POLITECNICO DI TORINO
Repository ISTITUZIONALE

A Monte Carlo Implementation of the zeta-Eigenvalue Equation

Original
A Monte Carlo Implementation of the zeta-Eigenvalue Equation / Raffuzzi, V., Abrate, N., Dulla, S.. - ELETTRONICO. - (2025), pp. 1198-1207. (International Conference on Mathematics and Computational Methods Applied to Nuclear Science and Engineering M&C 2025 Denver (USA) 27-30 Aprile 2025).

Availability:
This version is available at: 11583/3004462 since: 2025-10-25T12:23:47Z

Publisher:
Taylor and Francis

Published
DOI:

Terms of use:

This article is made available under terms and conditions as specified in the corresponding bibliographic description in the repository

Publisher copyright
Taylor and Francis postprint/Author's Accepted Manuscript

(Article begins on next page)

28 November 2026

A Monte Carlo Implementation of the ζ-Eigenvalue Equation

Valeria Raffuzzi [1,*], Nicolò Abrate[2,3], Sandra Dulla[2,3]

[1]Department of Engineering, University of Cambridge, UK;
[2]NEMO Group, Energy Department, Politecnico di Torino, Italy;
[3]I.N.F.N., Sezione di Genova, Via Dodecaneso, 33 - 16146 Genova, Italy

[leave space for DOI, which will be inserted by ANS]

ABSTRACT

The ζ-eigenvalue equation is a formulation of the neutron transport equation which ensures criticality by scaling the density of the nuclides or materials of choice with the parameter ζ. As a consequence, the value of ζ provides quantitative information about what material density or nuclide concentration can make a system critical. This paper proposes a Monte Carlo algorithm to solve the ζ-eigenvalue equation. The method, based on a fixed point iteration scheme, is implemented in the code SCONE and tested on some cases of practical utility. Examples include the search of critical gadolinium concentration in a PWR assembly, and the search of critical boron in BEAVRS. The Monte Carlo implementation is successful at finding the critical densities requested, and in the cases tested the runtime of a ζ calculation is generally comparable to that of a k-eigenvalue calculation.

Keywords: ζ-eigenvalue, Monte Carlo, Critical search

1. INTRODUCTION

Several eigenvalue formulations exist that allow one to solve the neutron transport equation for a fundamental mode. The most common is the k-eigenvalue formulation, where the neutron multiplication factor k_{eff} is used to scale the fission source. Alternatively, the α-eigenvalue, where α is the fundamental time-decay constant of the system, is used to capture time-dependent effects [1]; Davison [2] proposed the c-eigenvalue (also indicated as γ), used as a scaling factor of the fission and scattering source; Ronen [3] introduced the δ-eigenvalue, which acts as a density scaling for all the materials in the system. While all of these formulations are useful for some applications, when applied to reactor design, they all fall short of providing the designer with quantitative information on how to make the system critical.

More recently, Abrate [4] proposed the ζ-eigenvalue formulation. Similarly to δ, ζ is used as a density scaling parameter. However, instead of acting on all the materials in the system, it only acts on the materials or nuclides of choice: this allows for devising applications of practical interest for nuclear engineers. For example, the concentration of boron dissolved in a Pressurised Water Reactor (PWR) coolant that makes the reactor critical can be obtained from solving a ζ-eigenvalue equation. The same applies to the concentration of cadmium to be added to the control rods of a reactor, or burnable poisons in the fuel. This is in contrast to using an iterative search method, which normally consists of repeating a k-eigenvalue calculation multiple times and might fail to converge [5].

In [4], the ζ-eigenvalue formulation was implemented in a deterministic test platform, and applied to deal with simple one-dimensional problems. In this work, a Monte Carlo algorithm to solve the ζ-eigenvalue

*vr339@cam.ac.uk

equation is devised and implemented in the Monte Carlo code SCONE [6], under development at the University of Cambridge. Then, the method is tested on some realistic numerical examples. It must be noticed that a somewhat similar method is implemented in the Monte Carlo code Serpent [7]. Serpent allows the option to perform a critical density iteration on the nuclides of choice, that produces a density scaling factor; this can be applied only to reasonably strong absorbers. Despite the similarities, the method proposed here is more general and assumption-free. The rest of the paper proceeds as follow: Section 2 describes the theory of the ζ-eigenvalue equation and the Monte Carlo algorithm implemented; Section 3 presents the results of the numerical tests; Section 4 provides a summary and outlines avenues of future work.

