Progress in Nuclear Energy 162 (2023) 104731

Contents lists available at ScienceDirect

Progress in Nuclear Energy

journal homepage: www.elsevier.com/locate/pnucene

Method for control drum position critical search with Monte Carlo codes

Dean Price [a,b,*], Nathan Roskoff [a]

[a] Westinghouse Electric Company LLC, 1000 Westinghouse Dr, Cranberry Township, PA 16066 United States of America
[b] Department of Nuclear Engineering and Radiological Science, University of Michigan, 2355 Bonisteel Blvd., Ann Arbor, MI 48109, United States of America

ARTICLE INFO

Keywords:
eVinci
Control drum
Critical search
Monte Carlo
Burnup

ABSTRACT

The use of Monte Carlo codes when performing optimization or root-finding routines can lead to complications due to the statistical uncertainty associated with calculated results. Monte Carlo codes are often needed to model the complex geometries associated with microreactors so when a critical search is performed to find the rotational position of the control drums which yield a critical core, the noise in calculated criticalities can disrupt the progression of the search. In addition, gradient-based root-finding algorithms are most appropriate for this critical search due to the monotonicity of the control drum position versus criticality relationship but these algorithms are particularly susceptible to the noise in calculated criticalities. Therefore, these weaknesses are specifically addressed with a few variants of secant-like methods that are presented in this paper and evaluated on control drum worth curves generated from the eVinci^{TM} and Holos-Quad microreactors. In the end, a single secant-like method is recommended for implementation due to superior performance on these worth curves. In general, these root-finding methodologies are created with ease-of-implementation as a significant motivator such that they can be implemented using only calculated criticalities, with uncertainty, from any Monte Carlo neutronics code without the need for source code access. The scripting framework required to implement this method only requires the programming of 3 explicit mathematical equations with expressions clearly provided in this paper and an outline to use this method to find burnup dependent critical drum positions is provided with an application to the eVinci^{TM} microreactor.

1. Introduction

In conventional light water reactors, both soluble boron and control rods are used to maintain reactor criticality and a variety of methods can be used to find the depletion-dependent configuration of these reactivity control systems when using deterministic codes. For example, PARCS uses the secant method to find critical boron concentration (Downar et al., 2004) to maintain criticality during a burnup calculation. However, Monte Carlo codes are often used to perform depletion analysis of microreactors, as in Stauff et al. (2022), Guo et al. (2021) and Hernandez et al. (2019), due to their ability to explicitly model the complex geometries associated with these cores. Unlike some lower-fidelity simulation methods, these codes do not require any homogenization in space or energy. Naturally, deterministic methods certainly have a place in microreactor analysis. Deterministic methods have been used for multiphysics and transient modeling (Roskoff et al., 2022a,b). These methods are particularly useful in safety calculations where the design is fixed so the development of a mesh and cross section library only need to be performed once for analysis of a variety of transient scenarios. However, the large computational cost and uncertainties in results calculated with Monte Carlo codes lead to difficulties in the algorithms used for critical searches. Therefore, algorithms which can efficiently account for these uncertainties in calculated results are essential when performing depletion analysis with critical drum positions in microreactors. The current manuscript presents an easy-to-implement algorithm for determining burnup-dependent critical control drum positions for microreactors.

The modeling of reactor parameters such as boron concentration, coolant density distribution or control rod insertion during burnup can have significant effect on calculated quantities. A two-part study (Radaideh et al., 2019; Price et al., 2019) was published on how the modeling of these types of parameters affect spent fuel compositions. It was found that fuel storage casks loaded with fuel compositions predicted using the various modeling assumptions varied by over 5000 pcm. Mertyurek and Ilas (2022) performed an in-depth exploration of how modeling assumptions associated with coolant density, power history and fuel characteristics each impacted the agreement between code-predicted and measured fuel compositions. Although these studies

* Corresponding author at: Department of Nuclear Engineering and Radiological Science, University of Michigan, 2355 Bonisteel Blvd., Ann Arbor, MI 48109, United States of America.
E-mail address: deanrp@umich.edu (D. Price).

https://doi.org/10.1016/j.pnucene.2023.104731
Received 6 November 2022; Received in revised form 21 February 2023; Accepted 4 May 2023
Available online 13 June 2023
0149-1970/© 2023 Elsevier Ltd. All rights reserved.

pertain to conventional light water reactors, they highlight the importance of performing depletion calculations with a reactor configuration most similar to the real operational conditions as end-of-life (EOL) characteristics of the reactor are affected by these modeling assumptions. Unlike the depletion analyses performed by Mertyurek and Ilas (2022), there may be a lack of operational data which requires more detailed computational simulation to reconstruct realistic operating conditions as is done in Price et al. (2020) where the coolant density distribution is reconstructed using a multiphysics simulation. With no operational data on microreactors, there is always a need to approximate the realistic operating conditions, such as control drum position, using simulations. Unfortunately, the Monte Carlo codes used to perform depletion analyses on microreactors present unique challenges due to their extensive computational cost and statistical uncertainty associated with results.

Studies which address the use of Monte Carlo codes in determining critical configurations for reactivity control systems certainly exist. Topham et al. (2020) used a fission matrix method for calculating critical control rod position with Doppler feedback for a TRIGA reactor. The fission matrix method essentially created a surrogate model which could be used iteratively in a secant-like method for minimal computational cost. Similarly, Price et al. (2022b) used the surrogate model described in Price et al. (2022a) for control drum worths and evolutionary optimization methods to determine critical drum positions for the Holos-Quad microreactor. In both of these methods, no burnup was considered which would increase the complexity of generating and training the surrogate models. Using a neutron balance approach for critical search originally proposed in Dall'Osso (2008) for deterministic methods, Jo and Cho (2018) demonstrates a critical boron search with feedback using a Monte Carlo code on two test problems. The drawback of these methods is that they are comparatively difficult to implement or adapt to the control drum problem compared with the methodology described in the present study. Furthermore, criticality search capabilities are available in some neutronics codes. The Serpent 2 neutronics code (Leppänen et al., 2015) has the capability to perform a critical density search which can be used to find critical boron concentration. However, as of Serpent 2.1.32, the critical density search functionality only works on nuclides whose "reactivity effect [works] mainly through neutron absorption". In general, perturbation-based critical search methods are more difficult to apply to movable geometry problems. Similarly, Li et al. (2015) presents a method which is implemented in the RMC Monte Carlo code which uses a higher-order perturbation method to find solve the criticality search problem in a single criticality calculation by using perturbation tallies. This method was applied separately to fuel density, enrichment and boron concentration search problems with success. The CSAS5 module in the SCALE code system (Bowman, 2011) has the capability to modify densities or dimensions of models search for a target k_{eff} using a secant-like method (Rearden and Jessee, 2018). The MC21 neutronics code (Griesheimer et al., 2014) has the capability for criticality searches with movable geometry using a published algorithm (Gill et al., 2013). With built-in capabilities, critical searches require minimal human effort to perform but are limited to specific scenarios, as is the case of Serpent 2. The containment of a critical search algorithm within a specific neutronics code causes problems when a workflow for microreactor simulation has already been established in another code.

Therefore, the methodology described in this paper can be implemented using any neutronics code capable of performing burnup calculations, without source code access, and has a mathematical component which is simple to implement as it requires the programming of only 3 equations with explicit solutions. The methods described in this paper are shown to consistently converge for the critical control drum position search for two different reactor designs. Convergence is not guaranteed for any general critical geometry search, in particular, non-monotonic relationships between geometric position and criticality can prevent convergence in these methods. In this manuscript, different variants of the methodology are presented and their time-efficiency when solving a critical control drum positions search problem are tested using a surrogate model for both drum worth and a code-runtime model. The surrogate models for drum worth are created using a regression-based method on worth curves for the eVinci^{TM} microreactor separately for fresh, middle of life and end of life fuel compositions and the Holos-Quad model. A model for code runtime is used to simulate the time it would take to perform an iterative search with Monte Carlo model evaluations. This code runtime model reports a time based on real evaluation times for the Serpent 2 model of the eVinci^{TM} microreactor for a provided level of uncertainty. The different variants of the methodology include methods for selecting the next point in an iterative search process and different methods for selecting the number of histories to run for the next evaluation point. Finally, a demonstration of how this critical drum search methodology can be used for depletion with critical drum positions is presented with an application to the eVinci^{TM} microreactor.

The remainder of this paper is organized by first presenting a methodology section which is further subdivided into six subsections. The first of these subsections gives explanation and three methods on selecting the control drum angle with which to perform the next k_{eff} calculation in an iterative search process. Then, the next subsection provides some discussion on how the uncertainty in each k_{eff} calculation impacts this control drum angle. The two subsections after this discuss how to select the uncertainty which the criticality calculation should be performed with and how to select the Monte Carlo sampling parameters to achieve this uncertainty in a calculation. Then, the incorporation of this critical search process into a workflow to produce burnup-dependent critical control drum angles is presented. The final subsection of the methodology section discusses how surrogate models will be used to compare the performance of the different critical search methods. The results section has eight subsections. The first of these subsections compares the uncertainty in the next evaluation points in the various search routine methods. The second of these subsections evaluates how effective the equation used to translate a desired uncertainty in k_{eff} to the Monte Carlo sampling parameters required in code input. Next, the performance of the most simple of the search routines is evaluated on four different surrogate models. Then, the different methods are directly compared to one-another using a mean simulated runtime across a large number of independent search routines. The fifth subsection explores the use of variations of the search routine which differently select the uncertainty with which to evaluate the next search location. The sixth subsection uses a cosine transformation on the control drum angle versus criticality relationship to linearize it and analyses the effect that this transformation has on search routine performance. The penultimate subsection applies some different search routine variants to problems with differing search routine tolerances. The final subsection briefly shows the results when the methodology is applied to the eVinci^{TM} microreactor.

2. Methodology

2.1. Search location selection

A control drum position search can be formulated as a root-finding problem by finding the θ such that

$$f(\theta) \equiv k(\theta) - k_{tgt} = 0. \tag{1}$$

For notational simplicity, $f(\theta)$ is introduced as the objective function. Here, $k(\theta)$ represents the functional dependence of core criticality on the angle of the control drum θ (or a set of control drums with the same orientation) and k_{tgt} is the desired eigenvalue for the search (for criticality, $k_{tgt} = 1$). For the methods described here, θ will be bounded by 0° and 180° where 0° corresponds to the case where the absorbing material on each of the control drums is facing fully inwards. In iterative search methods, it is important to establish some termination criteria which will set guidelines on determining if some θ sufficiently satisfies the objective of the root-finding problem. In this manuscript there will be two criteria established to determine if θ is sufficient.

1. First, is a proximity criteria. If $f(\theta_n) \leq k_{tol}$, then the search can be terminated at the nth iteration where θ_n is the search location on the nth iteration. Next is a certainty criteria.
2. Due to the uncertainty in the Monte Carlo method, an additional threshold must be specified as the maximum acceptable uncertainty in $f(\theta_n)$ to determine if it is sufficient to terminate the search. Here, σ_{final} will be used to represent the maximum acceptable uncertainty in $f(\theta)$ such that the second termination criteria is $\sigma_{final} \geq \sigma_{f,n}$. $\sigma_{f,n}$ is the uncertainty reported in the Serpent 2 (as well as most Monte Carlo neutronics codes) code output for $f(\theta_n)$.

Both k_{tol} and σ_{final} are user-specified values. The remainder of this subsection describes three secant-based methods for root finding, a recommendation for which method to implement will be provided in the conclusion of this study. Although secant methods may fail to converge in their application to general functions, in this particular application, they consistently converge.

2.1.1. Secant method

One well-known method for finding θ would be the secant method. To use this method, two starting points are selected between 0° and 180° which can be denoted as θ_0 and θ_1. Then, $k(\theta_0)$ and $k(\theta_1)$ are calculated and a line is fit. θ_2 is then set to where the fitted line intersects with k_{tgt}. Mathematically, new search locations are iteratively selected with:

$$\theta_{n+1} = \frac{\theta_{n-1} f(\theta_n) - \theta_n f(\theta_{n-1})}{f(\theta_n) - f(\theta_{n-1})}. \tag{2}$$

A derivation for this expression is provided in Equation Appendix A.1. One adjustment to this search method which may affect convergence is to use a $\cos(\theta)$ transformation on θ to linearize $k(\theta)$. This transformation is inherent to the rotating geometry of control drums and will be demonstrated in Section 2.6 for control drum worth curves for the eVinci and Holos-quad microreactors. The application of this transformation to the search problem while still using the secant method described in Eq. (2) yields:

$$\theta_{n+1} = \cos^{-1}\left(\frac{\cos(\theta_{n-1}) f(\theta_n) - \cos(\theta_n) f(\theta_{n-1})}{f(\theta_n) - f(\theta_{n-1})}\right). \tag{3}$$

Of course, θ_{n-1}, θ_n and θ_{n+1} may need to be converted to radians depending on the requirements of the $\cos$ and $\cos^{-1}$ functions. Furthermore, boundary constraints should be applied prior to the application of the $\cos^{-1}$ function to ensure its argument is valid.

2.1.2. Rsecant method

In this application, the uncertainty from the evaluation of $k(\theta)$ with Monte Carlo methods can cause significant interruption in the progression of a search using the unmodified secant method. There is a more mathematical discussion on the effect of uncertainty on the different search methods presented in Appendix B. By making a sacrifice to the rate of convergence to the secant method, the Rsecant (regressive secant) method uses linear regression on $R+2$ points—where R is an integer selected by the user. Then, the location where the new regression line intersects with 0 is the next search location. By incorporating some aspect of "memory" into the search routine, the Rsecant method more gradually approaches the targeted drum position but is more resistant to uncertainty. Following a derivation provided in Equation Appendix A.2, the following equation can be obtained for the next search location:

$$\theta_{n+1} = \frac{\sum_{i=0}^{R+1} \theta_{n-i} \sum_{i=0}^{R+1} \theta_{n-i} f(\theta_{n-i}) - \sum_{i=0}^{R+1} f(\theta_{n-i}) \sum_{i=0}^{R+1} \theta_{n-i}^2}{(2+R)\sum_{i=0}^{R+1} \theta_{n-i} f(\theta_{n-i}) - \sum_{i=0}^{R+1} \theta_{n-i} \sum_{i=0}^{R+1} f(\theta_{n-i})}. \tag{4}$$

The secant method and Rsecant method are equivalent when $R = 0$. As such, when beginning the search, it is more effective to still only select two starting points as is done with the secant method. Then, the R used in Eq. (4) can be increased by 1 with each search iteration until the user-specified R is achieved. By applying a cosine transformation to the θ-axis, the following expression can be used for the next search location:

$$\theta_{n+1} = \cos^{-1}\left(\frac{\sum_{i=0}^{R+1} \cos(\theta_{n-i}) \sum_{i=0}^{R+1} \cos(\theta_{n-i}) f(\theta_{n-i}) - \sum_{i=0}^{R+1} f(\theta_{n-i}) \sum_{i=0}^{R+1} \cos(\theta_{n-i})^2}{(2+R)\sum_{i=0}^{R+1} \cos(\theta_{n-i}) f(\theta_{n-i}) - \sum_{i=0}^{R+1} \cos(\theta_{n-i}) \sum_{i=0}^{R+1} f(\theta_{n-i})}\right). \tag{5}$$

2.1.3. GRsecant method

The final modification to the method for selecting the next search location is the GRsecant (generalized regressive secant) method where linear regression is used but each point is weighted to account for differences in the Monte Carlo uncertainty associated with these points. This method will be especially useful because the Monte Carlo error in each evaluation of $k(\theta)$ will change due to the methods described in Section 2.3. With the GRsecant method, data points with less error will be treated with more importance. Formally, the uncertainty in an evaluation of $k(\theta_n)$ can be mathematically represented by a standard deviation, $\sigma_{f,n}$ which is reported in the Serpent 2 (as well as most Monte Carlo neutronics codes) code output. With a derivation presented in Appendix A.3, the next search location can be obtained with:

$$\theta_{n+1} = \frac{\sum_{i=0}^{R+1} \frac{\theta_{n-i}}{\sigma_{f,n-i}^2} \sum_{i=0}^{R+1} f(\theta_{n-i}) \frac{\theta_{n-i}}{\sigma_{f,n-i}^2} - \sum_{i=0}^{R+1} \frac{\theta_{n-i}^2}{\sigma_{f,n-i}^2} \sum_{i=0}^{R+1} \frac{f(\theta_{n-i})}{\sigma_{f,n-i}^2}}{\sum_{i=0}^{R+1} f(\theta_{n-i}) \frac{\theta_{n-i}}{\sigma_{f,n-i}^2} \sum_{i=0}^{R+1} \frac{1}{\sigma_{f,n-i}^2} - \sum_{i=0}^{R+1} \frac{f(\theta_{n-i})}{\sigma_{f,n-i}^2} \sum_{i=0}^{R+1} \frac{\theta_{n-i}}{\sigma_{f,n-i}^2}}. \tag{6}$$

