Multi-objective Optimization of Nuclear Microreactor Control System Operation with Swarm and Evolutionary Algorithms

[a]Dean Price*, [b]Majdi I. Radaideh, [a]Brendan Kochunas

[a]*Department of Nuclear Engineering and Radiological Science, University of Michigan, 2355 Bonisteel Blvd., Ann Arbor, MI 48109, United States*

[b]*Department of Nuclear Science and Engineering, Massachusetts Institute of Technology (MIT), Cambridge, MA 02139, United States*

Abstract

To improve the marketability of novel microreactor designs, there is a need for automated and optimal control of these reactors. This paper presents a methodology for performing multi-objective optimization of control drum operation for a microreactor under normal and accident conditions. Two different case studies are used where the control drum configuration is optimized for the reactor to be critical with some desired power distribution that would satisfy peaking limits. In addition to these objectives, one case study seeks to minimize control drum travel distance where the other maximizes control drum differential worth. A surrogate model for power distribution is developed based on a feedforward neural network. The process for determining weights for scalarization of the multi-objective optimization problem is also detailed. Six optimization algorithms: evolutionary strategies, differential evolution, grey wolf optimization, Harris hawks optimization, moth flame optimization and particle swarm optimization, are all applied to these cases and the results analyzed. All algorithms demonstrated optima-seeking behavior and could present reasonable optima in minutes. The moth flame optimization algorithm was found to perform particularly well on both cases. Overall, it was found that the algorithms capable of supplying the best optima were also the most consistent. Finally, the found optima were verified with the original model used to train surrogates.

Keywords: Evolutionary Optimization; Swarm Optimization; Nuclear Reactor Control; Multi-objective Optimization; Surrogate Modeling; Microreactor

1. Introduction

To reach net-zero emissions, nuclear and renewable energy sources need to increase production in order to offset greenhouse gas emitting technologies. However, the energy market still favors coal, oil, and natural gas power plants in most cases due to their comparatively cheaper construction and operational costs; resulting in more nuclear power plant closures. The current situation puts more pressure on the nuclear industry to innovate by developing small modular (SMRs) and microreactors that are cheaper to build, safer to operate,

*Corresponding Author: Dean Price (deanrp@umich.edu)

and faster to license and construct [1]. A major design goal of these new reactors is automated and optimal operation of their systems to ensure economic and safe performance [2].

Optimization, at various levels of complexity and cost, exist in every part of the nuclear engineering field including reactor core design [3, 4], reactor safety [5] and operation [6, 7, 8]. In the field of reactor design, in [9], the authors conducted a study where parameters for a gas-cooled fast breeder reactor core were optimized for various objectives such as k_{eff}, breeding ratio and maximum burnup. At the assembly level in both 2D [10] and 3D [11] geometries, a physics-informed framework was developed where reinforcement learning algorithms are used to optimize boiling water reactor bundles. In these studies, reinforcement learning was either used as a direct optimizer [10] or as a tool to guide the search of evolutionary algorithms to focus on feasible regions that satisfy problem constraints [11]. In terms of reactor safety, one study [12] applied a fast optimization algorithm to optimize the shielding structure of light water reactors. It was found that well-designed optimization algorithms lead to more accuracy in the final result.

