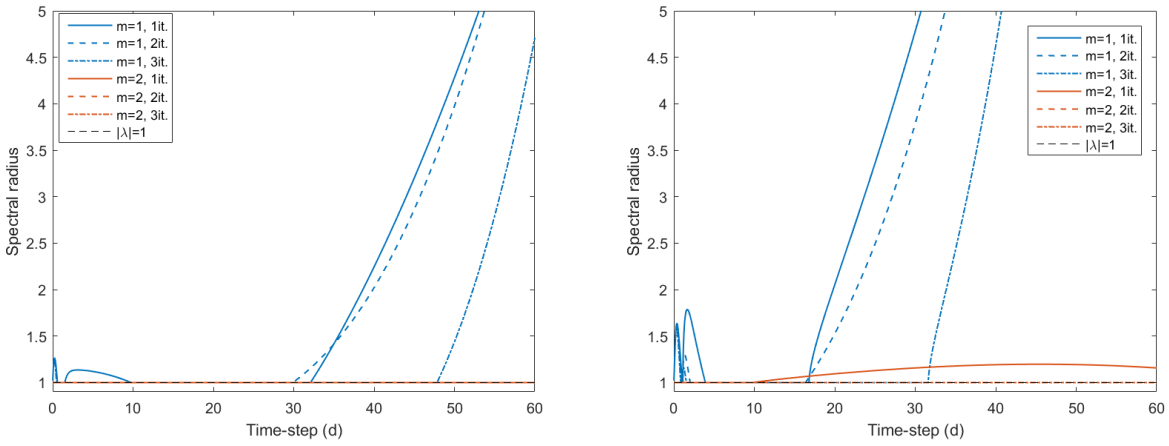


Fig. 10. Stability of the first mode for the explicit Euler scheme estimated with densities corresponding to xenon equilibrium.



(a) Predictor-corrector scheme

(b) LE/LI scheme, 10 substeps on predictor and corrector

Fig. 11. Spectral radii of the implicit predictor-corrector scheme and LE/LI scheme when performing multiple corrector iterations with a relaxation factor of $\alpha = 0.3$.

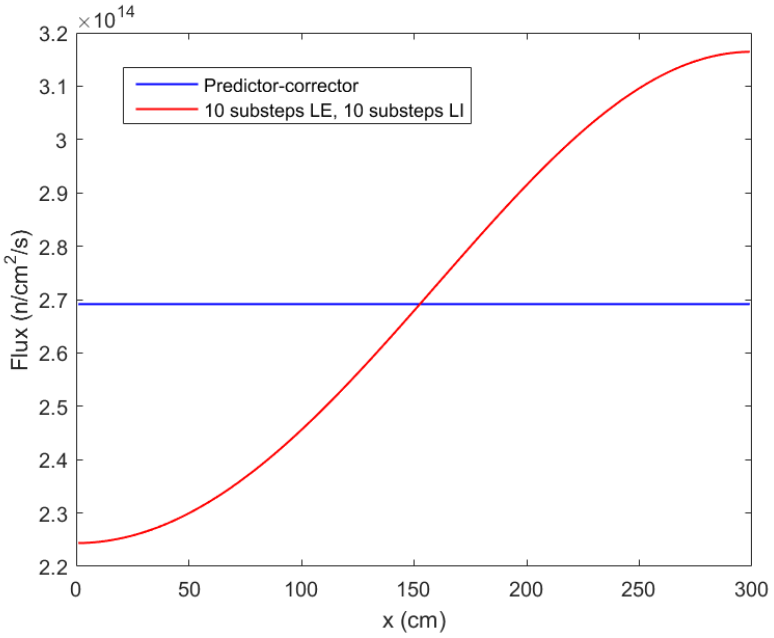


Fig. 12. Flux solutions of the implicit predictor-corrector and LE/LI schemes after 720 days when taking 40 day time-steps, performing three corrector iterations, and applying a relaxation factor of $\alpha = 0.3$.

Notably, in Fig. 11, for both schemes, relaxing and iterating the corrector step struggles to eliminate the short time-step instability entirely – this is less visible for the predictor-corrector scheme, but this instability remains present whether performing 1, 2, or 3 corrector iterations. In fact, this has been observed for a high-fidelity system previously: using a Monte Carlo transport solver, Josey (2017) applied the SIE scheme with 10 iterations to a uniform reflected pin, taking time-steps of 3 hours, with an oscillatory solution developing nevertheless. It may be that short time-step instabilities are not well-handled by current implicit methods.