2. THEORY AND METHODOLOGY

2.1. Theory

A generic, compact formulation of the neutron transport equation is given in Eq. (1):

$$\frac{1}{v}\frac{d\psi}{dt} + (L+T)\psi = (F+S)\psi, \tag{1}$$

where ψ is the angular neutron flux, t is time, v is velocity, and L, T, F and S are the leakage, collision, fission and scattering operators, respectively. In order to solve this equation for the fundamental mode of the angular flux, it is normally posed as an eigenvalue equation. The most common formulation is the k-eigenvalue equation, where the neutron multiplication factor k is introduced to scale the fission source in order to maintain the system critical:

$$(L+T-S)\psi_k = \frac{1}{k}F\psi_k. \tag{2}$$

As mentioned in Section 1, the alternative ζ-eigenvalue equation ensures criticality by scaling the density of a nuclide, ensemble of nuclides, or material m chosen by the user. Naturally, the contributions of m and everything else, referred to as $\overline{m}$, have to be explicitly split as in Eq. (3).

$$(L + T_{\overline{m}} + \frac{1}{\zeta}T_m)\psi_\zeta = (F_{\overline{m}} + \frac{1}{\zeta}F_m + S_{\overline{m}} + \frac{1}{\zeta}S_m)\psi_\zeta \tag{3}$$

Rearranging the terms to write Eq. (3) as an eigenvalue equation, one gets:

$$(L + T_{\overline{m}} - F_{\overline{m}} - S_{\overline{m}})\psi_\zeta = \frac{1}{\zeta}(F_m + S_m - T_m)\psi_\zeta. \tag{4}$$

If one were to solve this problem by power iteration as is commonly done, the term in brackets on the right hand side of Eq. (4) (for brevity, B_m) should be the source to be produced each iteration for the next; the operator in brackets on the left hand side (for brevity, $B_{\overline{m}}$) should be inverted to calculate the estimate of the flux for the following iteration. This is described by Eq. (5).

$$\psi_\zeta^{(n+1)} = \frac{1}{\zeta^{(n)}}B_{\overline{m}}^{-1}B_m\psi_\zeta^{(n)} \tag{5}$$

However, there is no guarantee that the operators B_m and $B_{\overline{m}}$ are positive: this depends on the properties of the system modelled, such as the cross sections, geometry and boundary conditions. For example, if m

is a non-fissile nuclide, B_m will always be negative. Negative operators are difficult to interpret in a Monte Carlo framework, hence using the conventional power iteration to solve the ζ-eigenvalue equation in Monte Carlo was not pursued in this work.

To overcome this issue, Eq. (3) can be solved as a fixed point iteration, as shown in Eq. (6), with initial guesses $\psi_\zeta^{(0)} = \psi_{\zeta,0}$ and $\zeta^{(0)} = \zeta_0$. Then, ζ is updated as shown in Eq. (7), where the integrals are over parts of the phase-space P: the domain V is divided into V_m and $V_{\overline{m}}$, which are respectively the regions where m is contained and everywhere else. It must be noticed that in this process, the scattering operators $S_{\overline{m}}$ and S_m have been moved to the left hand side of the equation and are not subtracted to the total collision operators. This was done to maximise similarities between the Monte Carlo implementation of Eq. (6) and the classical k-eigenvalue power iteration algorithm. The details of the Monte Carlo algorithm adopted to solve this equation are explained in the next section.

$$\psi_\zeta^{(n+1)} = (L + T_{\overline{m}} + \frac{1}{\zeta^{(n)}} T_m - S_{\overline{m}} - \frac{1}{\zeta^{(n)}} S_m)^{-1} (F_{\overline{m}} + \frac{1}{\zeta^{(n)}} F_m) \psi_\zeta^{(n)} \tag{6}$$

$$\zeta^{(n+1)} = \frac{\int_{V_m,\Omega,E} (F_m + S_m - T_m)\psi^{(n)} \mathrm{dP}}{\int_{V,\Omega,E} L\psi^{(n)} \mathrm{dP} + \int_{V_{\overline{m}},\Omega,E} (T_{\overline{m}} - F_{\overline{m}} - S_{\overline{m}})\psi^{(n)} \mathrm{dP}} \tag{7}$$

Contrarily to the k-eigenvalue equation, where the fission source is rescaled to ensure criticality, the ζ-eigenvalue equation is not expected to be biased. The fundamental mode of the converged system is expected to be exactly that of the system with scaled atomic densities of m, which should be critical.

2.2. Monte Carlo Implementation

The Monte Carlo code used for this work is SCONE [6]. SCONE has the capability of solving fixed source problems, the k-eigenvalue equation, the $\alpha - k$ or α-static equation [8], and the c-eigenvalue equation [9]. For this work, the fixed point iteration scheme described in Eqs. (6) and (7) was implemented.