Similarly to the Rsecant method, to begin a search, only two initial points should be specified. Then, with each sucessive search point, R should be increased by 1 until the user-specified R is achieved. With a cosine transformation,

$$\theta_{n+1} = \cos^{-1}\left(\frac{\sum_{i=0}^{R+1} \frac{\cos(\theta_{n-i})}{\sigma_{f,n-i}^2} \sum_{i=0}^{R+1} f(\theta_{n-i}) \frac{\cos(\theta_{n-i})}{\sigma_{f,n-i}^2} - \sum_{i=0}^{R+1} \frac{\cos(\theta_{n-i})^2}{\sigma_{f,n-i}^2} \sum_{i=0}^{R+1} \frac{f(\theta_{n-i})}{\sigma_{f,n-i}^2}}{\sum_{i=0}^{R+1} f(\theta_{n-i}) \frac{\cos(\theta_{n-i})}{\sigma_{f,n-i}^2} \sum_{i=0}^{R+1} \frac{1}{\sigma_{f,n-i}^2} - \sum_{i=0}^{R+1} \frac{f(\theta_{n-i})}{\sigma_{f,n-i}^2} \sum_{i=0}^{R+1} \frac{\cos(\theta_{n-i})}{\sigma_{f,n-i}^2}}\right). \tag{7}$$

2.2. Implications of uncertainty from Monte Carlo calculation of k_{eff}

Due to the uncertainty in calculated k_{eff} values from the Monte Carlo calculation method, the Rsecant and GRsecant method can outperform the normal secant method in the criticality search despite their lower theoretical orders of convergence (they approach the optima less aggressively than the secant method). Also, the GRsecant method is the only method which explicitly considers this uncertainty in its calculation of θ_{n+1} as shown in Eq. Eq. (6). One useful route for analysis of the performance of these methods is to quantify the uncertainty in next selected search location (θ_{n+1}) due to the uncertainty in k_{eff} from the Monte Carlo calculation method. This uncertainty propagation is done for additional insight on the performance of these methods; this analysis is certainly not necessary for one to implement the core of the methodology described in this paper . First, it is useful to formalize the treatment of uncertainty. Any time Serpent 2 (or any similar Monte Carlo code) is used to calculate k_{eff} with some control drum angle, some uncertainty in k_{eff} is reported by the code. This uncertainty allows for code results to be represented as a random variable with mean $k(\theta)$ and variance σ_k^2. In Eq. (1), from algebra of random variables, it should be clear that $\sigma_k^2 = \sigma_f^2$.

By treating each evaluation of $f(\theta)$ as a random variable, mathematical statistics can be used to propagate the randomness in each $f(\theta)$ through to θ_{n+1}. A rigorous treatment of the mathematics required to do this for the untransformed versions of the search methods (Eqs. (2),

(4) and (6)) is provided in Appendix B. These methods are only valid when $\sigma_f / f(\theta)$ is reasonably small. The additional requirements to carry out this uncertainty propagation for the transformed versions of the search methods (Eqs. (3), (5) and (7)) is described in Appendix B.4. In conclusion, expressions for propagating these uncertainties with high accuracy are provided in these appendices and the results from such analysis will be included in the results of this study. These uncertainty propagation methods are kept in appendices—instead of included in the main text—to emphasize that they are not required to implement the search method described in this paper.

2.3. Monte Carlo uncertainty selection

In most simple root-finding problems, it is sufficient to provide an expression for the next search location based on a set of previous search locations as is done in Section 2.1. However, when using Monte Carlo methods to calculate $f(\theta_{n+1})$ it is also necessary to provide $\sigma_{f,n+1}$, or the uncertainty $f(\theta_{n+1})$ should be evaluated to. Larger values for $\sigma_{f,n+1}$ will lead to smaller evaluation times for $f(\theta_{n+1})$ but more noise in the result. With Serpent 2, the user does not directly specify $\sigma_{f,n+1}$, instead, they specify sampling parameters used in the simulation. Although the names of these sampling parameters change depending on which Monte Carlo neutronics code is used, in Serpent 2, they are referred to as active generations, inactive generations and neutrons per generation. The selection of these sampling parameters to yield some target $\sigma_{f,n+1}$ is relatively straightforward and provided in Section 2.4.

In this methodology, $\sigma_{f,n+1}$ is selected with:

$$\sigma_{f,n+1} = 0.95\sigma_{final}\left(\frac{\min\left\{|f(\theta_k)| \, : \, k = 0, 1, \dots, n\right\}}{k_{tol}}\right)^p . \tag{8}$$

In this equation p is a parameter which must be set by the user and this same equation is used regardless of whether secant, Rsecant or GRsecant is used. A recommended value for this parameter will be given in the conclusion of this paper. It is also recommended that $\sigma_{f,n+1}$ be set to $0.95\sigma_{final}$ if $\sigma_{f,n+1} < 0.95\sigma_{final}$ to avoid excessive computation times. $\min\left\{|f(\theta_k)| \, : \, k = 0, 1, \dots, n\right\}$ represents the best value of $f(\theta)$ obtained so far in the search. The form of this equation is empirically determined based on numerical tests and has three theoretical motivations:

1. When $|f(\theta_n)| < k_{tol}$, $\sigma_{f,n+1} < \sigma_{final}$. With the assumption of monotonically decreasing $|f(\theta_n)|$ over each search iteration, this means that both convergence criteria are satisfied within one iteration of eachother.
2. Due to the random nature of the search, monotonically decreasing $|f(\theta_n)|$ is *not* guaranteed. However, with this equation, monotonically decreasing $\sigma_{f,n+1}$ *is* guaranteed which helps minimize cyclic search behavior.
3. The factor of 0.95 often reduced the number of required iterations for convergence because it permitted both convergence criteria to be satisfied on the same iteration. Without this factor, it is impossible for the $\sigma_{f,n+1} < \sigma_{final}$ convergence criteria to be satisfied until the iteration after the $|f(\theta_n)| < k_{tol}$ criteria is satisfied.

Note that when $p = 0$, $\sigma_{f,n+1}$ is held constant at σ_{final} for the duration of the search. Furthermore, this formulation requires that the uncertainty in the first two evaluation points ($\sigma_{f,0}$, $\sigma_{f,1}$) be specified by the user to be roughly the same order of magnitude as σ_{final}. Excessively high specifications of $\sigma_{f,0}$ or $\sigma_{f,1}$ can be problematic because code estimations for σ_f may be invalid when too few generations are run. Although only $\sigma_{f,0}$ is required for the determination of $\sigma_{f,n+1}$ using the formula above, translating $\sigma_{f,n+1}$ into the Monte Carlo sampling parameters required by code inputs requires at least two evaluation points. One limitation of this approach is that it relies heavily on code-reported uncertainties in k$_{eff}$ and these reported uncertainties are biased to be underestimations of the true uncertainty in k$_{eff}$ due to the Monte Carlo calculation method (Brown, 2009).

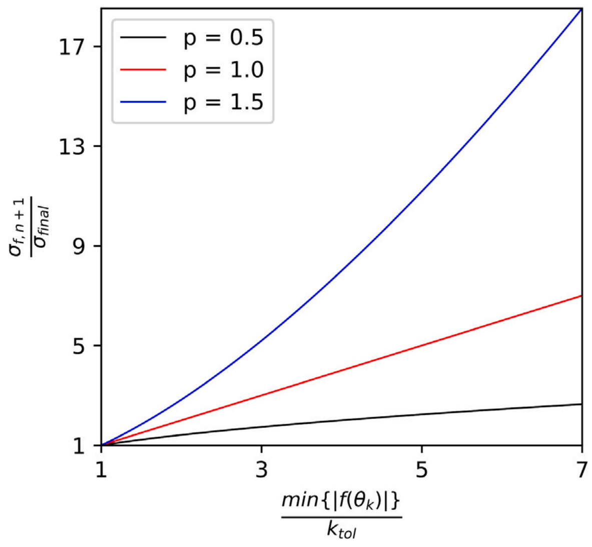

Fig. 1. Relationship between convergence in $f(\theta)$ evaluation uncertainty and proximity of $f(\theta)$ to optimal value. These specific quantities are selected for plotting because $\frac{\sigma_{f,n+1}}{\sigma_{final}}$ represents the closeness to satisfying the $\sigma_{final} \le \sigma_{f,n}$ termination criteria and $\frac{\min\{|f(\theta_k)|:k=0,1,\dots,n\}}{k_{tol}}$ is indicative of the closeness to satisfying the $f(\theta_n) \le k_{tol}$ termination criteria.

In Fig. 1, the relationship between $\frac{\min\{|f(\theta_k)|:k=0,1,\dots,n\}}{k_{tol}}$ and $\frac{\sigma_{f,n+1}}{\sigma_{final}}$ is presented for different values of p. Note that the denominators of these quantities are user-specified and do not change over the course of the search routine. These specific quantities are selected for plotting because $\frac{\sigma_{f,n+1}}{\sigma_{final}}$ represents the closeness to satisfying the $\sigma_{final} \le \sigma_{f,n}$ termination criteria and $\frac{\min\{|f(\theta_k)|:k=0,1,\dots,n\}}{k_{tol}}$ is indicative of the closeness to satisfying the $f(\theta_n) \le k_{tol}$ termination criteria. When both of these fractions are less than or equal to 1, the search is terminated (with the assumption of monotonically decreasing $|f(\theta_n)|$). This assumption will not hold in many search routines, it is only used here as a demonstration for the intuition behind the form of Eq. (8). Nevertheless, from these trends, it is clear that higher p values lead to higher values for $\sigma_{f,n+1}$ and would essentially prioritize a fast evaluation of $f(\theta_n)$ despite the higher uncertainties. Lower p values do the opposite, they tend to have lower values for $\sigma_{f,n+1}$ which have slower evaluation times of $f(\theta_n)$ but benefit from lower uncertainties. From these trends alone, it is difficult to determine the best p value for a drum positions search problem but this will be explored later.

2.4. Sampling parameter translation to uncertainty

As mentioned in Section 2.3, $\sigma_{f,n+1}$ cannot be directly specified in Monte Carlo code inputs. Instead, the Monte Carlo sampling parameters (active generations, inactive generations and neutrons per generation) used in calculations need to be selected to achieve some $\sigma_{f,n+1}$. The purpose of this section is to guide the selection of these sampling parameters. The number of inactive generations and neutrons per generation should be selected based on prior experience with a model, for some general guidance on the selection of these parameters, see Brown (2009). Then, active generations can be tuned in order to achieve the desired $\sigma_{f,n+1}$. If g_n is the number of active generations used in a previous evaluations of $f(\theta_n)$ to achieve some uncertainty of $\sigma_{f,n}$, then

$$g_{n+1} = \exp\left(\left(\frac{\sum_{i=0}^{n} \ln\left(\sigma_{f,i}\right) \sum_{i=0}^{n} \ln\left(g_i\right)^2 - \sum_{i=0}^{n} \ln\left(\sigma_{f,i}\right) \ln\left(g_i\right) \sum_{i=0}^{n} \ln\left(g_i\right)}{(n+1)\sum_{i=0}^{n} \ln\left(g_i\right)^2 - \left(\sum_{i=0}^{n} \ln\left(g_i\right)\right)^2}\right.\right.$$

$$- \ln\left(\sigma_{f,n+1}\right)\Bigg)\Bigg) \tag{9}$$

$$\times \frac{(n+1)\sum_{i=0}^{n}\ln\left(g_i\right)^2 - \left(\sum_{i=0}^{n}\ln\left(g_i\right)\right)^2}{\sum_{i=0}^{n}\ln\left(\sigma_{f,i}\right)\sum_{i=0}^{n}\ln\left(g_i\right) - (n+1)\sum_{i=0}^{n}\ln\left(\sigma_{f,i}\right)\ln\left(g_i\right)}\Bigg). \tag{10}$$

can be used to calculate the number of generations required in the code input (g_{n+1}) to reach $\sigma_{f,n+1}$ level of uncertainty. The number of inactive generations and neutrons per generation should be the same across all evaluations of f. Here, the uncertainty in the first two evaluation points ($\sigma_{f,0}$, $\sigma_{f,1}$) needs to be specified by the user. The derivation for this equation is provided in Appendix C. In summary, linear regression is used to characterize the approximately linear relationship between $\ln(g)$ and $\ln(\sigma_f)$ based on previous evaluations of f. Then, this linear relationship is used to calculate g_{n+1} based on $\sigma_{f,n+1}$. The method described here is certainly approximate but performs adequately for this application as shown in Section 3.2.

2.5. Implementation framework of proposed methodology

So far, an iterative search method where control drum rotational position is modified to seek some selected k_{tgt} is described. In this section, the implementation of this search process in a practical workflow is presented because, in its full use case, this search process will need to be performed at multiple burnup steps in order to find burnup-dependent control drum positions which yield k_{tgt}. Additionally some general recommendations on the implementation of the search process will be presented. Fig. 2 shows a flow chart of a potential workflow which uses the described search methodology. This figure will be used as a platform for the following discussion.

Before beginning any calculations, search method, k_{tgt}, p, R, k_{tol} and σ_{final} should all be selected. k_{tgt}, k_{tol} and σ_{final} will directly impact the control drum positions this framework yields and should be carefully selected based on the needs of the analyst. Higher values for k_{tol} and σ_{final} will make the search process faster but will yield less precise control drum positions. The search method, p and R do not directly affect the final control drum positions, they modify details of the search process and may affect total runtime. The results portion of this study given in Section 3 is primarily concerned with optimally selecting these parameters for efficient searches, recommendations on the selections of these values is presented in Section 4 to conclude the work. Moving forward to begin with a fresh core, initial guesses for θ_0 and θ_1 should be selected along with g_0 and g_1. For some guidance on these selections, θ_0 and θ_1 should be at least 15° apart to prevent the uncertainty in $f(\theta_0)$ and $f(\theta_1)$ to overpower $f(\theta_1) - f(\theta_0)$. g_0 and g_1 can be selected such that $\sigma_{f,0}$ and $\sigma_{f,1}$ are roughly around $2 \times \sigma_{final} - 3 \times \sigma_{final}$ but they must be different in order for Eq. (9) to be applied properly. Although in these initial steps, the model to calculate $\sigma_{f,0}$ and $\sigma_{f,1}$ based on g_0 and g_1 does not yet exist, a very broad estimate for g_0 and g_1 is sufficient. Then, Serpent criticality calculations should be run to yield $f(\theta_0;\sigma_{f,0})$ and $f(\theta_1;\sigma_{f,1})$. The notation $f(\theta_n;\sigma_{f,n})$ is used to indicate that f is evaluated at θ_n with an uncertainty with a standard deviation of $\sigma_{f,n}$. Formally, convergence can be checked for each of these first two evaluations as shown in the green diamond labeled "Converged?" according to the criteria described at the beginning of Section 2.1.

After convergence is checked for both $f(\theta_0;\sigma_{f,0})$ and $f(\theta_1;\sigma_{f,1})$, the next search location (θ_{n+1}) can be calculated using equations provided in 2.1 for the selected method. For the GRsecant and Rsecant methods, R used in those equations should be the minimum of the user-selected R and $n-2$. Essentially, higher values of R require more data points so R should steadily increase as more data points are accumulated until the user-specified value of R is achieved. Following its calculation, $\theta_{n+1} \in [0°, 180°]$ should be enforced by reducing θ_{n+1} to 180° or by increasing θ_{n+1} to 0°. In some searches, the calculated θ_{n+1} can be out of this realistic range due to the uncertainty f evaluations. A continuation of the search process after confining θ_{n+1} will fix this problem. On the other hand, if the reactor cannot reach k_{tgt} for any control drum position due to high burnup or design deficiencies, the search process will continually require the enforcement of the $\theta_{n+1} \in [0°, 180°]$ condition. In these cases, the continual enforcement of bounds should serve as an indication that the search process should be terminated. Moving forward, the uncertainty ($\sigma_{f,n+1}$) in the next evaluation of f should be selected based on Eq. (8). Then, the g_{n+1} required for code input to achieve a $\sigma_{f,n+1}$ level of uncertainty can be calculated using Eq. (9). With these the next iteration of the search can begin where n is increased by 1.