In some sense, optimization in the field of reactor control is seen as an essential aspect of the economic viability of SMRs and microreactors. Efficient operation is required to offset the loss of economy-at-scale savings that smaller-capacity reactors experience [13]. The study by [14] details the state of economic analyses for SMRs which includes a proposition for shared control systems of multi-module reactors to reduce operations and maintenance costs but would require sophisticated control systems. Optimization in the field of reactor control can take many forms. In model predictive control, optimization must be performed in real time to determine an optimal control system response to achieve some performance objective. In [15], the optimization problem is solved using quadratic programming, this is fairly common in model predictive control [16]. Quadratic programming is efficient but only works for a quadratic objective functions. More sophisticated optimization strategies can be used for other control strategies such as fuzzy-PID [17, 18]. In [19], a fuzzy-PID control strategy for the H.B. Robinson nuclear power plant was designed and tested using point reactor models and good performance was observed. PID-based control strategies typically perform very well but are limited to single-input-single-output systems. For control system design, the study [20] used combinatorial optimization on control system design for core k_{eff}, rod worth and anti-shadowing effects while ensuring reactor safety by enforcing minimum rod group worth. With more algorithm-focused studies, the work of [21] demonstrated the use of different particle swarm optimization (PSO) variants on both a reactor core design problem and a fuel loading problem for the Angra-1 nuclear reactor. Another study by [22] demonstrated a hybrid PSO strategy which combines a modified Nelder–Mead simplex algorithm with PSO on pressurizer design and pressurized water reactor (PWR) secondary loop operation.

Evolutionary and swarm algorithms [23] are well-known for optimization due to their advantages of inherent randomness to enhance exploration, scalability in high dimensions, orientation for parallel search, and gradient-free characteristics. Although there are many differences between evolutionary and swarm algorithms, we will focus on a major difference, which is the use of the crossover operators. Evolutionary algorithms such as genetic algorithms (GA) [24], evolutionary strategies [25], and differential evolution [26] exhibit special

crossover operators that mix population individuals to generate new offspring. These crossover operators mimic the evolution and genetic exchange in human or animals. The swarm algorithms do not support this feature, instead, the weak individuals try to improve themselves by either observing and following the other fittest individuals in the swarm or by employing nature-inspired mutation operators. For example, particle swarm optimization [27] is a famous swarm algorithm, where each particle is influenced by its local best position and by the global best position in the swarm. In addition, the grey wolf optimizer [28] employs nature-inspired steps of hunting, searching for prey, and attacking prey to improve the wolf positions, while the leadership hierarchy is controlled by the top four wolves in the group. Other swarm algorithms relevant to this study are Harris hawks optimization [29], and moth flame optimization [30].

The contribution of this work is three-fold. First, we demonstrate the value of multi-objective optimization and machine learning in optimizing the operational performance of advanced nuclear reactor reactivity control systems. Since advanced reactors including microreactors are new, significant optimization and analysis of their design and applications are required to ensure safe and economic performance. Second, we use empiricism to determine the weights of the multi-objective function and supervised learning to build surrogate models to accelerate and increase the efficiency of the optimizer. Third, we deploy and benchmark a variety of evolutionary and swarm algorithms by investigating their tendencies for exploration and exploitation to find an optimal reactor control drum configuration. The methodology is demonstrated using two case studies. First, under conditions where one control drum is inoperable, the configuration of the remaining control drums in a microreactor are optimized to achieve core criticality with a favorable power distribution. Next, with all drums fully functional, the control drum positions are optimized in the same microreactor for nominal operation.

The remaining sections of this paper are organized as follows: Section 2 presents the mathematical theory of nuclear reactor control and modeling and simulation tools used in this work. In Section 3, the methodology of this work is described, that includes machine learning methods, formation of the multi-objective function, and the evolutionary/swarm optimizers. The results of this paper are presented and discussed in Section 4, followed by the work summary in Section 5.

2. Reactor Models

The case studies featured in this work are focused around the HOLOS-Quad reactor design [31]. This reactor is designed with modular construction where separate units can be transported independently and assembled at the deployment site. Some parameters of the core design were selected in a design optimization process where a genetic algorithm was used to minimize reactor weight and maximize core lifetime [32, 33]. The version of the HOLOS core design used in this paper is detailed in [34]. To summarize, the HOLOS-Quad core is a 22 MWt high-temperature gas-cooled microreactor (HTGR) which is controlled by 8 cylindrical control drums. The core uses TRISO fuel particles contained in hexagonal graphite blocks for moderation. The graphite blocks have channels which allow for helium gas to provide cooling. A labeled cross sectional

view of the core is given in Figure 1 to show the geometrical layout of the HOLOS-Quad core.