When writing the fixed point iteration scheme needed to solve the ζ-eigenvalue equation as in Eq. (6), the Monte Carlo algorithm is almost identical to the power iteration used to solve the k-eigenvalue equation. In fact, the neutron source to be used in each iteration is simply the fission source calculated in the previous iteration. This approach requires minimal code modifications.

The main steps of the ζ calculation are described in the following:

- Define the problem by specifying which nuclides one wants to rescale the density of. These can be one or more nuclides belonging to one or more materials. However, the scaling factor is common to them all, since it is the global factor $\frac{1}{\zeta}$;
- Provide initial guesses for $\psi_{\zeta,0}$ and ζ_0. In SCONE, by default, the initial guess for ζ_0 is 1;
- Perform transport as usual, by saving the fission neutrons produced as the source of the successive iteration (n+1). Note that in this case, the number of fission neutrons produced is not scaled by k_{eff}. Every time macroscopic cross sections are looked-up, the atomic densities of the nuclides which belong to m are scaled by a factor of $\frac{1}{\zeta^{(n)}}$;
- During transport, tally the separate reaction rate contributions of m and $\overline{m}$ and leakage, in order to estimate ζ as shown in Eq. (7). In SCONE, the collision flux estimator is used for this purpose, and an analogue estimator is used for leakage.
- The calculation should converge to the fundamental ζ and fission source during the inactive cycles, and conclude at the end of the active cycles. While ζ converges to its critical value, k_{eff}, which can

be estimated in parallel with the usual implicit estimator shown in Eq. (8), should converge to 1. It should be noticed that k_{eff} can be calculated during a ζ simulation to obtain additional information, but it is not used.

$$k_{\text{eff}} = \frac{\int_{V,\Omega,E} F\psi \mathrm{dP}}{\int_{V,\Omega,E} (L + T - S)\psi \mathrm{dP}} \tag{8}$$

While the Monte Carlo algorithm is easy to adapt from k to ζ calculations, there are some subtleties that must be reported. For a start, the problem must be well posed in order to have a solution: ζ must have a positive value to be physically meaningful. It is clear from the estimator definition in Eq. (7), that this might not always be the case. Physically, ζ represents the ratio between the net neutron production in m over the net neutron losses due to leakage and $\overline{m}$. In the case, for example, of a boron search in a PWR, the numerator of Eq. (7) would be negative since boron is a net absorber. If the neutron losses in the rest of the system were more prominent than neutron production, the denominator would be positive and ζ negative. This example represents the case of a reactor which is subcritical in the absence of boron: there is no amount of boron that could make the reactor critical. Care must be taken when defining the problem, to ensure that the search is physically meaningful.

From an implementation point of view, it must be noticed that certain standard acceleration techniques, such as precalculating the majorant cross section at the beginning of the calculation when using delta tracking, cannot be used naively. This is the case because the material cross sections depend on ζ, and change at each iteration. This implies that the majorant cross section should be recalculated at the beginning of each iteration. To circumvent this issue, all the models tested in this work were run with surface tracking except for BEAVRS. Fortunately, when simulating a large number of particles, recalculating the majorant at the beginning of each cycle is a negligible overhead.

Despite the similarities with the k-eigenvalue power iteration, the two algorithms do not necessarily converge at the same rate. In the following numerical examples, some cases were slow to converge. In SCONE, during transport, it is conventional to use the cumulative estimate of the eigenvalue at the end of the previous iteration, rather than the cycle-wise one. This is the case also during the inactive cycles, i.e., during convergence. Clearly, the cumulative estimate of the eigenvalue is affected by the initial guess, and it takes longer to converge to the fundamental eigenvalue than the cycle-wise one. In order to accelerate convergence, tally flushing was implemented. This consists of resetting the cumulative estimate of the eigenvalue every fixed number of cycles, chosen by the user.

3. NUMERICAL RESULTS

In this section, the methodology presented in Section 2 is applied to some test cases. All the calculations presented in the following use the JEFF-3.1.1 cross sections library, and are run on 100 cores of an AMD Ryzen Threadripper PRO 5995WX node.