Once both convergence criteria are satisfied, the control drum rotational position can be used to recalculate the spatial flux distribution and then deplete the fuel. In Serpent 2, these fuel compositions can be captured in restart files which can be input to criticality calculations to simulate the reactor that is loaded with partially burned fuel. For criticality searches using non-fresh fuel, more informed selections of θ_0 and θ_1 can be made. One strategy is to use the results from past critical searches and extrapolate to form the first guesses for a new critical search process.

2.6. Surrogate model testing

In order to practically evaluate the performance of the search routine described in this paper without excessive computation times, a series of surrogate models are created and the critical searches are run using these models instead of Serpent 2 calculations. In addition, these surrogate models will have simulated Monte Carlo uncertainty which will be based on a simulated "runtime". This approach allows for estimation of the true runtime of a search routine with different options and starting points without the excessive calculation time associated with running actual Serpent 2 calculations. Although the surrogate models will not perfectly recreate the search problem, it is thought that it is sufficient in their use to evaluate the differing search strategies presented in this study. Three different surrogate models are created based on the eVinci^{TM} microreactor design:

- Zero year model (0YM): this surrogate model is created with 37 Serpent 2 k_{eff} calculations on a beginning of life model with control drums rotating from 0° to 180° in 5° degree increments.
- Three year model (3YM): this surrogate model is created with 37 Serpent 2 k_{eff} calculations with fuel that has been depleted for 3 years at 15 MW and control drums rotating from 0° to 180° in 5° degree increments.
- Six year model (6YM): this surrogate model is created with 37 Serpent 2 k_{eff} calculations with fuel that has been depleted for 6 years at 15 MW and control drums rotating from 0° to 180° in 5° degree increments.

Then, one additional surrogate model (HMM) is created based on a Holos microreactor design using the control drum worth curves presented in Price et al. (2022a) where only 19 data points are available. The k$_{eff}$ calculations for models based on the eVinci^{TM} microreactor design have a reported Monte Carlo uncertainty of approximately 9 pcm while those for the HMM have a reported Monte Carlo uncertainty of approximately 14 pcm. Both of these surrogate models used fixed temperature distributions such that no thermal feedback is considered in response to a changing control drum position. These control drum worth curves pertain to a past iteration of the Holos-quad microreactor design which has since been updated (Stauff et al., 2022). Nevertheless, each of these four surrogate models will calculate $k(\theta)$ in Eq. (1) in place of Serpent 2. All surrogate models are fit to data similarly using regression with a modified Fourier basis. In each model, enough basis functions are included in the regression until all residuals are within the 95% confidence interval of the data point with its associated error. The fitting process is described in more detail in Appendix D and the data points used to fit each of the surrogate models, as well as the results

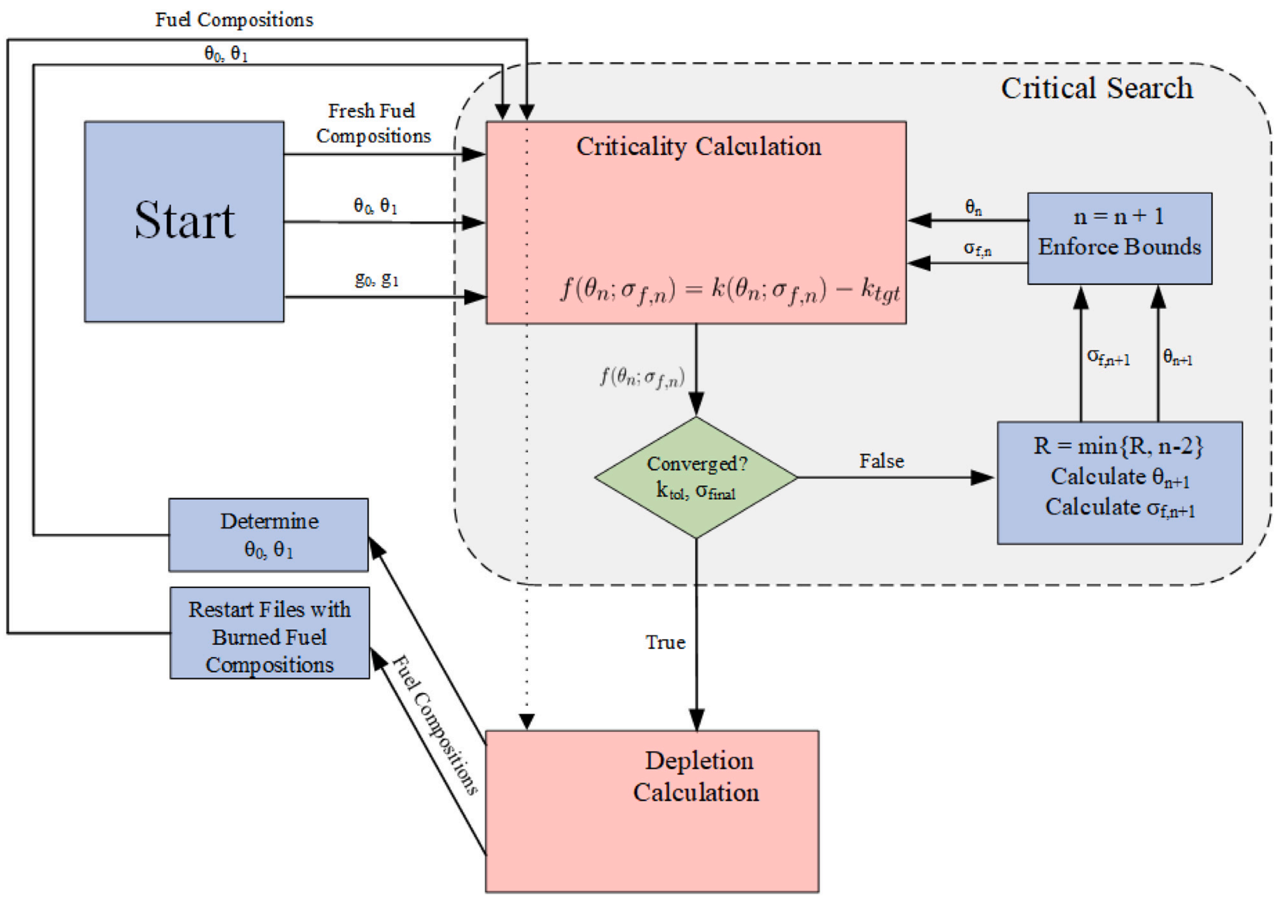

Fig. 2. Flow chart for potential workflow which employs the drum position search described in this paper.

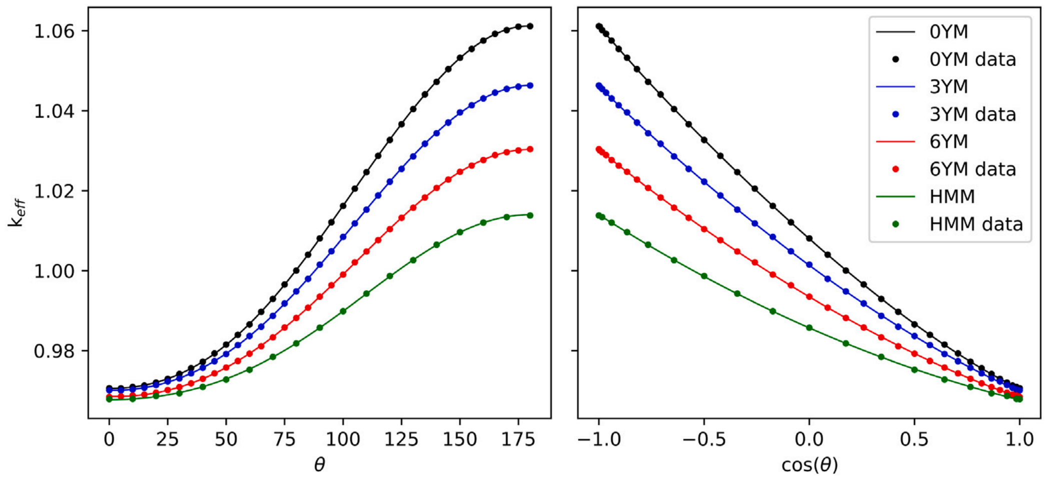

Fig. 3. Control drum worth curves with fitted surrogate models for 0YM, 3YM, 6YM and HMM. Trends are also plotted with a cosine transformation applied to the x-axis to demonstrate the relatively linear relationship between $\cos(\theta)$ and k_{eff}.

from the surrogate models themselves is shown in Fig. 3. Additionally, these trends are plotted with a cosine transformation applied to the x-axis to demonstrate the relatively linear relationship between $\cos(\theta)$ and k_{eff}.

For all four surrogate models, the same runtime model is used. For any evaluation of these surrogates, a runtime will be assigned to mimic the real-life computational time. This runtime model is based on an eVinci™ model where k_{eff} is calculated with Serpent 2 on a single processor. For simplicity, a runtime of 1000 min to achieve an uncertainty of about 30 pcm (neglecting code startup time) is used with the assumption of perfect $1/\sqrt{t}$ scaling such that:

$$t = \left(\frac{950\left[\mathrm{pcm}\sqrt{\mathrm{min}}\right]}{\sigma_f}\right)^2 + s. \tag{11}$$

Although the coefficient of 950 $\left[\text{pcm}\sqrt{\text{min}}\right]$ will certainly be highly dependent on the model used, the application of this single runtime model to all four surrogate models is reasonable because the order of convergence ($t \propto 1/\sigma_f^2$) is universal to Monte Carlo methods. In other words, total runtime may scale by a set factor for different models but the shape of the t versus σ_f relationship is universal. Here, s is included to represent the time required to read input, process cross sections, plot geometry or any other task which is not directly simulating the transport of neutrons. In this study, the effect on s on the optimal search method/options will be explored but is assumed to be 0 unless otherwise stated.

The different search strategies will be evaluated on a number of search routines using these surrogate models with simulated runtimes and simulated uncertainties. Every evaluation of the surrogate model will have an additional noise term which is sampled from a normal distribution described by the uncertainty associated with that model evaluation. In general, consistency in the optimal search method and options between the four different surrogate models will suggest a universality in the most favorable search method/options. Then the best-performing search method/options across the four surrogate models will be explicitly stated in the conclusions given in Section 4. In practice, the human time-investment required to set up the surrogate model testing framework described in this section to find the optimal search method with associated options, or configuration, for a specific reactor may not be justified from the saved computational time. Instead, it is expected that analysts will only implement the most optimal search method, as reported by this study, and any computational time lost over the use of a sub-optimal search method for that specific application will be compensated for in the saved human work time. It is helpful to explicitly mention that the search methods and options which are being explored will only affect the computational time required to solve a problem. They will not have any effect on the theoretical accuracy of the final results which are completely determined by k_{tol} and σ_{final}.

3. Results

3.1. Search location uncertainty

This section is a qualitative analysis on the behavior of the uncertainty in the next search location $\sigma_{\theta,n+1}$ given its proximity to the solution to the root-finding problem ($|\theta_{n+1} - \theta_{tgt}|$) for the secant, Rsecant and GRsecant methods. To explain, each evaluation of $f(\theta_n)$ has an uncertainty associated with it ($\sigma_{f,n}$) which arises from the Monte Carlo calculation method. This uncertainty can be propagated through the equations for calculating θ_{n+1} given in Section 2.1 to arrive at some uncertainty in θ_{n+1}, referred to here as $\sigma_{\theta,n+1}$. Then, if θ_{tgt} is defined such that $f(\theta_{tgt}) \equiv k_{tgt}$, this uncertainty in θ_{n+1} is plotted against $|\theta_{n+1} - \theta_{tgt}|$ in Fig. 4 to see how the uncertainty shrinks as the search converges to θ_{tgt}. For these results, a set of 70 independent search routines where $\sigma_{final} = 5$ pcm and $k_{tol} = 5$ pcm were executed on the 3YM with two different values for p. Here, $R = 2$ for both the Rsecant and GRsecant methods and no cosine transformation is used. In all of these search routines, each time θ_{n+1} was calculated, the impact of the uncertainties in $f(\theta_n)$, $f(\theta_{n-1})$, …, $f(\theta_{n-R-1})$ were propagated through that calculation and $\sigma_{\theta,n+1}$ was obtained. Then, these θ_{n+1} and $\sigma_{\theta,n+1}$ were recorded. Fig. 4 shows a portion of these two quantities across many iterations of many search routines to broadly depict their relationship. A line—in logarithm space–is fitted to the data points for each model to summarize the general trend in the data; this is not done to suggest a particular convergence behavior. Nevertheless, for all methods for both $p = 0$ and $p = 0.5$, $\sigma_{\theta,n+1}$ decreases as $|\theta_{n+1} - \theta_{tgt}|$ decreases which suggests that the uncertainty in the next search location due to the uncertainty in the Monte Carlo calculation method decreases as the search approaches its solution. Overall, $\sigma_{\theta,n+1}$ is lower for search routines where $p = 0$ than those where $p = 0.5$. This is because search routines run with $p = 0$ will have lower $\sigma_{f,n}$, $\sigma_{f,n-1}$, …, $\sigma_{f,n-R-1}$. When propagated through Eqs. (2), (4) or (6), this will ultimately result in a lower $\sigma_{\theta,n+1}$. Furthermore, the Rsecant and GRsecant methods tend to have a lower $\sigma_{\theta,n+1}$ than those for the secant method. This is due to the regressive character of these methods. The "memory" of past $f(\theta)$ inherent in these methods essentially allows the search to have lower σ_{θ_n+1}. Also, by weighting evaluations of f with lower uncertainty, the GRsecant method further reduces the uncertainty in θ_{n+1}. To summarize, there are two key findings from the results presented in Fig. 4:

1. Uncertainty in θ_{n+1} due to the Monte Carlo calculation method of $f(\theta_n)$, $f(\theta_{n-1})$, …, $f(\theta_{n-R-1})$ tends to decrease as the search approaches θ_{tgt}.
2. The Rsecant and GRsecant methods tend to reduce the uncertainty in θ_{n+1} for the same $|\theta_{n+1} - \theta_{tgt}|$, compared to the secant method.

This behavior is not valid when $|\theta_{n+1} - \theta_{tgt}|$ becomes small because the assumptions required in the mathematics required to obtain $\sigma_{\theta,n+1}$ become invalid.

3.2. Evaluation of sampling parameter translation to uncertainty

In this section, the method presented in Section 2.4 for translating Monte Carlo sampling parameters (active generations, inactive generations and neutrons per generation) into uncertainty from the Monte Carlo calculation method in k$_{eff}$ is evaluated for the current application. A pool of k$_{eff}$ with uncertainties (σ_f) are generated by creating 71 Serpent 2 models of the eVinci$^{\text{TM}}$ microreactor with fresh fuel and control drum angles uniform randomly sampled between 0° and 180° and with active generations (g) uniformly randomly sampled between 1000 and 5000 with the neutrons per generation fixed at 20,000 and inactive generations fixed at 25. This active generation sampling range results in σ_f ranging from about 10 pcm to about 30 pcm. The control drum angle is modified across all models to simulate how these angles would be different for the iterative calculations performed in a critical search but the relationship between g and σ_f described by Eq. (9) should remain similar across all these control drum angles. The left portion of Fig. 5 is included to show strength in the linear relationship between $\ln(g)$ and $\ln(\sigma_f)$ for the 71 calculations as assumed in Eq. (9). Linear regression is used to fit a line to this relationship and the $\sigma_f \propto 1/\sqrt{g}$ theoretical convergence in Monte Carlo methods (assumed in the simulated surrogate runtime model given by Eq. (11)) is observed by the −0.50 slope of this line.

In order to simulate the use of Eq. (9) to calculate g_{n+1}, $n+2$ data points (each data point is a pair of g and σ_f) are selected from the pool of 80 calculations. The first $n+1$ of these data points are used based on Eq. (9) to predict the uncertainty ($\sigma_{f,n+1}$) of the remaining point given the number of generations (g_{n+1}). Of course, this is the inverse of Eq. (9) (which can be generated analytically by solving for $\sigma_{f,n+1}$) but it leads to more a more intuitive presentation of the results. Then, the predicted σ_f is compared to the σ_f reported by Serpent 2. This straightforward procedure is done 500 times each for $n = 1$, where 2 points are used in the prediction of $\sigma_{f,2}$, through to $n = 8$, where 9 points are used in the prediction of $\sigma_{f,9}$. The mean absolute error in the predicted number of generations across the 500 trials for each value of n is presented in the right portion of Fig. 5. The mean absolute error (MAE) in the prediction of σ_f tends to be small, around 2 pcm or less. Although this may lead to some small inefficiencies in the search routine, the $\sigma_{f,n} < \sigma_{final}$ convergence criteria is checked after the Serpent 2 calculation using the $\sigma_{f,n}$ provided by the code. Which means the inaccuracies in the use of Eq. (9) will not affect the enforcement of this convergence criteria.