5. Conclusions

This paper is a further extension to Densmore et al. (2013) and Cosgrove et al. (2020c), accounting for both higher-order neutronics-depletion coupling schemes and more accurate Bateman operators. The analysis reproduces a number of results which have been observed in previous studies featuring more complex depletion systems. Notably, that linear extrapolation has relatively poor stability properties, that higher-order predictor-corrector schemes require more aggressive relaxation, and that short time-step instabilities persist even when using implicit coupling schemes. The paper has also highlighted a deficiency of the stability analysis, namely that non-linear behaviour causes stability estimates to vary with burn-up. However, it appears that this limitation does not invalidate the results of the stability analysis, provided they are applied with some caution.

While this work has brought neutronics-depletion stability analysis closer to application

in realistic reactor problems, additional analysis remains to be done. Effects which may accentuate non-linearities should also be investigated, including the changing neutron spectrum with burn-up affecting both neutronics and depletion, and whether stochastic noise inherent to a Monte Carlo solver leads to different stability properties. Finally, to treat a common approach to burn-up, this work should be extended to quantifying the effects of enforcing xenon equilibrium. This would require considering how the iteration between the neutronics governing equation and the equilibrium xenon and iodine densities affects stability.

6. Data availability statement

To the best of the authors' knowledge, this paper and references herein contain all the data needed to reproduce and validate the results presented.

Declarations

Declarations of interest: none.

References

Adamowicz, N., Cosgrove, P., 2021. Recent advances in the stability analysis of burn-up calculations, in: Proc. M&C 2021, Raleigh, North Carolina.

Bell, G., Glasstone, S., 1970. Space-dependent reactor dynamics and related topics: Burnup Problems, in: Nuclear Reactor Theory. US Atomic Energy Commission, Washington, DC. chapter 10.2.

Cetnar, J., 2006. General solution of Bateman equations for nuclear transmutations. Annals of Nuclear Energy 33, 640–645. doi:10.1016/j.anucene.2006.02.004.

Cosgrove, P., Adamowicz, N., 2021. Stability analysis of spatial discretisation in burn-up calculations, in: Proc. M&C 2021, Raleigh, North Carolina.

Cosgrove, P., Shwageraus, E., Parks, G.T., 2020a. A simple implicit coupling scheme for Monte Carlo neutronics and isotopic depletion. Annals of Nuclear Energy 141. doi:10.1016/j.anucene.2020.107374.

Cosgrove, P., Shwageraus, E., Parks, G.T., 2020b. Neutron clustering as a driver of Monte Carlo burn-up instability. Annals of Nuclear Energy 137. doi:10.1016/j.anucene.2019.106991.

Cosgrove, P., Shwageraus, E., Parks, G.T., 2020c. Stability analysis of predictor-corrector schemes for coupling neutronics and depletion. Annals of Nuclear Energy 149. doi:10.1016/j.anucene.2020.107781.

Densmore, J.D., Gill, D.F., Griesheimer, D.P., 2013. Stability analysis of burnup calculations. Transactions of the American Nuclear Society 109, 695–698. doi:10.1007/s10967-012-2210-3.2.

Duderstadt, J.J., Hamilton, L.J., 1976. Nuclear Reactor Analysis. John Wiley and Sons, Inc., New York.

Dufek, J., Hoogenboom, J.E., 2009. Numerical stability of existing Monte Carlo burnup codes in cycle calculations of critical reactors. Nuclear Science and Engineering 162, 307–311. doi:10.13182/NSE08-69TN.

Dufek, J., Kotlyar, D., Shwageraus, E., 2013. The stochastic implicit Euler method: A stable coupling scheme for Monte Carlo burnup calculations. Annals of Nuclear Energy 60, 295–300. doi:10.1016/j.anucene.2013.05.015.

Fung, T.C., 2004. Computation of the matrix exponential and its derivatives by scaling and squaring. International Journal for Numerical Methods in Engineering 59, 1273–1286. doi:10.1002/nme.909.

Gebremedhin, A.H., Walther, A., 2020. An introduction to algorithmic differentiation. doi:10.1002/widm.1334.

Griesheimer, D.P., 2010. In-line xenon convergence algorithm for Monte Carlo reactor calculations, in: Proc. PHYSOR 2010, Pittsburgh, Pennsylvania.

Griesheimer, D.P., Carpenter, D.C., Stedry, M.H., 2017. Practical techniques for large-scale Monte Carlo reactor depletion calculatons[sic]. Progress in Nuclear Energy 101, 409–423. doi:10.1016/j.pnucene.2017.05.018.

Isotalo, A., Aarnio, P., 2011a. Higher order methods for burnup calculations with Bateman solutions. Annals of Nuclear Energy 38, 1987–1995. doi:10.1016/j.anucene.2011.04.022.

Isotalo, A., Aarnio, P., 2011b. Substep methods for burnup calculations with Bateman solutions. Annals of Nuclear Energy 38, 2509–2514. doi:10.1016/j.anucene.2011.07.012.