3.1. The POPSY benchmark

The first test case was the critical benchmark POPSY. This consists of a sphere of plutonium surrounded by a uranium reflector (the exact benchmark specifications can be found in [10]). Here, the atom density of ^{239}Pu was initially set to 0.03 atoms/barn/cm, and the ζ algorithm was used to find the critical density of ^{239}Pu. It must be emphasised that the model should converge to the correct critical density regardless of the initial density specified; however, the choice of initial density might affect the number of iterations required to converge. The initial configuration has a k_{eff} of 0.87964 ± 6 pcm. For the ζ calculation, a neutron

population of 50,000 neutron per cycle was run, with 200 inactive cycles and 5,000 active cycles. While the problem converges to the correct eigenvalue and fission source relatively quickly, such a large number of active cycles was necessary to obtain an acceptable uncertainty on the value of ζ. The ζ calculation took around 3.5 minutes to complete, and the results are shown in Table I. The uncertainty in the ^{239}Pu critical atomic density was propagated from the uncertainty in ζ. k_{eff} was also estimated during the ζ simulation. In this case, the standard deviation on ζ is larger than that on k_{eff}. This is the case because k_{eff} is an integral parameter, while ζ is affected by the distinction between m and $\overline{m}$: if, for example, m was localised and had a relatively low reaction rate, the uncertainty of ζ could be large.

Table I. ζ calculation results for a critical ^{239}Pu search in POPSY

ζ	k_{eff}	critical density [atoms/barn/cm]
$0.82225 \pm 12 \times 10^{-5}$	0.99994 ± 8 pcm	$3.648\,53 \times 10^{-2} \pm 5.3 \times 10^{-6}$

When performing a k-eigenvalue calculation with the density found by the ζ calculation, the system resulted critical with a k_{eff} of 0.99998 ± 7 pcm. It was confirmed that the energy spectra in the fuel and reflector regions also match within uncertainties between the two calculation types. The reference k-eigenvalue calculation was run with the same neutron population and settings, and it took around 2.5 minutes to run.

3.2. 2D PWR Pincell

A standard 2D PWR pincell with 3.5w% enriched UO_2 and initial water density of $1.0\,g/cm^3$ (corresponding to atomic densities of 6.68×10^{-3} and 3.34×10^{-3} atoms/barn/cm for respectively ^{1}H and ^{16}O) was also simulated. In this case, the water density was adjusted to obtain criticality.

As known from reactor physics, this problem should have two solutions: the PWR lattice will be critical in an under-moderated state and in an over-moderated state. However, the Monte Carlo ζ algorithm can converge only to a solution, equivalent to the over-moderated scenario in this case. This is the case almost regardless of the initial guess ζ_0 and of the initial water density. The only exception is when the choice of initial water density scaled by $\frac{1}{\zeta_0}$ results in a system which is subcritical and under-moderated. In that situation, the estimate of ζ results in a negative value because both water and fuel are net absorbers: this results in Eq. (7) returning a negative value. Then, the calculation is interrupted. It must be noticed that this does not happen when the system is subcritical and over-moderated: in that case, water is a net absorber but the fuel is a net producer, hence Eq. (7) returns a positive value. Future work will focus on alternative formulations of the ζ equation, that might allow converging on the under-moderated solution.

k_{inf} as a function of the moderator-to-fuel ratio is shown in Fig. 1, together with the ζ solution; the curve is produced by a sweep of k-eigenvalue calculations with varied water density.

3.3. 3D PWR Assembly

A 3D PWR assembly was also modelled. Its radial cross section is shown in Fig. 2; axially, the assembly was limited to 100 cm for computational efficiency. For this model, the search consisted in finding the critical densities of the six gadolinium isotopes listed in Table II. Their corresponding initial atomic density values are also reported. The fuel pins containing gadolinium are in bright green in Fig. 2. The k_{eff} of the model with the starting gadolinium densities is 1.08718 ± 8 pcm.

The ζ calculation was performed using 1 million particles per cycles, 100 inactive cycles and 100 active cycles, and it took around 5 minutes to complete. It produced a ζ value of $9.407 \times 10^{-2} \pm 4.4 \times 10^{-5}$ and a k_{eff} of 0.99999 ± 9 pcm. The resulting gadolinium critical densities are listed in Table II with uncertainties

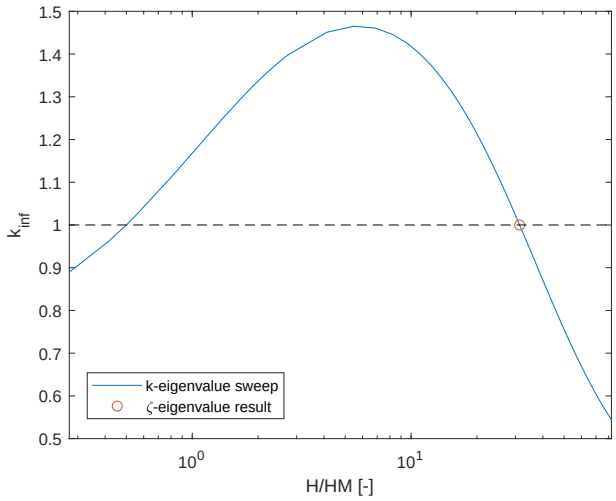

Figure 1. k_{inf} as a function of the moderator-to-fuel ratio, calculated as number of hydrogen atoms over number of heavy metal atoms in the system, for the 2D PWR pincell.

propagated from the ζ value.