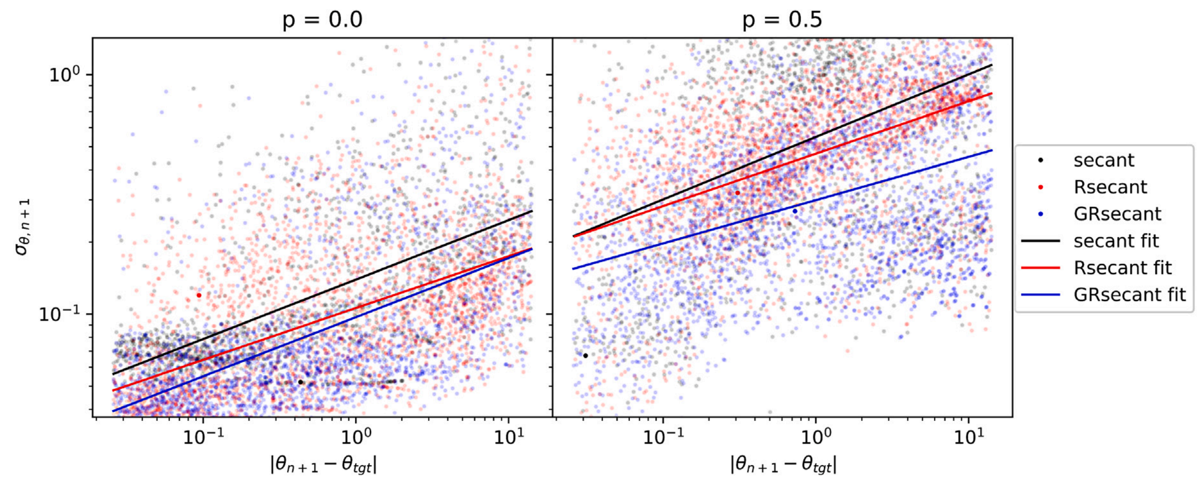

Fig. 4. Estimated uncertainty in next search location selection versus proximity to a known control drum angle for a set of 70 independent search routines where $\sigma_{final} = 5$ pcm and $k_{tol} = 5$ pcm. Linear regression is used to demonstrate differences in the mean behavior of the different search methods.

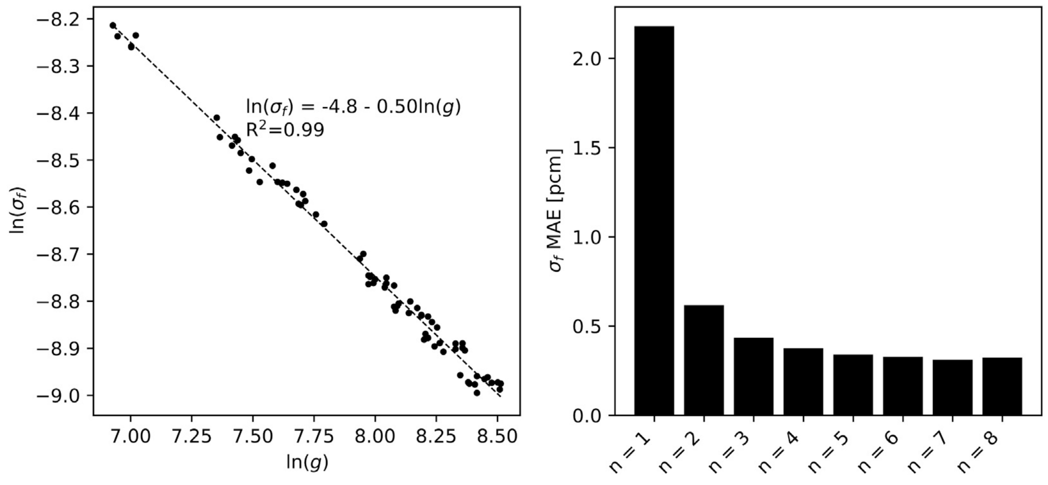

Fig. 5. Data points used to evaluate validity of linear relationship between $\ln(g)$ and $\ln(\sigma_f)$ assumed in Eq. (9) (left). Average error in using 500 sets of $n+2$ data points to evaluate the effectiveness of Eq. (9) (right).

3.3. Performance of secant method across surrogate models

To begin analyzing the results from tests run with surrogate models, the secant method is used to perform many search routines for each of the surrogate models. For each surrogate models, two sets of 100,000 independent search routines are performed to find θ where $k_{tgt} = 1$. $p = 0$ and $p = 0.5$ are used to simulate two $\sigma_{f,n+1}$ selection strategies. For each of the independent search routines, θ_0 and θ_1 are uniformly randomly selected between 0° and 180°, $\sigma_{final} = 20$ pcm, $k_{tol} = 20$ pcm, $s = 0$, $\sigma_{f,0}$ and $\sigma_{f,1}$ are uniformly randomly selected between σ_{final} and $5\sigma_{final}$. The differences in each run is introduced in the selections of θ_0, θ_1, $\sigma_{f,0}$ and $\sigma_{f,1}$ as well as the noise inserted into each evaluation of the surrogate models to simulate the Monte Carlo calculation method. This large set of independent runs is performed to demonstrate the performance of a search method across these parameters which are likely to be arbitrarily selected by the analyst. Fig. 6 shows the mean simulated runtime and mean number of iterations for convergence across the 100,000 search routines is shown for each surrogate model for each value of p. The standard deviations in these simulated runtimes and number of iterations across the search routines is also shown as vertical bars in this figure. These bars do not represent an “error” on the mean simulated runtime, with 100,000 independent search routines, any variation between batches of search routines is very small. Using known laws in the scaling of estimated test statistics (the standard deviation in sample statistic mean is given by standard deviation of the population over square-root of sample size), the standard deviation in the estimate for mean simulated runtime is 0.31 h.

Based on the large bars which represent the standard deviation in runtime, it is clear that there is large variation in search routine runtime across the 100,000 search routines. This suggests that, although there are certainly differences in the *mean* runtime across different surrogate models with different p values, within the costly single routine that may be run with full Serpent 2 calculations, there is no guarantee for a lower runtime using a particular value of p over another. Nevertheless, for these results based on the secant method, $p = 0$ leads to lower runtimes than $p = 0.5$. The number of iterations until convergence is significantly higher for the search routines run with $p = 0.5$ because the f evaluations in these routines are faster, with higher σ_f. This makes the searches with higher values of p require more, shorter, calculations. An explanation for this is that the secant method is not created to efficiently deal with noise in model evaluations so the reduced noise obtained by running all model evaluations to $0.95\sigma_{final}$ is worth the extra runtime required per evaluation of f. Furthermore, the simulated computational time required to perform a search routine is roughly independent of which surrogate model is used. To some degree, this is an artifact of the similarity in the shape of the control drum worth curves for each model shown in Fig. 3 as well as the use of the same runtime model across all surrogates. As mentioned previously,

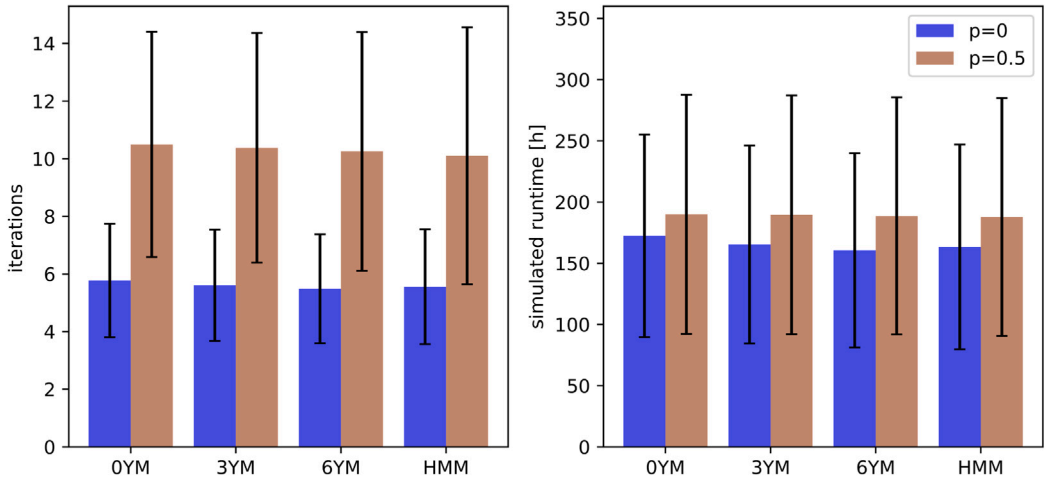

Fig. 6. Mean simulated runtime and mean number of iterations for convergence across the 100,000 search routines for each surrogate model for two values of p. Vertical bars show the standard deviations in these quantities across the search routines.

although simulated runtimes may vary across surrogate models for real Serpent 2 model evaluations, the order of convergence remains the same. Therefore, the consistency in the superior performance of $p = 0$ over $p = 0.5$ across all surrogate models for these routines using the secant method is notable. This was confirmed by the authors by changing the coefficient of 950 $\left[\text{pcm}\sqrt{\text{min}}\right]$ used in the runtime model provided in Eq. (11) to a few different values and regenerating the results. All runtime results were simply scaled up, or down, by a factor. The relative relationships between the runtimes for routines using $p = 0$ and those using $p = 0.5$ remained unchanged.

3.4. Methods comparison

The purpose of this section is to demonstrate differences in average runtime between the secant, Rsecant and GRsecant methods. A number of independent search routines are used for the Rsecant and GRsecant methods for different values of R. As mentioned in Section 2.1.2, when $R = 0$, the secant method and Rsecant method are equivalent; the same goes for GRsecant when $R = 0$. The same testing framework of 100,000 independent search routines is used where θ_0 and θ_1 are uniformly randomly selected between 0° and 180°, $\sigma_{final} = 20$ pcm, $k_{tol} = 20$ pcm, $s = 0$, $\sigma_{f,0}$ and $\sigma_{f,1}$ are uniformly randomly selected between σ_{final} and $5\sigma_{final}$. In these search routines, $p = 0.5$. Fig. 7 shows the mean simulated runtimes with standard deviations across the 100,000 independent search routines as they are performed with different R values for both the Rsecant and GRsecant methods. The 3YM and HMM surrogate models are arbitrarily selected as the results are relatively unaffected by surrogate model selection. Also, the performance of the secant method is shown by replacing the results from the Rsecant and GRsecant methods when $R = 0$. As mentioned previously, when $R = 0$, the Rsecant and GRsecant methods are identical to the secant method.

The mean search routine runtimes for the Rsecant and GRsecant methods can be faster than those for the secant method, given a proper selection for R. However, note that $p = 0.5$ in these search routines which is not an optimal p selection for the secant method. Nevertheless, for the Rsecant method, the most efficient search routine runtimes are observed for $R = 1$ and $R = 2$. The simulated search routine runtimes for the GRsecant method, which demonstrates the best performance, tend to be the same for $R > 1$. The tendency to favor evaluations with lower σ_f is built into the GRsecant method. Therefore, as σ_f is decreased during the search, the GRsecant method will naturally favor these more recent search locations—which will also tend to be closer to the critical drum position. As such, the tendency for the GRsecant method to more slowly converge to an optimal location due to the consideration of past search locations is minimized. These conclusions do not change between surrogate models. Moving forward, the Rsecant method and GRsecant methods will only be used with $R = 2$ because this is where their best performance is observed.

3.5. Selection of p

In the previous section, methods for selecting the next search location have been analyzed. Here, the method for selecting the uncertainty with which to evaluate the next search location is explored by analyzing the effect that p in Eq. (8) has on search routine runtime. As mentioned previously, lower values of p will result search routines with in fewer iterations which are run to lower uncertainty (σ_f). With higher values of p, more calculations are required but these calculations should run faster because of the higher uncertainty. Fig. 8 shows the mean simulated runtimes across 20,000 independent search routines with varying p between 0 and 1 for the 3YM and HMM surrogate models. These independent search routines are run with the same testing framework as the previous two sections. The noise in these trends arises from the finite number of independent search routines used to estimate the mean search routine runtime for a select value of p; in the limit where an infinite number of independent search routines are run, this noise will disappear. In these results for all three methods, there is a small dip in the mean simulated runtimes that occurs for small values of $p \approx 0.02$. It occurs because of an interaction between the 0.95 factor in Eq. (8) and the tendency for some search routines to “miss” the $\sigma_f < \sigma_{final}$ convergence criteria by a small margin on the same iteration the $|f(\theta_n)| < k_{tol}$ criteria is satisfied. Nevertheless, although the lowest mean runtime for the secant method occurs in this dip, the exact location of this minima is difficult to predict across surrogate models. Therefore, in the interest of generality of the optima search routine options for differing control drum problems, values of p close to 0 (but not 0) will not be considered. This disregard has no impact on the selection best search routine configuration as it occurs when $p \approx 0.5$ for the GRsecant method for both surrogate models. The Rsecant method demonstrates inferior performance compared to the GRsecant method for all values of p but demonstrates its best performance at $p \approx 0.3$.

So far, the time required for code startup has been neglected in these analyses. In reality, some portion of the total time required to run a calculation with a Monte Carlo transport code will go towards tasks such as geometry plotting and cross section processing. Depending on the model, this startup time can be significant and may have an impact on the optimal selection for p in these search routines. To explain, if higher values of p require more evaluations of f then the consideration of startup time may hurt these methods because more time will be dedicated to geometry plotting compared to search routines with fewer

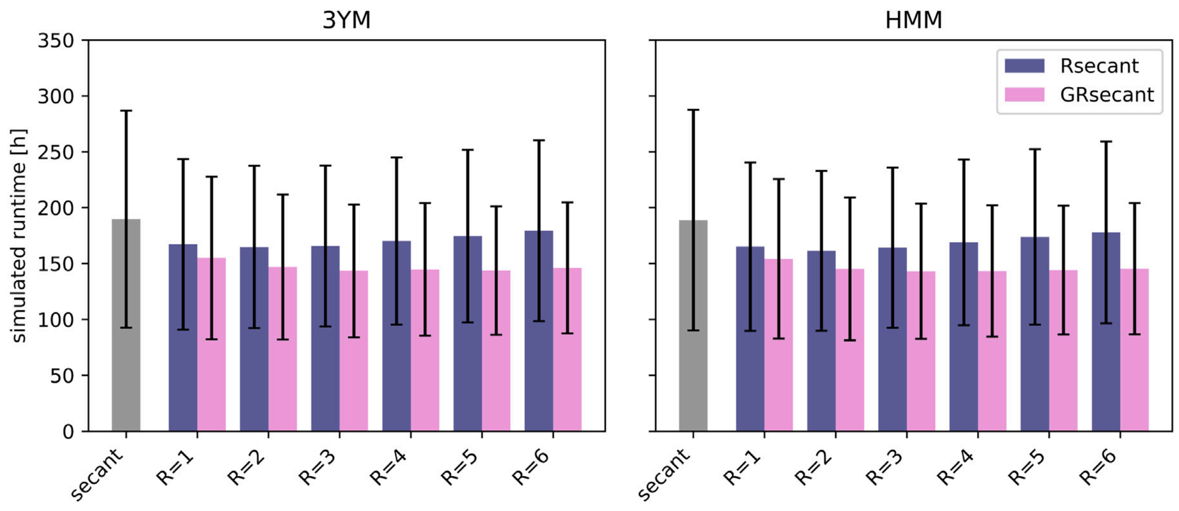

Fig. 7. Mean simulated runtimes for Rsecant and GRsecant methods for 100,000 independent search routines with different values for R. Black vertical bars show the standard deviations of simulated runtimes across independent search routines.