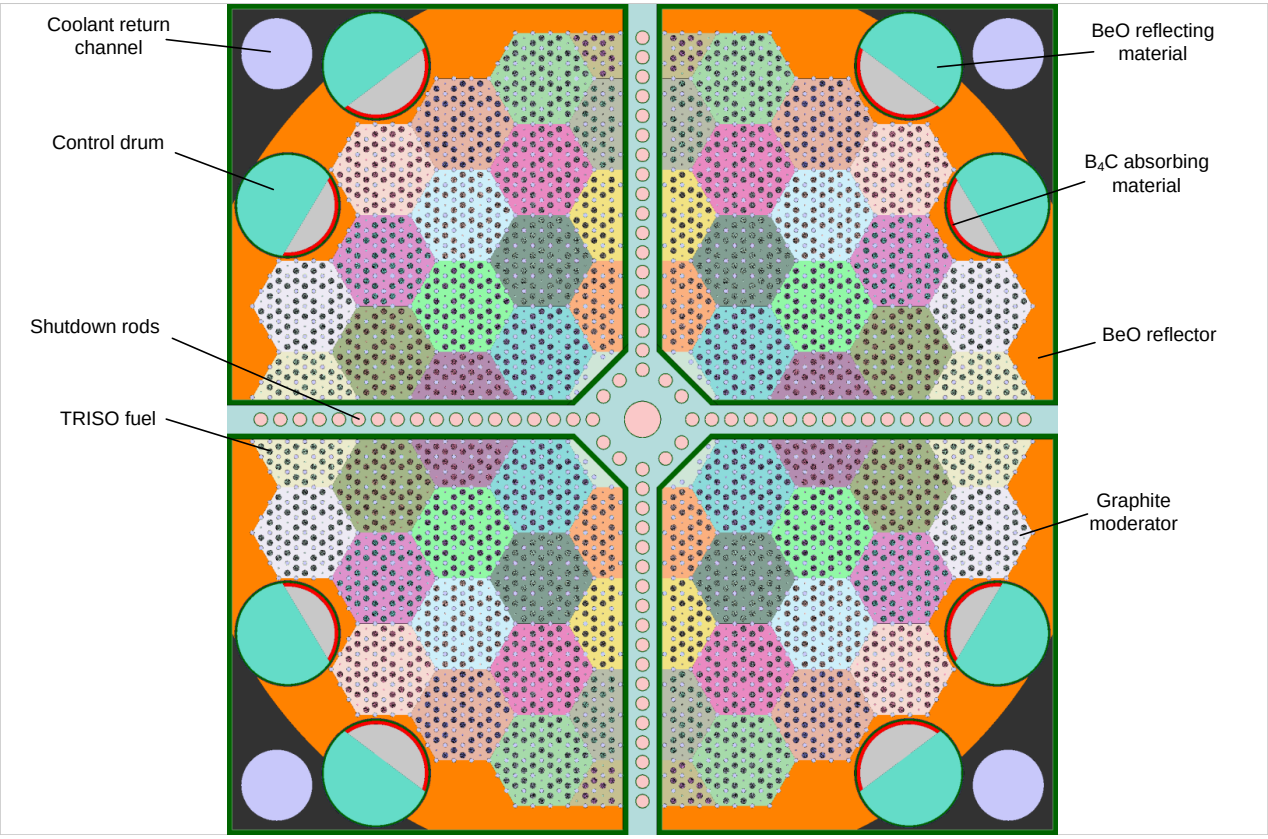

Figure 1: Geometrical layout of major components of HOLOS-Quad reactor.