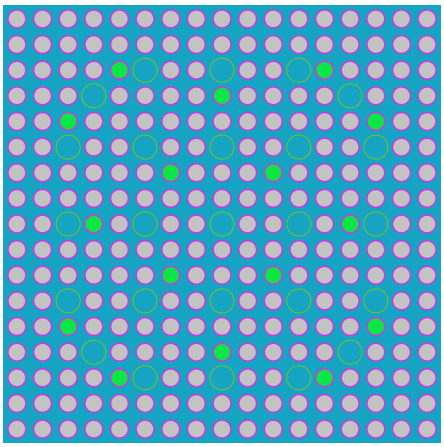

Figure 2. Radial geometry cross section of the 3D PWR assembly model

The cycle-wise values of ζ, k_{eff}, and Shannon entropy produced in the ζ calculation during the inactive cycles are plotted in Fig. 3. The Shannon entropy is calculated in two separate ways: using a purely axial discretisation with 20 axial bins, and a radial pin-by-pin mesh. Here, tally flushing was applied to ζ every 10 cycles, in order to facilitate convergence. The effect of this feature can be observed by the steep drops in ζ and k_{eff} that occur as soon as the cumulative value of ζ, used to scale the densities during transport, is reset.

The Shannon entropy plots confirm that the converge of ζ calculations can be extremely different from that of k-eigenvalue calculations. For instance, similar 3D PWR assembly problems were shown in the past to have a relatively slow convergence for the axial fission source, while the radial fission source converges immediately [11]. In this case, the axial fission source converges relatively quickly, since the gadolinium

Table II. Gadolinium isotopes used for the critical search in the 3D PWR model, and their starting and critical densities obtained by the ζ calculation. The densities are in [atoms/barn/cm]

Isotope	Starting density	Critical density
^{154}Gd	4.6173×10^{-5}	$4.9084 \times 10^{-4} \pm 1.7 \times 10^{-9}$
^{155}Gd	2.9711×10^{-4}	$3.1584 \times 10^{-3} \pm 1.3 \times 10^{-8}$
^{156}Gd	4.1355×10^{-4}	$4.3962 \times 10^{-3} \pm 1.8 \times 10^{-8}$
^{157}Gd	3.1518×10^{-4}	$3.3505 \times 10^{-3} \pm 1.4 \times 10^{-8}$
^{158}Gd	4.9786×10^{-4}	$5.2924 \times 10^{-3} \pm 2.2 \times 10^{-8}$
^{160}Gd	4.3764×10^{-4}	$4.639\,93 \times 10^{-3} \pm 1.9 \times 10^{-8}$

concentration is axially uniform. On the other hand, the radial fission source converges at the same rate as ζ and k_{eff}, as it is strongly affected by the local changes in gadolinium density.

Similarly to what observed in the other cases, the final relative error in ζ is larger than that in k_{eff}. However, during convergence, k_{eff} seems to be subject to large cycle-to-cycle fluctuations. Since ζ is scaling the density of gadolinium, which is a strong neutron poison, the reason might be that a small fluctuation in ζ produces a large fluctuation in k_{eff}. Thoroughly studying the convergence properties of ζ is outside the scope of this paper, but will be the subject of future work.