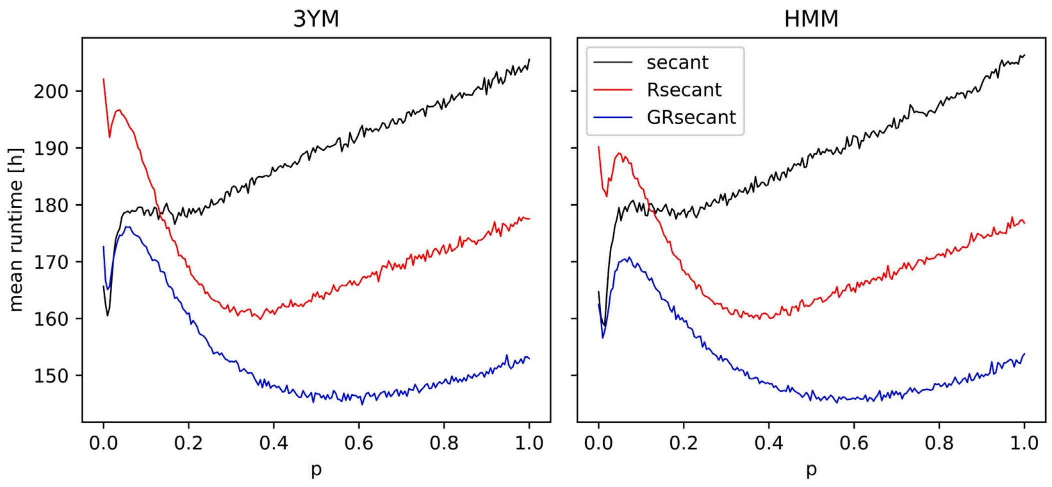

Fig. 8. Mean simulated runtime across 20,000 independent search routines for varying values of p for 3 search methods. Here code startup time (s) is taken to be 0 min in the simulated code runtime model.

overall f evaluations. To explore the optimal p values under consideration of this effect, the same procedure is performed as explained in the previous paragraph but an additional 200 min ($s = 200$) is added to the simulated runtimes of any evaluation of f. The results of this analysis is given in Fig. 9. This number is roughly based on an upper limit for the time required for the eVinci model$^{\text{TM}}$. Although this additional consideration does not change the conclusion that the GRsecant method with $R = 2$ and $p \approx 0.5$ is still the best-performing search routine configuration, the secant method with $p = 0$ demonstrates stronger relative performance. Moving forward, the secant method with $p = 0$, the Rsecant method with $p = 0.3$ and $R = 2$ and the GRsecant methods with $p = 0.5$ and $R = 2$ will be considered as the optimal options for each of these search strategies.

3.6. Effect of cosine transformation

In Section 2.1, each search location selection method is presented with forms given in Eqs. (3), (5) and (7) which use a cosine transformation to linearize the relationship between k_{eff} and control drum rotational position relationship. The degree of linearization this transformation achieves can be seen in Fig. 3 where k_{eff} versus control drum angle is shown for all four surrogate models with, and without, a cosine transformation applied to the x-axis.

In order to evaluate the effectiveness of this transformation, the same testing framework used in the previous sections is used to compare simulated runtimes of search routines across all four surrogate models as shown in Fig. 10. The secant method with $p = 0$, the Rsecant method with $p = 0.5$ and $R = 2$ and the GRsecant methods with $p = 0.5$ and $R = 2$ are used. Although there is some minor reduction in average runtimes when using the cosine transformation, this reduction is not consistent. In particular, for the 3YM, there is an increase in mean simulation runtimes when using the cosine transformation for the Rsecant and GRsecant methods. Also, the standard deviation in runtimes, shown as black vertical lines in these plots, has no noticeable change from the transformation. The small effect caused by this transformation is likely due to the fact that the untransformed cosine-shapes of the control drum worth curves tend to be relatively linear anyways from about 30° to about 150° degrees—the critical drum positions for all four of the surrogate models shown in this figure. The majority of the search routines will focus in this range. Furthermore, this behavior will likely be seen for most applications because critical drum position, even at beginning of life is likely to be above 30 ° due to the need for a sufficient shutdown margin and the control drums will only be above 150° for a short time at end of life. With these results, the performance improvements gained from the application of the cosine transformation may not warrant the extra complexity added to the implementation.

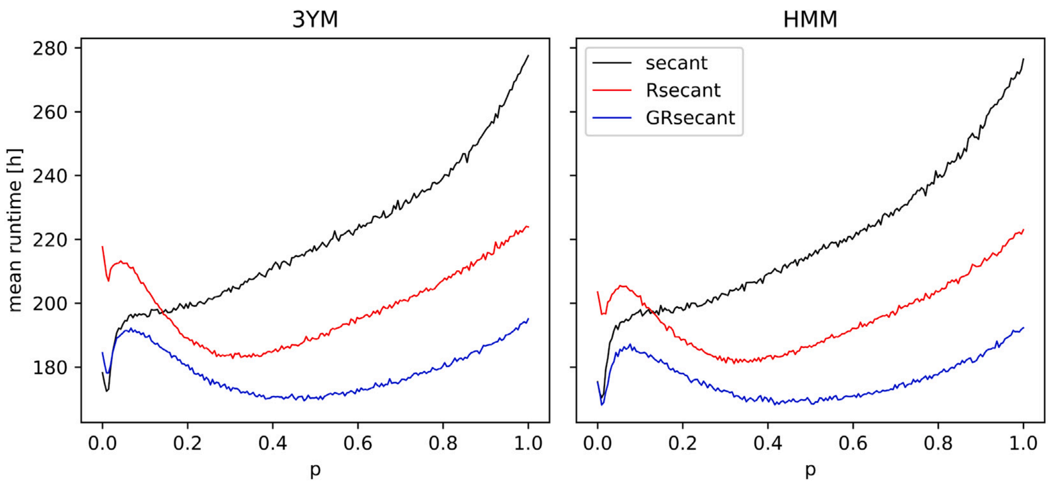

Fig. 9. Mean simulated runtime across 20,000 independent search routines for varying values of p for 3 search methods. Here code startup time (s) is taken to be 200 min in the simulated code runtime model. This value of a may be considered an extreme upper limit.

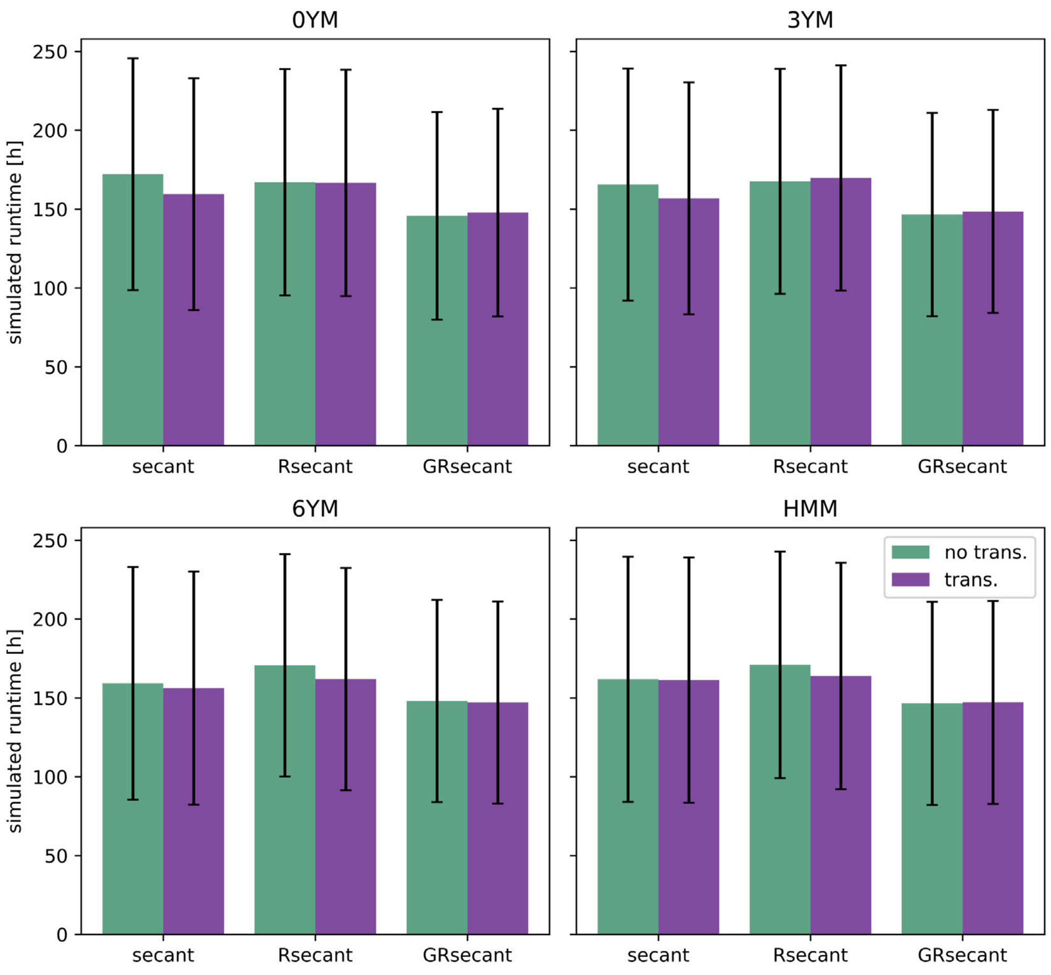

Fig. 10. Mean simulated runtimes across surrogate models, with and without cosine transformation used in Eqs. (3), (5) and (7). Black vertical lines show the standard deviations of simulated runtimes across independent search routines.

3.7. Method selection and k_{tol} and σ_{final}

So far, exploration on the optimal search routine configuration has only been done with σ_{final} = 20 pcm and k_{tol} = 20 pcm. In practice, analysts may select values for σ_{final} and k_{tol} different from these values and this may affect which of the discussed methods is optimal. As such, mean simulated runtimes of 100,000 search routines performed on two surrogate models is shown in Table 1. These search routines were performed similarly to those described in previous sections where θ_0 and θ_1 are uniformly randomly selected between 0° and 180°, $s = 0$ and $\sigma_{f,0}$ and $\sigma_{f,1}$ are uniformly randomly selected between σ_{final} and $5\sigma_{final}$. Values of σ_{final} and k_{tol} both ranging from 40 pcm to 10 pcm

Table 1
Mean simulated runtimes (in hours) across 100,000 independent search routines for different selections of σ_{final} and k_{tol}.

	3YM			HMM		
	secant	Rsecant	GRsecant	secant	Rsecant	GRsecant
k_{tol} = 40 pcm, σ_{final} = 40 pcm	40.1	40.2	36.5[a]	40	40.1	36.5[a]
k_{tol} = 40 pcm, σ_{final} = 20 pcm	86.8	82.6	72.6[a]	85.5	83.2	72.9[a]
k_{tol} = 40 pcm, σ_{final} = 5 pcm	479	421	353[a]	466	431	361[a]
k_{tol} = 20 pcm, σ_{final} = 40 pcm	51.5	53.8	50.5[a]	51.4	53.7	50.3[a]
k_{tol} = 20 pcm, σ_{final} = 20 pcm	105	104	93.6[a]	104	103	93.6[a]
k_{tol} = 20 pcm, σ_{final} = 5 pcm	515	450	386[a]	502	463	390[a]
k_{tol} = 5 pcm, σ_{final} = 40 pcm	93.3[a]	109	105	93.4[a]	109	105
k_{tol} = 5 pcm, σ_{final} = 20 pcm	176[a]	192	182	174[a]	191	182
k_{tol} = 5 pcm, σ_{final} = 5 pcm	689	650	585[a]	677	651	586[a]
Total	2	0	7	2	0	7

[a]Indicate the best performing search configuration for a select σ_{final}, k_{tol} and surrogate model.

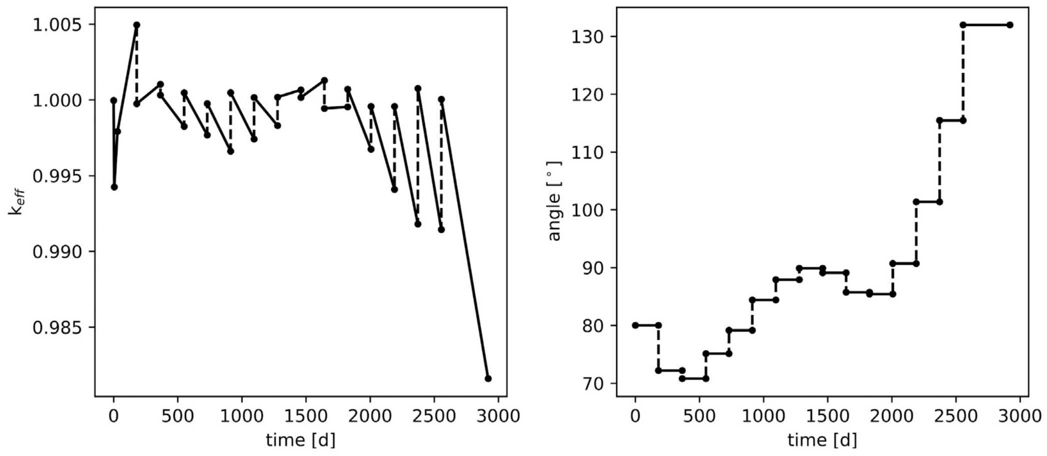

Fig. 11. Core criticality (k_{eff}) over lifetime when using critical control drum positions. Control drum position shown as piecewise constant, the transitions between two control drum positions happens at any burnup step where a critical search routine is performed to find a new control drum position.

are selected and the best performing search routine configuration for each surrogate model for each selection of σ_{final} and k_{tol} is indicated by an asterisk. Then, the total number of times each search configuration reported the lowest runtime for a given surrogate model and selection for σ_{final} and k_{tol} is given in a final row labeled "Total". The columns labeled "secant" correspond to mean runtimes with the secant method where $p = 0$. The columns labeled "Rsecant" and "GRsecant" correspond to mean runtimes with the named method with $R = 2$ and $p = 0.3$ or $p = 0.5$, respectively. From these results, the GRsecant method shows the best performance out of the 3 search routine configurations. The secant method performs comparatively better for low values of k_{tol} but higher values for σ_{final}. In practice, it is unlikely that significantly different values of σ_{final} and k_{tol} will be used, these values are only reported here for completeness. When both σ_{final} and k_{tol}, the GRsecant method demonstrates superior performance.

3.8. Brief presentation of application on eVinci^{TM} microreactor

So far, all results pertaining to search routines have employed surrogate models. It may be useful to demonstrate the methodologies discussed on a Serpent 2 model of the eVinci^{TM} microreactor where a search routine is performed at multiple points during depletion such that the reactor is depleted with the control drums in approximately critical positions following the framework presented in Fig. 2. As these calculations are computationally expensive, only a single depletion calculation (with critical searches performed at multiple depletion step) will be performed using the GRsecant method with $R = 2$ and $p = 0.5$. The tolerances used for each critical search is k_{tol} = 20 pcm and σ_{final} = 20 pcm. Fig. 11 shows k_{eff} as a function of time resulting from this process. Initially, the reactor starts as critical because a critical drum position search is performed for the fresh core. Then, three burnup steps are performed and another critical search routine is performed. As the next depletion step is initiated with a different control drum position than the previous one which causes a discontinuity in the k_{eff} versus time plot. Then, critical search routines are performed at each depletion step until the end of the calculation. Fig. 11 also shows the position of the control drums as a function of time. The control drum positions are represented by a piecewise constant function over time, the transitions between two control drum positions happens at any burnup step where a critical search routine is performed to find a new control drum position. In practice, any further analysis requiring critical control drum positions should not require any search algorithms. Instead, these control drum positions as a function of time can be incorporated into models using surface rotation transformations.

4. Conclusion

In this manuscript, a method for finding critical drum positions for microreactors with Monte Carlo codes is presented. Although a few different variations on the search method were evaluated using surrogate models for drum worths with simulated run times, a single configuration is identified as the superior search routine configuration. That is, the GRsecant method where $R = 2$ and $p = 0.5$ with no cosine transformation. When setting up the scripting framework to implement this methodology only 3 mathematical equations need to be programmed:

1. Eq. (6) with $R = 2$ is used to determine the next control drum angle search point.

2. Eq. (8) with $p = 0.5$ is used to determine the Monte Carlo uncertainty which the next search point should be calculated with.
3. Eq. (9) is used to determine the number of generations to use in the Monte Carlo calculation to achieve the Monte Carlo uncertainty dictated by Eq. (8).

Although this paper provides thorough derivations and testing, this methodology was created with ease-of-implementation as a significant motivator such that it can be implemented using any neutronics code capable of performing burnup calculations can be used without the need for source code access. Furthermore, the in-depth comparisons between search method variants (secant, Rsecant and GRsecant) which are performed in this paper are certainly not necessary to implement the search routine. They are included to provide some justification for the exact form of the search suggested in the preceding text. Despite the exploration included in this paper, more optimal search routine configurations may exist for a specific reactor design but it is expected that the extra effort one would need to invest to find this configuration may not be worth the computational time saved as typically, finding burnup-dependent control drum positions should only need to be performed once for a particular reactor design. Also, the optimal search strategy did not change between surrogate models based on the eVinci$^{\mathrm{TM}}$ and Holos-Quad designs. Finally, the results from the application of this methodology to the eVinci$^{\mathrm{TM}}$ microreactor is included to demonstrate the method on a real reactor model, as opposed to surrogate models. Given the competitive industry developing around microreactor deployment, the method described in this paper should give engineers an easy-to-implement tool to find burnup-dependent critical drum positions. Future work may seek to incorporate thermal feedback into this critical position search algorithm.