Of particular importance in this study is the function of the control drums. These drums control core reactivity by rotating to vary the proximity of the B_4C on a portion of their outer edges to the fueled region of the core. These control mechanisms have been proposed in many microreactor designs due to their compactness [35, 36, 37] and are also in use at the Advanced Test Reactor facility at Idaho National Laboratory [38, 39]. One challenge associated with control drums is that the core power shape will shift in response to changes in the orientation of the control drums. This causes the prediction of control drum reactivity worths for arbitrary configurations to be nontrivial. The model used in this study to predict control drum worths for arbitrary configurations is described in detail in [40]. These changes in core power shape can also lead to undesired power peaking. Therefore, another model is developed in this study to predict core quadrant powers. The model to predict core reactivity and the model to predict core quadrant powers are described in the upcoming Section 2.1 and Section 2.2.

2.1. Control Drum Reactivity Worth Model

There are 8 control drums which can rotate a full 360° to control the reactor. In core operation, the reactivity required by the control drums will be a function of the dynamic characteristics of the reactor such as fuel temperature, moderator temperature, xenon concentration, and burnup. Characterizing these feedback

mechanisms is not the goal of the current study. Instead, given a target reactivity, this study seeks to optimize the positions of the 8 control drums independently to fit a particular power shape. We also consider a second case that seeks to maximize differential worth of the control drum configuration. The motivation for this is that a larger differential worth gives the control system more flexibility given the constraints of the maximum control drum rotation speeds. These control drum worths can be predicted using a Monte Carlo transport code (such as Serpent [41]) by creating a model of the core and rotating the control drums to the desired orientations. Unfortunately, the computational time required for this approach makes it infeasible when thousands of drum worth estimations are needed for a single optimization problem. As such, the model described in [40] is used in this work for calculating the control drum reactivity worth.

This hybrid control drum worth model (HDWM) provides control drum worth estimates in milliseconds that have been evaluated to have a 50 pcm mean absolute error. The perturbation-based model is comprised of two components, a physics-based component that captures the reactivity effect from the flux incident on the control drum and a statistical component to capture higher-order effects that arise from core-power shape shifts brought about from control drum rotations. One of the drawbacks of this model is the large up-front cost of training the statistical component of the model. In [40], it was shown that the most accurate form of the model took 70 Serpent model evaluations to train. Although this is significantly less than the thousands of Serpent calculations required to run the optimization algorithms shown in this study if the HDWM was not used, it is still an important consideration. One additional benefit of the model form of HDWM is that differential drum worths can be estimated with similar computational cost by simply differentiating the model with respect to a particular drum rotation angle. The derivation for this is shown in Appendix A.

2.2. Quadrant Power Model

Another important reactor parameter that should be monitored for safe and economic operation is the quadrant power tilt ratio (QPTR). This parameter is reflective of the differences in power distribution in each quadrant of the core. QPTR can be calculated by dividing the maximum quadrant flux by the average quadrant flux as follows

$$\text{QPTR} = \frac{4 \times \max\left(P_1, P_2, P_3, P_4\right)}{P_1 + P_2 + P_3 + P_4}, \tag{1}$$

According to the Westinghouse Technology Systems Manual [42], a QPTR exceeding 1.02 requires that the reactor power be reduced to reach required safety margins. During reactor operation, this quantity is monitored using two sets of excore detectors positioned in sets of four in the upper and lower regions of the core. Although typically used in PWRs, it is thought that this measure should be appropriate for the HOLOS-Quad core due to its cylindrical symmetry. where P_i indicates the flux in the i-th core quadrant. As with the control drum reactivity worth, it is infeasible to use a full core transport model to calculate this quantity for the large number of control drum angle configurations that are needed to run an optimization algorithm. Therefore, in this study we use a deep feedforward neural network that is trained on a dataset with about 3000 samples after