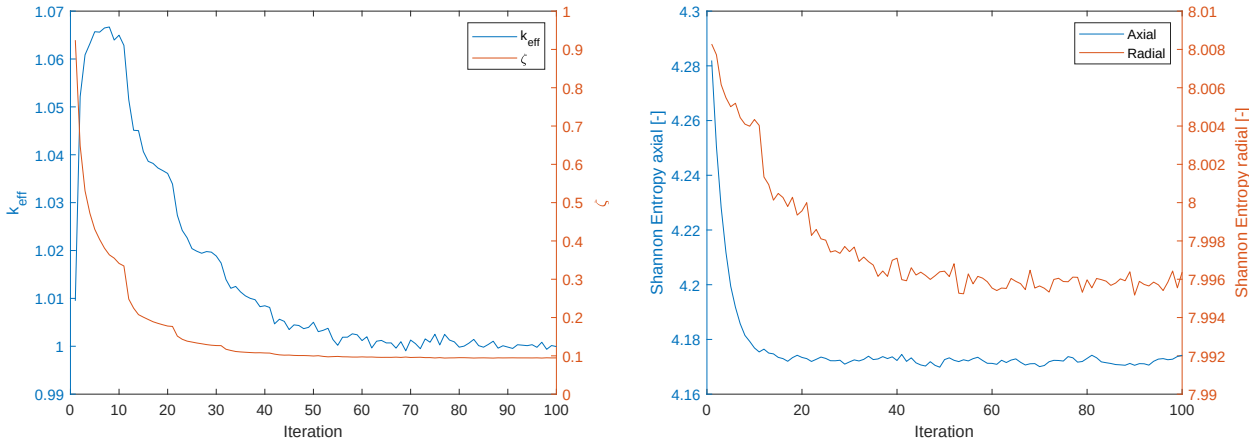

Figure 3. Convergence of the 3D PWR model: k_{eff} and ζ (left) and Shannon entropy (right) as a function of the inactive cycle number

For this model, a simple search using the secant method was performed. Despite more efficient search methods exist in the literature and are applied for similar purposes [12], this simple study was meant to investigate how many iterations are required to converge when performing a naive search. A rigorous performance comparison is outside the scope of this paper. The target value of the secant search performed was generously set to a k_{eff} of 1.0 ± 20 pcm. The secant method requires two initial density scaling values: these were set to 1 and 20, corresponding to ζ values of 1 and 0.05. The search converged in 6 iterations; each iteration was run with the same population parameters as the ζ calculations, except for the number of inactive cycles. In this case only 20 inactive cycles were used, since they were shown to be enough to converge the fission source. Each run took around 3.5 minutes to complete. The final ζ value was

9.3770×10^{-2}, which is in good agreement with the ζ calculation considering the uncertainties involved.

3.4. BEAVRS

The last model investigated in this work is the PWR BEAVRS, for which the critical boron concentration is searched for. A radial cross section of the 3D geometry is shown in Fig. 4, together with the fission rate radial profile calculated with the critical boron concentration.

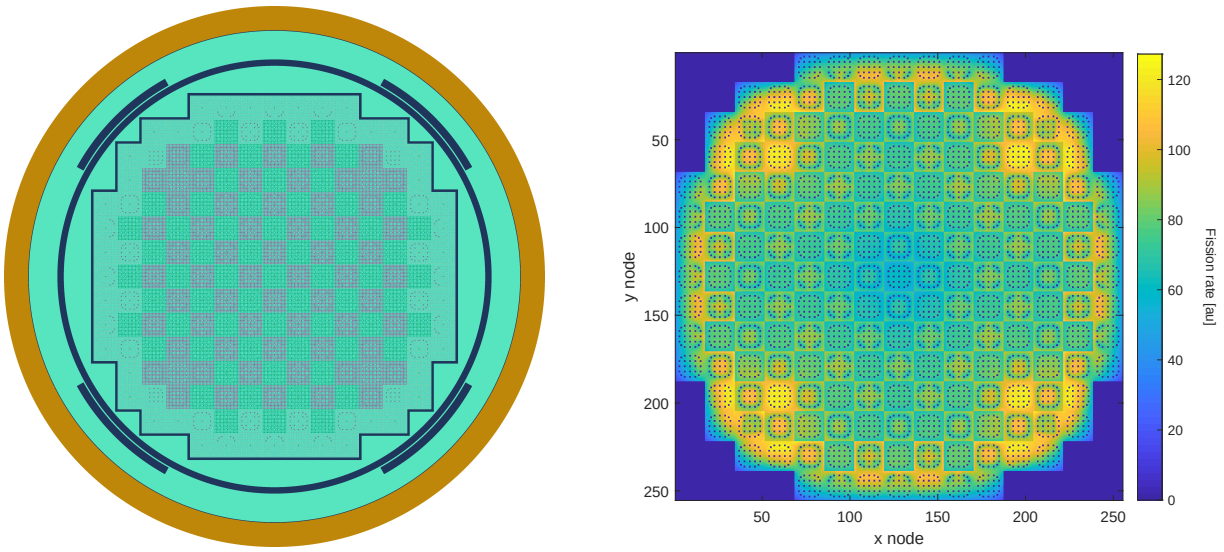

Figure 4. Radial geometry cross section (left) and fission rate profile (right) of the BEAVRS model