CRediT authorship contribution statement

Dean Price: Conceptualization, Methodology, Software, Formal analysis, Investigation, Writing – original draft, Visualization. **Nathan Roskoff:** Conceptualization, Methodology, Software, Validation, Writing – review & editing, Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The authors do not have permission to share data.

Acknowledgments

This material is based on work supported under a Department of Energy, Office of Nuclear Energy, Integrated University Program Graduate Fellowship.

Appendix A. Derivation for secant, Rsecant and GRsecant method

This section includes derivations for the expressions for the next search locations for the various iterative root-finding algorithms discussed in this paper.

A.1. Secant method

The derivation for this method is straightforward, a line is fit to two points and the search location is selected to be where that line intersects with 0. To begin, we consider the two most recent search locations, θ_n and θ_{n-1}, and the value of the objective function at these locations, $f(\theta_n)$ and $f(\theta_{n-1})$. An equation for a line in point-slope form which passes through these two points is:

$$f(\theta) - f(\theta_n) = \frac{f(\theta_n) - f(\theta_{n-1})}{\theta_n - \theta_{n-1}}(\theta - \theta_n). \tag{A.1}$$

The $n+1$-th search location is selected such that $f(\theta_{n+1}) = 0$ such that:

$$0 - f(\theta_n) = \frac{f(\theta_n) - f(\theta_{n-1})}{\theta_n - \theta_{n-1}}(\theta_{n+1} - \theta_n). \tag{A.2}$$

By solving for θ_{n+1}:

$$\theta_{n+1} = -f(\theta_n)\frac{\theta_n - \theta_{n-1}}{f(\theta_n) - f(\theta_{n-1})} + \theta_n, \tag{A.3}$$

$$\theta_{n+1} = \frac{\theta_{n-1}f(\theta_n) - \theta_n f(\theta_n)}{f(\theta_n) - f(\theta_{n-1})} + \theta_n, \tag{A.4}$$

$$\theta_{n+1} = \frac{\theta_{n-1}f(\theta_n) - \theta_n f(\theta_n)}{f(\theta_n) - f(\theta_{n-1})} + \frac{\theta_n f(\theta_n) - \theta_n f(\theta_{n-1})}{f(\theta_n) - f(\theta_{n-1})}, \tag{A.5}$$

$$\theta_{n+1} = \frac{\theta_{n-1}f(\theta_n) - \theta_n f(\theta_{n-1})}{f(\theta_n) - f(\theta_{n-1})}. \tag{A.6}$$

A.2. Rsecant method

The derivation for the Rsecant method begins by using linear regression to fit a line to $R+2$ points backward in the search history. To do this, a matrix Θ is defined as:

$$\Theta = \begin{bmatrix} 1 & \theta_n \\ 1 & \theta_{n-1} \\ \vdots & \vdots \\ 1 & \theta_{n-R-1} \end{bmatrix}. \tag{A.7}$$

A vector $\boldsymbol{y}$ is defined as:

$$\boldsymbol{y} = \begin{bmatrix} f(\theta_n) & f(\theta_{n-1}) & \dots & f(\theta_{n-R-1}) \end{bmatrix}^T. \tag{A.8}$$

Then, a line which minimizes the sum-of-squared squared distances from the $R+2$ points of the form $f(\theta) = a + b\theta$ can be found with the normal equations with:

$$\beta = (\Theta^T\Theta)^{-1}\Theta^T\boldsymbol{y} \tag{A.9}$$

where $\beta = \begin{bmatrix} a & b \end{bmatrix}^T$. Expressions for a and b can be written without matrix equations as:

$$a = \frac{\sum_{i=0}^{R+1} f(\theta_{n-i}) \sum_{i=0}^{R+1} \theta_{n-i}^2 - \sum_{i=0}^{R+1} \theta_{n-i} f(\theta_{n-i}) \sum_{i=0}^{R+1} \theta_{n-i}}{(R+2)\sum_{i=0}^{R+1} \theta_{n-i}^2 - \left(\sum_{i=0}^{R+1} \theta_{n-i}\right)^2}, \tag{A.10}$$

$$b = \frac{(R+2)\sum_{i=0}^{R+1} \theta_{n-i} f(\theta_{n-i}) - \sum_{i=0}^{R+1} f(\theta_{n-i}) \sum_{i=0}^{R+1} \theta_{n-i}}{(R+2)\sum_{i=0}^{R+1} \theta_{n-i}^2 - \left(\sum_{i=0}^{R+1} \theta_{n-i}\right)^2}. \tag{A.11}$$

By noting that the θ-coordinate of the intersection of a line expressed as $a + b\theta$ with the θ axis is $-\frac{a}{b}$, the next search location is:

$$\theta_{n+1} = \frac{\sum_{i=0}^{R+1} \theta_{n-i} f(\theta_{n-i}) \sum_{i=0}^{R+1} \theta_{n-i} - \sum_{i=0}^{R+1} f(\theta_{n-i}) \sum_{i=0}^{R+1} \theta_{n-i}^2}{(R+2)\sum_{i=0}^{R+1} \theta_{n-i} f(\theta_{n-i}) - \sum_{i=0}^{R+1} f(\theta_{n-i}) \sum_{i=0}^{R+1} \theta_{n-i}}. \tag{A.12}$$

A.3. GRsecant method

This method also begins by using linear regression to fit a line to $R+2$ points backward in the search history, however instead of using the normal equations which minimizes the sum-of-squared squared distances from the $R+2$ points, a different fitting criteria is used. That is, instead of minimizing

$$R(a,b)=\sum_{i=0}^{R+1}\left(f(\theta_{n-i})-a-b\theta_{n-i}\right)^2, \tag{A.13}$$

the goal is to minimize

$$R(a,b)=\sum_{i=0}^{R+1}\frac{\left(f(\theta_{n-i})-a-b\theta_{n-i}\right)^2}{\sigma_{f,n-i}^2} \tag{A.14}$$

where $\sigma_{f,n}$ is the standard deviation representing the uncertainty in the evaluation of $f(\theta_n)$ due to the Monte Carlo calculation method. Obtaining $\sigma_{f,n}$ is relatively straightforward. Based on Eq. (1) the uncertainty in any evaluation of f is the same as the uncertainty in any model evaluation of k. The Serpent 2 neutronics code (along with most others) report an estimate for the uncertainty in model evaluations of k with code output such that $\sigma_{f,n}$ can be directly obtained from code output. The values of a and b which minimize $R(a,b)$ can be found using the normal equations:

$$\beta=\left(\Theta^T W\Theta\right)^{-1}\Theta^T W\boldsymbol{y} \tag{A.15}$$

where $\beta=\begin{bmatrix}a & b\end{bmatrix}^T$,

$$W=\begin{bmatrix}\frac{1}{\sigma_{f,n}^2} & \cdots & & 0\\ \vdots & \frac{1}{\sigma_{f,n-1}^2} & & \vdots\\ & & \ddots & \\ 0 & \cdots & & \frac{1}{\sigma_{f,n-R-1}^2}\end{bmatrix} \tag{A.16}$$

and Θ and $\boldsymbol{y}$ follow the definitions given in Eqs. (A.7) and (A.8), respectively. Expressions for a and b can be written without matrix equations as:

$$a=\frac{\sum_{i=0}^{R+1}\frac{\theta_{n-i}^2}{\sigma_{f,n-i}^2}\sum_{i=0}^{R+1}\frac{f(\theta_{n-i})}{\sigma_{f,n-i}^2}-\sum_{i=0}^{R+1}\frac{\theta_{n-i}}{\sigma_{f,n-i}^2}\sum_{i=0}^{R+1}f(\theta_{n-i})\frac{\theta_{n-i}}{\sigma_{f,n-i}^2}}{\sum_{i=0}^{R+1}\frac{1}{\sigma_{f,n-i}^2}\sum_{i=0}^{R+1}\frac{\theta_{n-i}^2}{\sigma_{f,n-i}^2}-\left(\sum_{i=0}^{R+1}\frac{\theta_{n-i}}{\sigma_{f,n-i}^2}\right)^2} \tag{A.17}$$

$$b=\frac{\sum_{i=0}^{R+1}f(\theta_{n-i})\frac{\theta_{n-i}}{\sigma_{f,n-i}^2}\sum_{i=0}^{R+1}\frac{1}{\sigma_{f,n-i}^2}-\sum_{i=0}^{R+1}\frac{f(\theta_{n-i})}{\sigma_{f,n-i}^2}\sum_{i=0}^{R+1}\frac{\theta_{n-i}}{\sigma_{f,n-i}^2}}{\sum_{i=0}^{R+1}\frac{1}{\sigma_{f,n-i}^2}\sum_{i=0}^{R+1}\frac{\theta_{n-i}^2}{\sigma_{f,n-i}^2}-\left(\sum_{i=0}^{R+1}\frac{\theta_{n-i}}{\sigma_{f,n-i}^2}\right)^2}. \tag{A.18}$$

Again, the θ-coordinate of the intersection of a line expressed as $a+b\theta$ with the θ axis is $-\frac{a}{b}$. Therefore, the next search location is:

$$\theta_{n+1}=\frac{\sum_{i=0}^{R+1}\frac{\theta_{n-i}}{\sigma_{f,n-i}^2}\sum_{i=0}^{R+1}f(\theta_{n-i})\frac{\theta_{n-i}}{\sigma_{f,n-i}^2}-\sum_{i=0}^{R+1}\frac{\theta_{n-i}^2}{\sigma_{f,n-i}^2}\sum_{i=0}^{R+1}\frac{f(\theta_{n-i})}{\sigma_{f,n-i}^2}}{\sum_{i=0}^{R+1}f(\theta_{n-i})\frac{\theta_{n-i}}{\sigma_{f,n-i}^2}\sum_{i=0}^{R+1}\frac{1}{\sigma_{f,n-i}^2}-\sum_{i=0}^{R+1}\frac{f(\theta_{n-i})}{\sigma_{f,n-i}^2}\sum_{i=0}^{R+1}\frac{\theta_{n-i}}{\sigma_{f,n-i}^2}}. \tag{A.19}$$

Appendix B. Uncertainty in search location for secant, Rsecant and GRsecant methods

This section gives the uncertainty in the next search location which arises from the uncertainties in previous evaluations of $f(\theta)$ due to the Monte Carlo calculation method. Before deriving expressions for the uncertainty in the next search location for each of the methods, it is useful to provide some discussion on a mathematical operation which will show up in all derivations. Consider two random variables, $X\sim N(\mu_X,\sigma_X^2)$ and $Y\sim N(\mu_Y,\sigma_Y^2)$ which have a known covariance, σ_{XY}. The goal of this discussion is to find approximate expressions for $\mathrm{E}\left[\frac{Y}{X}\right]$ and $\mathrm{Var}\left[\frac{Y}{X}\right]$ in terms of μ_X, σ_X, μ_Y, σ_Y and σ_{XY}. The inversion of X can be problematic because it is a nonlinear operation (exponentiation to the power of -1). This is further complicated because the probability density function for $1/X$ is not finite. Therefore, a Taylor expansion about μ_X is used to approximate the inversion of X and it is important that $\frac{\sigma_X}{\mu_X}$ is small such that this approximation is accurate. Then, it can be assumed that $\frac{\sigma_X}{\mu_X}$ is normally distributed with the provided mean and variance.

To begin, write $\frac{Y}{X}$ using a Taylor expansion of $\frac{1}{X}$ about μ_X:

$$\frac{Y}{X}\approx\sum_{t=0}^{T}\frac{(-1)^t}{\mu_X^{t+1}}Y(X-\mu_X)^t. \tag{B.1}$$

The expectation will be treated first:

$$\mathrm{E}\left[\frac{Y}{X}\right]\approx\mathrm{E}\left[\sum_{t=0}^{T}\frac{(-1)^t}{\mu_X^{t+1}}Y(X-\mu_X)^t\right] \tag{B.2}$$

$$\mathrm{E}\left[\frac{Y}{X}\right]\approx\sum_{t=0}^{T}\frac{(-1)^t}{\mu_X^{t+1}}\mathrm{E}\left[Y(X-\mu_X)^t\right]. \tag{B.3}$$

It is nontrivial to find an expression for $\mathrm{E}\left[Y(X-\mu_X)^t\right]$ so it will be discussed here. First, consider that X and Y can be rewritten in terms of two other random variables A and B which are independent from eachother and normally distributed with means equal to 0 and variances equal to 1 as

$$X=\mu_X+\sigma_X A, \tag{B.4}$$

$$Y=\mu_Y+\frac{\sigma_{XY}}{\sigma_X}A+\sqrt{\sigma_Y^2-\frac{\sigma_{XY}^2}{\sigma_X^2}}B. \tag{B.5}$$

Then,

$$\mathrm{E}\left[Y(X-\mu_X)^t\right]=\mathrm{E}\left[\left(\mu_Y+\frac{\sigma_{XY}}{\sigma_X}A+\sqrt{\sigma_Y^2-\frac{\sigma_{XY}^2}{\sigma_X}}B\right)(\mu_X+\sigma_X A-\mu_X)^t\right] \tag{B.6}$$

$$=\mathrm{E}\left[\mu_Y\sigma_X^tA^t+\frac{\sigma_{XY}}{\sigma_X}\sigma_X^tA^{t+1}+\sqrt{\sigma_Y^2-\frac{\sigma_{XY}^2}{\sigma_X}}B\sigma_X^tA^t\right] \tag{B.7}$$

$$=\mu_Y\sigma_X^t\mathrm{E}\left[A^t\right]+\frac{\sigma_{XY}}{\sigma_X}\sigma_X^t\mathrm{E}\left[A^{t+1}\right]+\sigma_X^t\sqrt{\sigma_Y^2-\frac{\sigma_{XY}^2}{\sigma_X}}\mathrm{E}[B]\,\mathrm{E}\left[A^t\right] \tag{B.8}$$

$$=\mu_Y\sigma_X^t\mathrm{E}\left[A^t\right]+\frac{\sigma_{XY}}{\sigma_X}\sigma_X^t\mathrm{E}\left[A^{t+1}\right] \tag{B.9}$$

$\mathrm{E}\left[A^t\right]$ is a well-known quantity associated with a standard normal distribution known as its tth moment and can be expressed as

$$\mathrm{E}\left[A^t\right]=\begin{cases}0 & t\text{ odd}\\ (t-1)!! & t\text{ even.}\end{cases} \tag{B.10}$$

In this expression, !! indicates a double-factorial which is not simply a factorial applied twice. Note that $-1!!=1$ (Arfken and Weber, 1999). Such that

$$\mathrm{E}\left[Y(X-\mu_X)^t\right]=\begin{cases}t!!\sigma_X^{t-1}\sigma_{XY} & t\text{ odd}\\ (t-1)!!\mu_Y\sigma_X^t & t\text{ even.}\end{cases} \tag{B.11}$$

Now, substituting this expression for $\mathrm{E}\left[Y(X-\mu_X)^t\right]$ into Eq. (B.3)

$$\mathrm{E}\left[\frac{Y}{X}\right]\approx\sum_{t=0,t\text{ even}}^{T}\frac{(-1)^t}{\mu_X^{t+1}}(t-1)!!\mu_Y\sigma_X^t+\sum_{t=1,t\text{ odd}}^{T}\frac{(-1)^t}{\mu_X^{t+1}}t!!\sigma_X^{t-1}\sigma_{XY}. \tag{B.12}$$

Before finding an approximate expression for $\mathrm{Var}\left[\frac{Y}{X}\right]$, it is first necessary to find an approximate expression for $\mathrm{E}\left[\frac{Y^2}{X^2}\right]$. Again, approximate the inverse operation contained in $\frac{Y^2}{X^2}$ about the mean of its denominator.

$$\frac{Y^2}{X^2}\approx\sum_{t=0}^{T}\frac{(-1)^t}{\left(\mu_X^2+\sigma_X^2\right)^{t+1}}Y^2(X^2-\mu_X^2-\sigma_X^2)^t. \tag{B.13}$$

This time, the denominator is X^2 which has a mean of $\mathrm{E}\left[X^2\right] = \mu_X^2 + \sigma_X^2$. Then, apply the expectation operator.