using reactor symmetries to multiply samples. Each sample in this dataset has 8 control drum angles where each angle was selected from a uniform random distribution spanning -180° to 180° and the corresponding fluxes in each of the four quadrants is calculated using the Serpent code. The considerably larger dataset is needed because of the larger number of degrees of freedom associated with general statistical models such as deep neural networks (DNN). Using this data, a DNN was trained using the classical Keras/Tensorflow framework in Python [43]. The neural network can be expressed as

$$P_1, P_2, P_3, P_4 = DNN(\vec{x}, \beta) \tag{2}$$

where $\beta = (W, b)$ are the network weight and bias parameters to be determined, respectively, while $\vec{x}$ is a vector of the eight control drum angles, i.e. $\vec{x} = [\theta_1, \theta_2, ..., \theta_8]$. The DNN consists of an input layer, multiple hidden layers, and output layer. The output of each hidden layer can be calculated by applying the activation function over the multiplication of the weights and the layer inputs, and then adding the bias term as follows

$$\boldsymbol{h}_i = g_i(\boldsymbol{W}_i \boldsymbol{h}_{i-1} + \boldsymbol{b}_i), \tag{3}$$

where $g_i(.)$ is the activation function for hidden layer i, which is a mapping function to model the relationship between the input and the output. Rectified Linear Unit (ReLU) is a typical activation function for DNN. $\boldsymbol{W}$ is the weight matrix, $\boldsymbol{h_{i-1}}$ is the output of the previous hidden layer, and $\boldsymbol{b}$ is the bias or intercept term. After completing the forward propagation step in all layers, the predicted output ($\hat{y}$) can be calculated, which allows determining the loss function such as the mean absolute error (MAE)

$$MAE(\boldsymbol{W}, \boldsymbol{b}) = \frac{1}{n} \sum_{i=1}^{n} |\hat{y}(\boldsymbol{W}, \boldsymbol{b})_i - y_i|, \tag{4}$$

where y is the target output and n is the number of training samples generated by Serpent. For each training step, the errors in Eq.(4) are propagated backwards through their gradient values, where the weights of each layer are updated according to the loss gradient such that in the next batch of samples, the loss function is more likely to be minimized. The backpropagation step can be trained by gradient descent or using more advanced techniques such as the Adam optimizer (adaptive moment estimation)[44]. Adam optimizer has a hyperparameter called learning rate, which controls the step size the optimizer takes while searching for the minimum of the loss function. Every full forward and backward pass through the dataset is called an epoch, and the DNN network needs to be trained for many epochs to achieve satisfactory accuracy. The dataset is passed to the DNN in several batches with a mini-batch size (e.g. 8, 16, 32) to allow multiple model updates, and also accommodate the computer memory. The dataset is split into training and test sets to avoid model overfitting to the training data. Also, a small validation set is held from the training set to allow on-the-fly validation of the model during training. A typical size of the validation set is 10%-20% of the training set, depending on the number of available samples.

We utilize two hyperparameter search methods including grid and random searches to find the optimal number of layers, nodes per layer, learning rate, and batch size. In addition, we benchmark the performance of the DNN model against other machine learning methods through developing other quadrant power models using random forests, decision trees, and linear regression.

3. Methodology

The methodology adopted in this work is illustrated by the flowchart in Figure 2. The reader can notice that Section 2 already covered the HDWM and quadrant power models. The remaining parts of the flowchart are described in this section. Also it is worth mentioning that the flowchart is executed twice for the two case studies considered in this work.