For this model, a ζ calculation is run to find the critical densities of ^{10}B and ^{11}B dissolved in the reactor coolant. The k_{eff} obtained for the starting densities, i.e., 7.9714×10^{-6} and 3.2247×10^{-5} atoms/barn/cm respectively for ^{10}B and ^{11}B, is 0.99838 ± 5 pcm. ζ was found to be $1.01356 \pm 2.3 \times 10^{-4}$, which corresponds to boron densities of $7.864\,75 \times 10^{-6} \pm 1.8 \times 10^{-9}$ and $3.181\,56 \times 10^{-5} \pm 7.1 \times 10^{-9}$ atoms/barn/cm. An independent search was not performed for this case, since a naive search would have been too computationally expensive. A reference k-eigenvalue calculation was run with the critical boron densities, obtaining a value of k_{eff} equal to 1.00004 ± 3 pcm. These simulations were run with 1 million particles per cycles, 200 inactive cycles and 50 active cycles, and took between 3 and 4 hours to complete.

Fig. 5 shows how the model converges during the 200 inactive cycles. 200 cycles were chosen as they were shown to be enough to converge a k-eigenvalue calculation of 3D BEAVRS [13]. From Fig. 5, it can be seen that in this case, both the eigenvalues and the fission source converge at the same rate for the two calculation types. This is probably the case since the starting initial boron density was not far from the critical one, and boron is homogeneously distributed everywhere in the reactor rather than being local like the gadolinium in the previous example.

4. SUMMARY AND CONCLUSIONS

In this paper, a Monte Carlo implementation of the ζ-eigenvalue equation is proposed. The ζ-eigenvalue equation is a formulation of the neutron transport equation which ensures criticality by scaling the density of the nuclides or materials of choice with the parameter ζ. The Monte Carlo algorithm consists of a fixed point iteration scheme, where the particle source at each iteration is the fission source of the previous one. This allows the implementation to be trivial, in Monte Carlo codes that are able to solve k-eigenvalue problems.

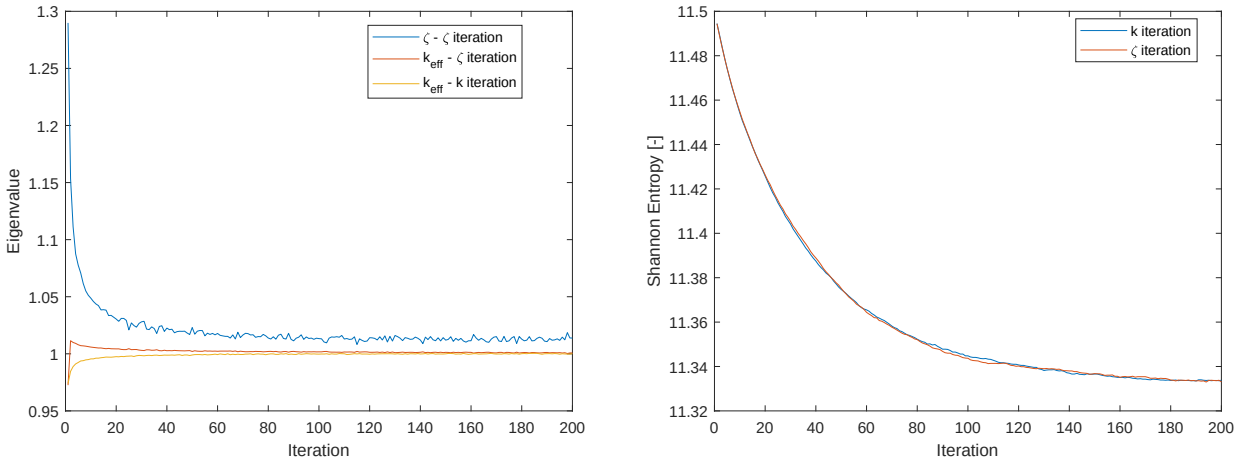

Figure 5. Convergence of the BEAVRS model: k_{eff} and ζ (left) and Shannon entropy (right) as a function of the inactive cycle number. The k_{eff} and the Shannon Entropy are plotted both for the ζ calculation, and for the k-eigenvalue calculation with the critical boron densities

The method was tested on some realistic numerical cases, like a critical gadolinium search in a 3D PWR assembly and a critical boron search in the BEAVRS reactor. The ζ algorithm could find the critical densities requested by the user in almost all cases; the only exception was that of a poorly posed problem, for which the Monte Carlo ζ estimators returns an unphysical negative value. This could happen, according to Eq. (7), when the material that one is performing the search on is a net absorber, and there are net neutron losses in the rest of the system. This scenario occured when trying to search the critical water density in a 2D PWR pincell, and the starting water density was such that the system was subcritical and under-moderated. In this model, the ζ algorithm always converged to the over-moderated critical water density over the under-moderated one. Future work will focus on alternative formulations of the ζ equation, that might allow converging on the under-moderated solution.