$$\mathrm{E}\left[\frac{Y^2}{X^2}\right] \approx \sum_{t=0}^{T} \frac{(-1)^t}{\left(\mu_X^2 + \sigma_X^2\right)^{t+1}} \mathrm{E}\left[Y^2(X^2 - \mu_X^2 - \sigma_X^2)^t\right]. \tag{B.14}$$

As was done previously, the expansions of X and Y provided in Eqs. (B.4) and (B.5), respectively, can be used to find

$$\mathrm{E}\left[Y^2(X^2 - \mu_X^2 - \sigma_X^2)^t\right] = \left(\mu_Y + \sigma_Y - \frac{\sigma_{XY}^2}{\sigma_X}\right)$$

$$\times \mathrm{E}\left[\left(\sigma_X^2\left(A + \frac{\mu_X}{\sigma_X}\right)^2 + (-\mu_X^2 - \sigma_X^2)\right)^t\right] \tag{B.15}$$

$$+ \frac{2\mu_Y \sigma_{XY}}{\sigma_X} \mathrm{E}\left[\left(\sigma_X^2\left(A + \frac{\mu_X}{\sigma_X}\right)^2 + (-\mu_X^2 - \sigma_X^2)\right)^t A\right] \tag{B.16}$$

$$+ \frac{\sigma_{XY}^2}{\sigma_X^2} \mathrm{E}\left[\left(\sigma_X^2\left(A + \frac{\mu_X}{\sigma_X}\right)^2 + (-\mu_X^2 - \sigma_X^2)\right)^t A^2\right]. \tag{B.17}$$

Using binomial expansions,

$$\mathrm{E}\left[\left(\sigma_X^2\left(A + \frac{\mu_X}{\sigma_X}\right)^2 + (-\mu_X^2 - \sigma_X^2)\right)^t\right] = \tag{B.18}$$

$$\sum_{k=0}^{t}\left(\binom{t}{k}\sigma_X^{2k}\left(-\mu_X^2 - \sigma_X^2\right)^{t-k} \sum_{p=0}^{k}\binom{2k}{2p}\left(\frac{\mu_X}{\sigma_X}\right)^{2k-2p} 2^{-p}\frac{(2p)!}{p!}\right), \tag{B.19}$$

$$\mathrm{E}\left[\left(\sigma_X^2\left(A + \frac{\mu_X}{\sigma_X}\right)^2 + (-\mu_X^2 - \sigma_X^2)\right)^t A\right] = \tag{B.20}$$

$$\sum_{k=0}^{t}\left(\binom{t}{k}\sigma_X^{2k}\left(-\mu_X^2 - \sigma_X^2\right)^{t-k} \sum_{p=1}^{k}\binom{2k}{2p-1}\left(\frac{\mu_X}{\sigma_X}\right)^{2k-2p+1} 2^{-p}\frac{(2p)!}{p!}\right) \tag{B.21}$$

and

$$\mathrm{E}\left[\left(\sigma_X^2\left(A + \frac{\mu_X}{\sigma_X}\right)^2 + (-\mu_X^2 - \sigma_X^2)\right)^t A^2\right] = \tag{B.22}$$

$$\sum_{k=0}^{t}\left(\binom{t}{k}\sigma_X^{2k}\left(-\mu_X^2 - \sigma_X^2\right)^{t-k} \sum_{p=0}^{k}\binom{2k}{2p}\left(\frac{\mu_X}{\sigma_X}\right)^{2k-2p} 2^{-p-1}\frac{(2p+2)!}{(p+1)!}\right) \tag{B.23}$$

can be obtained. Finally,

$$\mathrm{Var}\left[\frac{Y}{X}\right] = \mathrm{E}\left[\frac{Y^2}{X^2}\right] - \mathrm{E}\left[\frac{Y}{X}\right]^2 \tag{B.24}$$

where an approximation for $\mathrm{E}\left[\frac{X^2}{Y^2}\right]$ is provided in Eq. (B.14) and an approximation for $\mathrm{E}\left[\frac{Y}{X}\right]$ is provided in Eq. (B.12).

B.1. Secant method

Start by separating the numerator and denominator in Eq. (2) and assigning them to Y and X:

$$X = f(\theta_n) - f(\theta_{n-1}) \tag{B.25}$$

$$Y = \theta_{n-1} f(\theta_n) - \theta_n f(\theta_{n-1}) \tag{B.26}$$

With these simple expressions, algebra of random variables can be used to find:

$$\mu_X = f(\theta_n) - f(\theta_{n-1}) \tag{B.27}$$

$$\mu_Y = \theta_{n-1} f(\theta_n) - \theta_n f(\theta_{n-1}) \tag{B.28}$$

$$\sigma_X^2 = \sigma_{f,n}^2 + \sigma_{f,n-1}^2 \tag{B.29}$$

$$\sigma_Y^2 = \theta_{n-1}^2 \sigma_{f,n}^2 + \theta_n^2 \sigma_{f,n-1}^2 \tag{B.30}$$

$$\sigma_{XY} = \theta_{n-1}\sigma_{f,n}^2 + \theta_n \sigma_{f,n-1}^2. \tag{B.31}$$

With these definitions, $\mathrm{E}\left[\theta_{n+1}\right]$ can be found with the expression for $\mathrm{E}\left[\frac{Y}{X}\right]$ given in Eq. (B.12) and $\mathrm{Var}\left[\theta_{n+1}\right]$ can be found with the expression for $\mathrm{Var}\left[\frac{Y}{X}\right]$ given in Eq. (B.24).

B.2. Rsecant method

For the Rsecant method, linear algebra will be used to make the derivation more concise. To begin, consider Eq. (A.9) with the expectation operator applied to both sides:

$$\mathrm{E}[\beta] = \mathrm{E}\left[(\Theta^T\Theta)^{-1}\Theta^T \boldsymbol{y}\right]. \tag{B.32}$$

All the uncertainties arising from the Monte Carlo calculation method for k_{eff} are contained in $\boldsymbol{y}$, therefore:

$$\mathrm{E}[\beta] = (\Theta^T\Theta)^{-1}\Theta^T \mathrm{E}[\boldsymbol{y}]. \tag{B.33}$$

Furthermore, because each evaluation of f is assumed to be an unbiased estimate, $\mathrm{E}[f(\theta)] = f(\theta)$ so $\mathrm{E}[\boldsymbol{y}] = \boldsymbol{y}$:

$$\mathrm{E}[\beta] = (\Theta^T\Theta)^{-1}\Theta^T \boldsymbol{y}. \tag{B.34}$$

With this, $\mu_Y = -a$ following its definition in Eq. (A.10) and $\mu_X = b$ following its definition in Eq. (A.11).

For this derivation, the variance and covariance will be found using the concept of a covariance matrix, Σ_β. Where

$$\Sigma_\beta \equiv \begin{bmatrix} \sigma_Y^2 & \sigma_{XY} \\ \sigma_{XY} & \sigma_X^2 \end{bmatrix}. \tag{B.35}$$

It is also useful to define another covariance matrix which holds the information on the uncertainty in each evaluation of $f(\theta_n)$

$$\Sigma_{\boldsymbol{y}} \equiv \begin{bmatrix} \sigma_{f,n}^2 & \dots & & 0 \\ \vdots & \sigma_{f,n-1}^2 & & \vdots \\ & & \ddots & \\ 0 & \dots & & \sigma_{f,n-R-1}^2 \end{bmatrix}. \tag{B.36}$$

Here, note that the off-diagonal elements are 0 because each objective function evaluation is performed independently of one-another.

$$\Sigma_\beta = \mathrm{E}\left[\left(\Theta^+\boldsymbol{y}\right)\left(\Theta^+\boldsymbol{y}\right)^T\right] - \mathrm{E}\left[\Theta^+\boldsymbol{y}\right]\mathrm{E}\left[\Theta^+\boldsymbol{y}\right]^T \tag{B.37}$$

$$= \mathrm{E}\left[\Theta^+\boldsymbol{y}\boldsymbol{y}^T\Theta^{+T}\right] - \mathrm{E}\left[\Theta^+\boldsymbol{y}\right]\mathrm{E}\left[\boldsymbol{y}^T\Theta^{+T}\right] \tag{B.38}$$

$$= \Theta^+\mathrm{E}\left[\boldsymbol{y}\boldsymbol{y}^T\right]\Theta^{+T} - \Theta^+\mathrm{E}[\boldsymbol{y}]\,\mathrm{E}\left[\boldsymbol{y}^T\right]\Theta^{+T} \tag{B.39}$$

$$= \Theta^+\left(\mathrm{E}\left[\boldsymbol{y}\boldsymbol{y}^T\right] - \mathrm{E}[\boldsymbol{y}]\,\mathrm{E}\left[\boldsymbol{y}^T\right]\right)\Theta^{+T} \tag{B.40}$$

$$= \Theta^+\Sigma_{\boldsymbol{y}}\Theta^{+T} \tag{B.41}$$

Where $\Theta^+ \equiv (\Theta\Theta)^{-1}\,\Theta^T$. With these equations,

$$\mu_X = (R+2)\sum_{i=0}^{R+1}\theta_{n-i} f(\theta_{n-i}) - \sum_{i=0}^{R+1} f(\theta_{n-i}) \sum_{i=0}^{R+1}\theta_{n-i} \tag{B.42}$$

$$\mu_Y = \sum_{i=0}^{R+1}\theta_{n-i} f(\theta_{n-i}) \sum_{i=0}^{R+1}\theta_{n-i} - \sum_{i=0}^{R+1} f(\theta_{n-i}) \sum_{i=0}^{R+1}\theta_{n-i}^2 \tag{B.43}$$

$$\sigma_X^2 = \Sigma_{\beta,(2,2)} \tag{B.44}$$

$$\sigma_Y^2 = \Sigma_{\beta,(1,1)} \tag{B.45}$$

$$\sigma_{XY} = -\Sigma_{\beta,(1,2)}. \tag{B.46}$$

Here, the notation $\Sigma_{\beta,(i,j)}$ is used to refer to the value in the ith row and jth column of Σ_β. With these definitions, $\mathrm{E}\left[\theta_{n+1}\right]$ can be found with the expression for $\mathrm{E}\left[\frac{Y}{X}\right]$ given in Eq. (B.12) and $\mathrm{Var}\left[\theta_{n+1}\right]$ can be found with the expression for $\mathrm{Var}\left[\frac{Y}{X}\right]$ given in Eq. (B.24).

B.3. GRsecant method

Similarly to the Rsecant method, linear algebra will be used to make the derivation for the GRsecant method more concise. To begin, apply the expectation operator to both sides of Eq. (B.47):

$$\mathrm{E}\left[\beta\right] = \mathrm{E}\left[\left(\Theta^{T} W \Theta\right)^{-1} \Theta^{T} W \boldsymbol{y}\right] \tag{B.47}$$

Following the work provided in the derivation for the Rsecant method,

$$\mathrm{E}\left[\beta\right] = \left(\Theta^{T} W \Theta\right)^{-1} \Theta^{T} W \boldsymbol{y} \tag{B.48}$$

With this, $\mu_Y = -a$ following its definition in Eq. (A.17) and $\mu_X = b$ following its definition in Eq. (A.18).

Also, similarly to the derivation for the GRsecant method, if $\tilde{\Theta}^{+} = \left(\Theta^{T} W \Theta\right)^{-1} \Theta^{T} W$,

$$\Sigma_{\beta} = \tilde{\Theta}^{+} \Sigma_{\boldsymbol{y}} \tilde{\Theta}^{+T}. \tag{B.49}$$

$\tilde{\Theta}^{+}$ is occasionally referred to as a "hat" matrix.

This leads to:

$$\mu_X = \sum_{i=0}^{R+1} f(\theta_{n-i}) \frac{\theta_{n-i}}{\sigma^2_{f,n-i}} \sum_{i=0}^{R+1} \frac{1}{\sigma^2_{f,n-i}} - \sum_{i=0}^{R+1} \frac{f(\theta_{n-i})}{\sigma^2_{f,n-i}} \sum_{i=0}^{R+1} \frac{\theta_{n-i}}{\sigma^2_{f,n-i}} \tag{B.50}$$

$$\mu_Y = \sum_{i=0}^{R+1} \frac{\theta_{n-i}}{\sigma^2_{f,n-i}} \sum_{i=0}^{R+1} f(\theta_{n-i}) \frac{\theta_{n-i}}{\sigma^2_{f,n-i}} - \sum_{i=0}^{R+1} \frac{\theta^2_{n-i}}{\sigma^2_{f,n-i}} \sum_{i=0}^{R+1} \frac{f(\theta_{n-i})}{\sigma^2_{f,n-i}} \tag{B.51}$$

$$\sigma^2_X = \Sigma_{\beta,(2,2)} \tag{B.52}$$

$$\sigma^2_Y = \Sigma_{\beta,(1,1)} \tag{B.53}$$

$$\sigma_{XY} = -\Sigma_{\beta,(1,2)}. \tag{B.54}$$

With these definitions, $\mathrm{E}\left[\theta_{n+1}\right]$ can be found with the expression for $\mathrm{E}\left[\frac{Y}{X}\right]$ given in Eq. (B.12) and $\mathrm{Var}\left[\theta_{n+1}\right]$ can be found with the expression for $\mathrm{Var}\left[\frac{Y}{X}\right]$ given in Eq. (B.24).

B.4. Uncertainty propagation with cosine transformation

Of course, the above discussion on uncertainty propagation is only directly applicable to the untransformed expressions for determining the next search point, θ_{n+1} for each of the three methods. The equivalent uncertainty propagation with the transformed versions of the search methods, that is using Eqs. (3), (5) and (7), is explained in this section. First, the transform should be applied to the values for θ. This is simply done by replacing θ_n and θ_{n-1} (where they are outside of f) with $\cos\left(\theta_n\right)$ and $\cos\left(\theta_{n-1}\right)$ in Eqs. (B.27) through (B.31) for the secant method. For both the Rsecant and GRsecant method, an alternative definition of Θ can be used for the same effect:

$$\Theta = \begin{bmatrix} 1 & \cos\left(\theta_n\right) \\ 1 & \cos\left(\theta_{n-1}\right) \\ \vdots & \vdots \\ 1 & \cos\left(\theta_{n-R-1}\right) \end{bmatrix}. \tag{B.55}$$

Fortunately, this step has a minimal effect on the work required for uncertainty propagation, only a substitution is required. The next step, however, requires an additional step of propagating uncertainty through the cos− 1 operation. To begin, the mean and variance of $\frac{Y}{X}$ following the definitions of μ_X, μ_Y, σ_X, σ_Y and σ_{XY} (provided within the subsection corresponding to the relevant search method) should be calculated using the method described at the beginning of this section. Then, an assumption will be made that $\frac{Y}{X}$ is a normally distributed random variable with the calculated expectations and variances:

$$Q \equiv \frac{Y}{X} \sim N\left(\mu_Q = \mathrm{E}\left[\frac{Y}{X}\right], \sigma^2_Q = \mathrm{Var}\left[\frac{Y}{X}\right]\right) \tag{B.56}$$

Following this, an approximation of $\cos^{-1}(Q)$ can be made (Infinite Series, 2022):

$$\cos^{-1}(Q) \approx \frac{\pi}{2} - \sum_{t=0}^{T} \frac{(2t)!}{(t!2^t)^2} \frac{Q^{2t+1}}{2t+1}. \tag{B.57}$$

By applying the expectation operator and identifying the plain moment of a normal distribution:

$$\mathrm{E}\left[\cos^{-1}(Q)\right] \approx \frac{\pi}{2} - \sum_{t=0}^{T} \frac{(2t)!}{(t!2^t)^2} \frac{\sigma_Q^{2t+1}}{2t+1} \left(-\boldsymbol{i}\sqrt{2}\right)^{2t+1} \times \mathrm{U}\left(-t - \frac{1}{2}, \frac{1}{2}, -\frac{1}{2}\left(\frac{\mu_Q}{\sigma_Q}\right)^2\right). \tag{B.58}$$

Here, $\boldsymbol{i}$ is used to represent the well-known imaginary part of a complex number ($\sqrt{\boldsymbol{i}} = -1$). U is the Tricomi's (confluent hypergeometric) function or alternatively, the confluent hypergeometric function of the second kind.