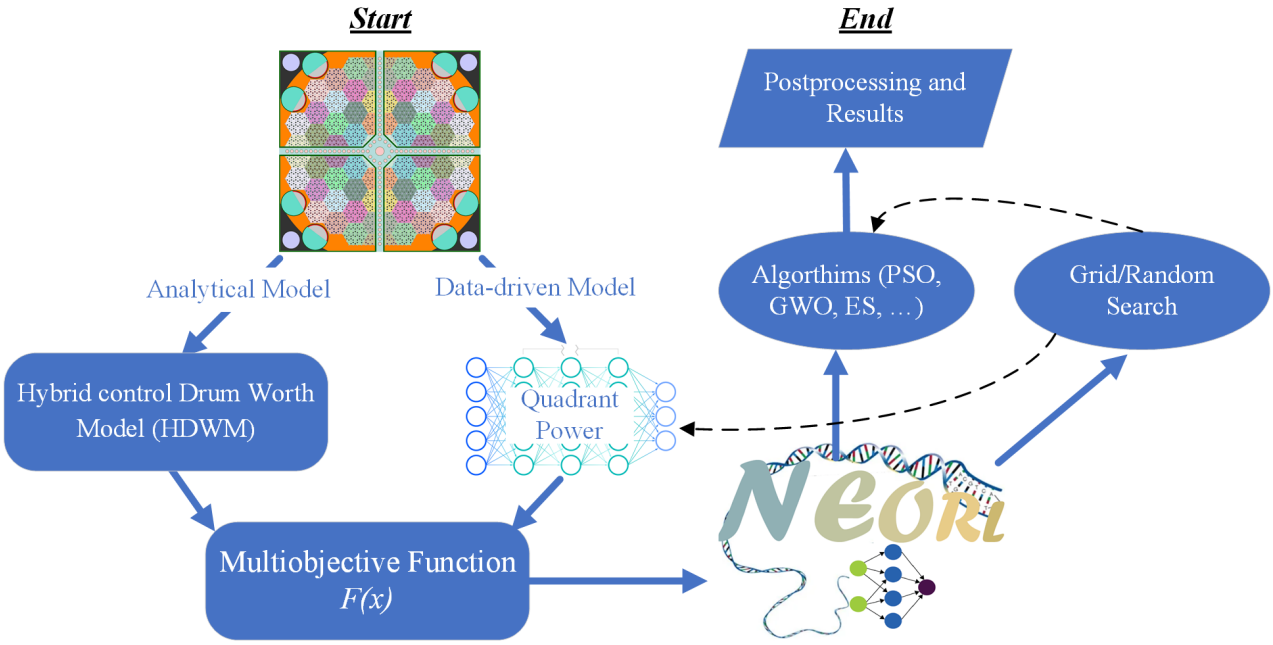

Figure 2: Flowchart of the methodology used in this work

3.1. Case Studies

Two separate optimization problems relevant to the control of the HOLOS-Quad microreactor were devised. In both cases, the reactivity required for the core to reach criticality is determined with a cold, zero-power configuration (room-temperature core and zero flux). The first case study, referred to as case study A (CSA), involves evaluating the possibility for reactor operation given that one of the drums becomes immobilized in the inward-facing direction. In this orientation, the immobilized drum will have inserted its maximum negative reactivity worth to the core. From a safety standpoint, core operation is more feasible than if the drum was immobilized in the outward facing direction. However, it is still important that the reactor power shape remains relatively uniform across the core. As such, the QPTR should be minimized. With this scenario, two

parameters can be identified that will be controlled using the orientations of the remaining control drums: reactivity and QPTR. The reactor should be able to reach criticality with a minimal QPTR. Furthermore, one additional objective is introduced into CSA where the maximum travel distance of the 8 drums from fully inserted should be minimized. In the practical sense, this behavior is desired to minimize the wear on the control drum drive mechanisms.

In the second case study, referred to as case study B (CSB), a control drum configuration will be found which brings the reactor to critical while maximizing the total differential worth of all drums. The result from this analysis may reflect an optimal control drum configuration for nominal reactor operation. Of course, this optimization is performed for a cold, zero-power core but the same logic applies for the full power case. The motivation for maximizing the total differential worth of all drums is that it provides a controller the greatest range of possible reactivity injections given a small amount of time due to the upper limit on drum rotation speed. One additional objective will be introduced to CSB that the QPTR should be as close as possible to 1.02 with the highest power quadrant set to the upper right quadrant. This objective is introduced to eliminate symmetry from the problem and provide a more complex search space for the optimization algorithms. Mathematical formulations for the optimizations of both case studies are given in Section 3.2.