From the other numerical tests, which all converged smoothly, some general observations can be drawn. The relative uncertainty of the calculated ζ, at the end of the active cycles, is always larger than that of k_{eff} from the same calculation: this is the case since estimating ζ requires obtaining some local reaction rates, while k_{eff} is an integral parameter. In the case where the material whose density is scaled by ζ is very localised, or has low cross sections, the ζ estimator can have substantial statistical noise.

The numerical convergence of some models was monitored during the inactive cycles. Despite the similarities in algorithms, there is no reason to assume that the ζ calculations will converge at the same speed as k-eigenvalue calculations for the same model. This was shown in the case of the 3D PWR assemblies with gadolinium. When the density search is performed on a localised material with extremely large cross sections, like gadolinium-bearing pins, convergence might be slow. In other cases, like the critical boron search in BEAVRS, ζ calculations and k-eigenvalue calculations converged exactly at the same rate. Overall, the runtimes of ζ calculations and k-eigenvalue calculations are comparable if the same population settings are used.

Future work on this topic will focus on rigorous performance comparisons with the most efficient search methods in the literature, and more detailed convergence analysis.

ACKNOWLEDGEMENTS

This work was supported by the EPSRC grant MaThRad (EP/W026899/1). Peter Benie assisted through maintaining the Cambridge Nuclear Energy Group workstations. Valeria Raffuzzi acknowledges Paul Cosgrove for useful discussions.

This work has been carried out under the auspices of the Italian National Group of Mathematical Physics (Gruppo Nazionale di Fisica Matematica, GNFM) of the National Institute of High Mathematics (Instituto Nazionale di Alta Matematica, INDAM).

REFERENCES

[1] D. E. Cullen, C. J. Clouse, R. Procassini, and R. C. Little. "Static and Dynamic Criticality: Are They Different?" Technical Report UCRL-TR-201506, Lawrence Livermore National Laboratory (2003).

[2] B. Davison. *Neutron Transport Theory*. Clarendon Press, Oxford (1957).

[3] Y. Ronen, D. Shalitin, and J. J. Wagschal. "A Useful Different Eigenvalue for the Transport Equation." *Transactions of the American Nuclear Society*, **volume 24**, p. 474 (1976).

[4] N. Abrate, S. Dulla, P. Ravetto, and P. Saracco. "A Generalized Eigenvalue Formulation for Core-Design Applications." *Nuclear Science and Engineering*, **volume 197**(8), pp. 2047–2071 (2024).

[5] D. Mancusi and A. Zoia. "Chaos in Eigenvalue Search Methods." *Annals of Nuclear Energy*, **volume 112**, pp. 354–363 (2018).

[6] M. A. Kowalski, P. Cosgrove, J. Broman, and E. Shwageraus. "SCONE: A Student-Oriented Modifiable Monte Carlo Particle Transport Framework." *Journal of Nuclear Engineering*, **volume 2**(1), pp. 57–64 (2021).

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

[8] A. Zoia, E. Brun, and F. Malvagi. "Alpha Eigenvalue Calculations with TRIPOLI-4." *Annals of Nuclear Energy*, **volume 63**, pp. 276–284 (2014).

[9] B. C. Kiedrowski. "Evaluation of Computing c-Eigenvalues with Monte Carlo." In *Proceedings of the 2012 ANS Annual Meeting*. Chicago, IL (2012).

[10] "International Handbook of Evaluated Criticality Safety Benchmark Experiments." (2023). Available online: https://www.oecd-nea.org/science/wpncs/icsbep/.

[11] V. Raffuzzi, E. Shwageraus, L. Morgan, and P. Cosgrove. "Monte Carlo Source Convergence Acceleration by Hybrid Multigroup and Continuous Energy Neutron Transport." *Nuclear Science and Engineering*, **volume 197**(3), pp. 364–380 (2023).

[12] D. F. Gill, B. R. Nease, and D. P. Griesheimer. "Movable Geometry and Eigenvalue Search Capability in the MC21 Monte Carlo Code." In *Proceedings of the M&C 2013*. Sun Valley, Idaho (2013).

[13] B. R. Herman. *Monte Carlo and Thermal Hydraulic Coupling using Low-Order Nonlinear Diffusion Acceleration*. Doctor of science thesis, Massachusetts Institute of Technology, Cambridge, MA (2014).