Isotalo, A., Pusa, M., 2016. Improving the accuracy of the Chebyshev rational approximation method using substeps. Nuclear Science and Engineering 183, 65–77. doi:10.13182/NSE15-67.

Isotalo, A., Sahlberg, V., 2015. Comparison of Neutronics-Depletion Coupling Schemes for Burnup Calculations. Nuclear Science and Engineering 179, 434–459. doi:10.13182/NSE14-35.

Isotalo, A.E., Leppänen, J., Dufek, J., 2013. Preventing xenon oscillations in Monte Carlo burnup calculations by enforcing equilibrium xenon distribution. Annals of Nuclear Energy 60, 78–85. doi:10.1016/j.anucene.2013.04.031.

Josey, C., 2017. Development and Analysis of High Order Neutron Transport-Depletion Coupling Algorithms. Ph.D. thesis. Massachusetts Institute of Technology.

Keady, K.P., Larsen, E.W., 2016. Stability of Monte Carlo k-eigenvalue simulations with CMFD feedback. Journal of Computational Physics 321, 947–964. doi:10.1016/j.jcp.2016.06.002.

Kotlyar, D., Shwageraus, E., 2013. On the use of predictor-corrector method for coupled Monte Carlo burnup codes. Annals of Nuclear Energy 58, 228–237. doi:10.1016/j.anucene.2013.03.034.

Kotlyar, D., Shwageraus, E., 2014. Numerically stable Monte Carlo-burnup-thermal hydraulic coupling schemes. Annals of Nuclear Energy 63, 371–381. doi:10.1016/j.anucene.2013.08.016.

Kotlyar, D., Shwageraus, E., 2016. Stochastic semi-implicit substep method for coupled depletion Monte-Carlo codes. Annals of Nuclear Energy 92, 52–60. doi:10.1016/j.anucene.2016.01.022.

Leppänen, J., Pusa, M., Viitanen, T., Valtavirta, V., Kaltiaisenaho, T., 2015. The Serpent Monte Carlo code: Status, development and applications in 2013. Annals of Nuclear Energy 82, 142–150. doi:10.1016/j.anucene.2014.08.024.

Martins, J.R., Sturdza, P., Alonso, J.J., 2003. The complex-step derivative approximation. ACM Transactions on Mathematical Software 29, 245–262. doi:10.1145/838250.838251.

Moler, C., Van Loan, C., 2003. Nineteen Dubious Ways to Compute the Exponential of a Matrix, Twenty-Five Years Later. SIAM Review 45, 3–49. doi:10.1137/S00361445024180.

Najfeld, I., Havel, T.F., 1995. Derivatives of the matrix exponential and their computation. Advances in Applied Mathematics 16, 321–375. doi:10.1006/aama.1995.1017.

Pusa, M., 2011. Rational approximations to the matrix exponential in burnup calculations. Nuclear Science and Engineering 169, 155–167. doi:10.13182/NSE10-81.

Pusa, M., 2013. Numerical Methods for Nuclear Fuel Burnup Calculations. Ph.D. thesis. VTT Technical Research Centre of Finland.

Pusa, M., 2016. Higher-order Chebyshev rational approximation method and application to burnup equations. Nuclear Science and Engineering 182, 297–318. doi:10.13182/NSE15-26.

Pusa, M., Leppänen, J., 2010. Computing the matrix exponential in burnup calculations. Nuclear Science and Engineering 164, 140–150. doi:10.13182/NSE09-14.

Robbins, H., Monro, S., 1951. A Stochastic Approximation Method. The Annals of Mathematical Statistics 22, 400–407. doi:10.1214/aoms/1177729586.

Romano, P.K., Horelik, N.E., Herman, B.R., Nelson, A.G., Forget, B., Smith, K., 2015. OpenMC: A state-of-the-art Monte Carlo code for research and development. Annals of Nuclear Energy 82, 90–97. doi:10.1016/j.anucene.2014.07.048.

Romano, P.K., Josey, C.J., Johnson, A.E., Liang, J., 2021. Depletion capabilities in the OpenMC Monte Carlo particle transport code. Annals of Nuclear Energy 152.

Tuominen, R., 2015. Coupling Serpent and OpenFOAM for neutronics - CFD multi-physics calculations.

Master's thesis. Aalto University.
Valtavirta, V., Leppänen, J., 2018. New stochastic substep based burnup scheme for Serpent 2, in: Proc. PHYSOR 2018, Cancún, Mexico.
Weinstein, M.J., Rao, A.V., 2016. A source transformation via operator overloading method for the automatic differentiation of mathematical functions in MATLAB. ACM Transactions on Mathematical Software 42, 1–44. doi:10.1145/2699456.