In order to find the $\mathrm{Var}\left[\cos^{-1}(Q)\right]$, it is first necessary to find $\mathrm{E}\left[\cos^{-1}(Q)^2\right]$. Using the representation for $\sin^{-1}(x)^2$ provided by (Borwein et al., 2004):

$$2\left(\sin^{-1}\left(\frac{x}{2}\right)\right)^2 = \sum_{n=1}^{\infty} \frac{x^{2n}}{n^2 \binom{2n}{n}}. \tag{B.59}$$

The following approximation can be obtained for $\cos^{-1}(Q)^2$:

$$\cos^{-1}(Q)^2 \approx \frac{1}{2} \sum_{t=1}^{T} \frac{4^t Q^{2t}}{t^2 \binom{2t}{t}} + \frac{\pi^2}{4} - \pi \sum_{t=0}^{T} \frac{(2t)!}{(t!2^t)^2} \frac{Q^{2t+1}}{2t+1}. \tag{B.60}$$

Then, apply the expectation operator:

$$\mathrm{E}\left[\cos^{-1}(Q)^2\right] \approx \frac{1}{2} \sum_{t=1}^{T} \frac{4^t}{t^2 \binom{2t}{t}} \mathrm{E}\left[Q^{2t}\right] + \frac{\pi^2}{4} - \pi \sum_{t=0}^{T} \frac{(2t)!}{(t!2^t)^2 (2t+1)} \mathrm{E}\left[Q^{2t+1}\right]. \tag{B.61}$$

Now replace with the appropriate expressions for the plain moment of a normal distribution:

$$\mathrm{E}\left[\cos^{-1}(Q)^2\right] \approx \frac{1}{2} \sum_{t=1}^{T} \frac{4^t \sigma_Q^{2t}}{t^2 \binom{2t}{t}} (-\boldsymbol{i}\sqrt{2})^{2t} \mathrm{U}\left(-t, \frac{1}{2}, -\frac{1}{2}\left(\frac{\mu_Q}{\sigma_Q}\right)^2\right) \tag{B.62}$$

$$+ \frac{\pi^2}{4} - \pi \sum_{t=0}^{T} \frac{(2t)! \sigma_Q^{2t+1}}{(t!2^t)^2 (2t+1)} (-\boldsymbol{i}\sqrt{2})^{2t+1} \mathrm{U}\left(-t - \frac{1}{2}, \frac{1}{2}, -\frac{1}{2}\left(\frac{\mu_Q}{\sigma_Q}\right)^2\right). \tag{B.63}$$

With these expressions,

$$\mathrm{Var}\left[\cos^{-1}(Q)\right] = \mathrm{E}\left[\cos^{-1}(Q)^2\right] - \mathrm{E}\left[\cos^{-1}(Q)\right]^2. \tag{B.64}$$

Appendix C. Sampling parameter translation to uncertainty

In this section, a derivation will be presented on how to approximately select the number of generations to run in a Monte Carlo code to achieve some target uncertainty. The number of generations run on search step n to achieve some uncertainty in the Monte Carlo calculated result, $\sigma_{f,n}$ will be referred to as g_n. An approximation for the uncertainty in the Monte Carlo calculated result, $\sigma_{f,n}$ is typically provided by the Monte Carlo code. To begin, start with a general expression for the uncertainty in the result of a Monte Carlo calculated quantity (σ) as a function of the number of trials run (g):

$$\sigma = \frac{k}{g^r}. \tag{C.1}$$

With k and r as undetermined coefficients, the above expression is general, but typically, $r \approx 0.5$ for Monte Carlo calculation methods. Nevertheless, k and r will be fitted using past evaluations of $f(\theta_n)$ based on the g_n used in those evaluations and the resulting $\sigma_{f,n}$. To do this, first we apply the natural log to both sides of the equation:

$$\ln(\sigma) = \ln\left(\frac{k}{g^r}\right). \tag{C.2}$$

Then, with some algebra, a linear relationship can be identified between $\ln(\sigma)$ and $\ln(g)$:

$$\ln(\sigma) = \ln(k) - r\ln(g). \tag{C.3}$$

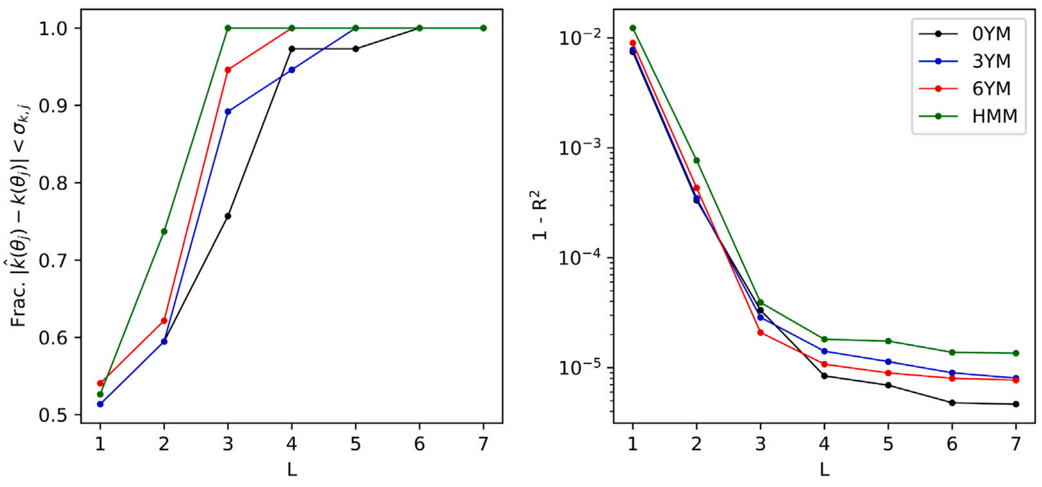

Fig. D.12. Fraction of $\hat{k}(\theta_j)$ which fall within the 95% confidence interval of $k(\theta)$ (left) and $1 - R^2$ (right) versus L for all four surrogate models.

With this linear form, regression can be used on all past evaluations of $f(\theta)$ to find best-fit values for k and r:

$$\ln(k) = \frac{\sum_{i=0}^{n} \ln(\sigma_{f,i}) \sum_{i=0}^{n} \ln(g_i)^2 - \sum_{i=0}^{n} \ln(\sigma_{f,i}) \ln(g_i) \sum_{i=0}^{n} \ln(g_i)}{(n+1)\sum_{i=0}^{n} \ln(g_i)^2 - \left(\sum_{i=0}^{n} \ln(g_i)\right)^2} \tag{C.4}$$

$$r = \frac{\sum_{i=0}^{n} \ln(\sigma_{f,i}) \sum_{i=0}^{n} \ln(g_i) - (n+1)\sum_{i=0}^{n} \ln(\sigma_{f,i}) \ln(g_i)}{(n+1)\sum_{i=0}^{n} \ln(g_i)^2 - \left(\sum_{i=0}^{n} \ln(g_i)\right)^2}. \tag{C.5}$$

Then, using Eq. (C.2) to determine g_{n+1} provided σ_{n+1}:

$$g_{n+1} = \exp\left(\left(\frac{\sum_{i=0}^{n} \ln(\sigma_{f,i}) \sum_{i=0}^{n} \ln(g_i)^2 - \sum_{i=0}^{n} \ln(\sigma_{f,i}) \ln(g_i) \sum_{i=0}^{n} \ln(g_i)}{(n+1)\sum_{i=0}^{n} \ln(g_i)^2 - \left(\sum_{i=0}^{n} \ln(g_i)\right)^2}\right.\right.$$

$$\left.\left. - \ln\left(\sigma_{n+1}\right)\right)\right) \tag{C.6}$$

$$\left.\times \frac{(n+1)\sum_{i=0}^{n} \ln(g_i)^2 - \left(\sum_{i=0}^{n} \ln(g_i)\right)^2}{\sum_{i=0}^{n} \ln(\sigma_{f,i}) \sum_{i=0}^{n} \ln(g_i) - (n+1)\sum_{i=0}^{n} \ln(\sigma_{f,i}) \ln(g_i)}\right). \tag{C.7}$$

Appendix D. Surrogate model fitting

In this section, a detailed description on how the surrogate models are fitted to data points calculated by Serpent 2 is described. The jth data point for each model is composed of a control drum position, denoted as θ_j, a core criticality at that control drum position, denoted as $k(\theta_j)$, and an uncertainty represented by a standard deviation $\sigma_{k,j}$. In total, the data sets used to fit each model are composed of J data points ($J = 37$ for the models based on the eVinci$^{\text{TM}}$ design and $J = 19$ for the model based on data from the Holos-quad design). For regression, a modified Fourier basis is used:

$$\varphi_\ell(\theta) = \begin{cases} 1 & \ell = 0 \\ \cos\left(\lfloor \frac{\ell+1}{2} \rfloor \theta\right) & n > 0 \wedge n \text{ odd} \\ \sin\left(\lfloor \frac{\ell+1}{2} \rfloor \theta\right) & n > 0 \wedge n \text{ even} \end{cases}. \tag{D.1}$$

It is worth noting that this basis is not orthonormal over the domain $[0°, 180°]$ but this has minimal repercussions in its use here.

The objective of this fitting procedure is to obtain some approximation for $k(\theta)$, denoted as $\hat{k}(\theta)$, by selecting $a_1, a_2, \ldots, a_L$ such that:

$$|\hat{k}(\theta_j) - k(\theta_j)| < 1.96\sigma_{k,j} \quad \forall j \in \{1, \ldots, J\} \tag{D.2}$$

where

$$\hat{k}(\theta) = \sum_{\ell=1}^{L} a_\ell \varphi_\ell(\theta). \tag{D.3}$$

In words, the evaluation of the surrogate model on all control drum positions in the dataset should be within the 95% confidence interval of the Serpent-reported k_{eff}. Although a variety of methods exist for selecting $a_1, a_2, \ldots, a_L$, here regression with the aforementioned Fourier basis is used to minimize the least-squares residual. L will be increased from $L = 1$ until the criteria described in Eq. (D.2) is satisfied. Satisfaction of this criteria is guaranteed for some $L \leq J$ because $\hat{k}(\theta_j) = k(\theta_j)$ is true for all j if $L = J$.

Nevertheless, the regression procedure is relatively easy to perform using the normal equations, given some L. To begin, define:

$$X = \begin{bmatrix} \varphi_0(\theta_1) & \varphi_1(\theta_1) & \ldots & \varphi_L(\theta_1) \\ \varphi_0(\theta_2) & \varphi_1(\theta_2) & \ldots & \varphi_L(\theta_2) \\ \vdots & \vdots & & \vdots \\ \varphi_0(\theta_J) & \varphi_1(\theta_J) & \ldots & \varphi_L(\theta_J) \end{bmatrix}, \tag{D.4}$$

$$\boldsymbol{h} = \begin{bmatrix} k(\theta_1) & k(\theta_2) & \ldots & k(\theta_J) \end{bmatrix}^T \tag{D.5}$$

and

$$\boldsymbol{\alpha} = \begin{bmatrix} a_1 & a_2 & \ldots & a_L \end{bmatrix}^T. \tag{D.6}$$

Then the solution for $\boldsymbol{\alpha}$ which minimizes the square residual is given by:

$$\boldsymbol{\alpha} = \left(X^T X\right)^{-1} X^T \boldsymbol{h}. \tag{D.7}$$

Fig. D.12 is provided to show how the accuracy in the surrogate model evolves with higher L. The fraction of $\hat{k}(\theta_j)$ which fall within the 95% confidence interval of $k(\theta)$ is shown for L up to 8. From these results, $L = 3$ for the HMM, $L = 4$ for the 6YM, $L = 5$ for the 3YM and $L = 6$ for the 0YM. Information on the coefficient of determination (R^2) is also shown in this figure to demonstrate that, even for low L, the surrogate models fit the data closely.

References

Arfken, G.B., Weber, H.J., 1999. Mathematical methods for physicists.

Borwein, J.M., Bailey, D.H., Girgensohn, R., 2004. Experimentation in Mathematics: Computational Paths To Discovery. AK Peters/CRC Press.

Bowman, S.M., 2011. SCALE 6: comprehensive nuclear safety analysis code system. Nucl. Technol. 174 (2), 126–148.

Brown, F.B., 2009. A review of best practices for Monte Carlo criticality calculations.

Dall'Osso, A., 2008. A neutron balance approach in critical parameter determination. Ann. Nucl. Energy 35 (9), 1686–1694.

Downar, T., Lee, D., Xu, Y., Kozlowski, T., Staudenmier, J., 2004. PARCS v2. 6 US NRC core neutronics simulator theory manual. School of Nuclear Engineering Purdue University.

Gill, D.F., Nease, B., Griesheimer, D., 2013. Movable geometry and eigenvalue search capability in the MC21 Monte Carlo code. Tech. Rep., American Nuclear Society, 555 North Kensington Avenue, La Grange Park, IL.

Griesheimer, D., Gill, D., Nease, B., Sutton, T., Stedry, M., Dobreff, P., Carpenter, D., Trumbull, T., Caro, E., Joo, H., et al., 2014. MC21 v. 6.0–a continuous-energy Monte Carlo particle transport code with integrated reactor feedback capabilities. In: SNA+ MC 2013-Joint International Conference on Supercomputing in Nuclear Applications+ Monte Carlo. EDP Sciences, p. 06008.

Guo, H., Feng, K., Gu, H., Yao, X., Bo, L., 2021. Neutronic modeling of megawatt-class heat pipe reactors. Ann. Nucl. Energy 154, 108140.

Hernandez, R., Todosow, M., Brown, N.R., 2019. Micro heat pipe nuclear reactor concepts: Analysis of fuel cycle performance and environmental impacts. Ann. Nucl. Energy 126, 419–426.

Infinite Series, 2022. Inverse trigonometric functions — Wikipedia, the free encyclopedia.

Jo, Y., Cho, N.Z., 2018. Inline critical boron concentration search with p-CMFD feedback in whole-core continuous-energy Monte Carlo simulation. Ann. Nucl. Energy 120, 402–409.

Leppänen, J., Pusa, M., Viitanen, T., Valtavirta, V., Kaltiaisenaho, T., 2015. The serpent Monte Carlo code: Status, development and applications in 2013. Ann. Nucl. Energy 82, 142–150.

Li, Z., Wang, K., Deng, J., 2015. Perturbation based Monte Carlo criticality search in density, enrichment and concentration. Ann. Nucl. Energy 76, 350–356.

Mertyurek, U., Ilas, G., 2022. Nuclide inventory benchmark for BWR spent nuclear fuel: Challenges in evaluation of modeling data assumptions and uncertainties. J. Nucl. Eng. 3 (1), 18–36.

Price, D., Kinast, S., Barr, K., Kochunas, B., Filippone, C., 2022a. A perturbation-based hybrid methodology for control drum worth prediction applied to the HOLOS-quad microreactor concept. Ann. Nucl. Energy 168, 108903.

Price, D., Radaideh, M.I., Kochunas, B., 2022b. Multiobjective optimization of nuclear microreactor reactivity control system operation with swarm and evolutionary algorithms. Nucl. Eng. Des. 393, 111776.

Price, D., Radaideh, M.I., Mui, T., Katare, M., Kozlowski, T., 2020. Multiphysics modeling and validation of spent fuel isotopics using coupled neutronics/thermal-hydraulics simulations. Sci. Technol. Nucl. Install. 2020.

Price, D., Radaideh, M.I., O'Grady, D., Kozlowski, T., 2019. Advanced BWR criticality safety part II: Cask criticality, burnup credit, sensitivity, and uncertainty analyses. Prog. Nucl. Energy 115, 126–139.

Radaideh, M.I., Price, D., O'Grady, D., Kozlowski, T., 2019. Advanced BWR criticality safety part I: Model development, model benchmarking, and depletion with uncertainty analysis. Prog. Nucl. Energy 113, 230–246.

Rearden, B.T., Jessee, M.A., 2018. SCALE code system. Tech. Rep., Oak Ridge National Lab.(ORNL), Oak Ridge, TN (United States).

Roskoff, N., Kucukboyaci, V., Levinsky, A., Laboure, V., Harter, J., Zabriskie, A., Charlot, L., 2022a. Modeling and analysis of a micro-reactor using the DireWolf code suite. In: International Conference on Physics of Reactors 2022. American Nuclear Society.

Roskoff, N., Kucukboyaci, V., Levinsky, A., Laboure, V., Harter, J., Zabriskie, A., Charlot, L., 2022b. Transient analysis of a micro-reactor using the DireWolf code suite. In: International Conference on Physics of Reactors 2022. American Nuclear Society.

Stauff, N., Lee, C., Filippone, C., 2022. Core Design of the Holos-Quad Microreactor. Tech. Rep., Argonne National Lab.(ANL), Argonne, IL (United States).

Topham, T.J., Rau, A., Walters, W.J., 2020. An iterative fission matrix scheme for calculating steady-state power and critical control rod position in a TRIGA reactor. Ann. Nucl. Energy 135, 106984.