3.2. Objective Function Formulation

Without loss of generality, multi-objective optimization can be formulated as follows:

$$\begin{aligned} &\min_{\vec{x}} F(\vec{x}) = (f_1(\vec{x}), f_2(\vec{x}), ..., f_k(\vec{x})) \\ &\text{subject to,} \\ &g_i(\vec{x}) \geq 0, \quad i = 1, 2, ..., m, \\ &h_j(\vec{x}) = 0, \quad j = 1, 2, ..., p, \\ &k \geq 2, \end{aligned} \tag{5}$$

where k is the number of single objective functions, $g_i(\vec{x})$ is an inequality constraint, m is the number of inequality constraints, $h_i(\vec{x})$ is an equality constraint, and p is the number of equality constraints. The set of solutions $\vec{x}$ that satisfies all constraints defines the feasible region Ω, i.e. $\vec{x} \in \Omega$. In this application, $\vec{x}$ is a vector in $\mathbb{R}^8$ where each element corresponds to the rotation angle of a control drum. In addition, there are three objectives to be optimized, i.e. $k = 3$, while there are no explicit equality (h) or inequality (g) constraints in this problem. Nevertheless, an implicit constraint is imposed on the input space, where each drum angle in $\vec{x}$ is allowed to vary between -180° to 180°.

Multi-objective optimization proves to be a challenge due to the possibility for competing objectives. In any nontrivial multi-objective optimization problem, the $\vec{x}$ which optimizes one objective will not be the same $\vec{x}$ which optimizes another. The set of solutions for which any improvement in one objective will result in the deterioration of another objective is called a Pareto front. In the case studies presented here, it is necessary to select a single $\vec{x}$ which can be considered the solution to the optimization problem. Therefore, some additional

criteria must be applied to the optimization problem to select a single solution from the set of solutions contained in the Pareto front. Here, a technique called scalarization will be used [45]. Scalarization uses a single composite objective to represent the collective interests of each individual objective. Typically—and as is done in this study—the set of objectives represented by $(f_1(\vec{x}), f_2(\vec{x}), ..., f_k(\vec{x}))$ can be combined into a single objective using a weighted summation:

$$F(\vec{x}) = \sum_{i=1}^{k} \pm w_i f_i(\vec{x}) \tag{6}$$

The process of selecting the set of weights $(w_1, w_2, ..., w_k)$ is problem-specific. They are given for the case studies contained in this work in Section 4. For objectives to be maximized, a negative sign is placed in front of the weight. Furthermore, it is often ideal to scale the results of $(f_1(\vec{x}), f_2(\vec{x}), ..., f_k(\vec{x}))$ to ensure that the outputs are all roughly comparable in scale. In this study, if $\hat{f}_i(\vec{x})$ corresponds to the unscaled objective, $\hat{f}_{i,max}$ corresponds to the maximum value of the objective and $\hat{f}_{i,min}$ corresponds to the minimum possible value of the objective, then the scaled objective can be given as:

$$f_i(\vec{x}) = \frac{\hat{f}_i(\vec{x}) - \hat{f}_{i,min}}{\hat{f}_{i,max} - \hat{f}_{i,min}} \tag{7}$$

Next we describe how each of the objectives used in the case studies is formulated. The maximum and minimum values of the unscaled objectives were estimated when a bound was not obvious. In the end, this has little effect on the results as any shortcoming in these estimations will be accounted for when selecting the weights for scalarization. First, the criticality objective for both CSA and CSB is identical. The unscaled criticality objective can be written mathematically as:

$$\hat{f}_c(\vec{x}) = |\rho_{start} - \rho(\vec{x})| \tag{8}$$

$\rho(\vec{x})$ is calculated with the the hybrid worth model and ρ_{start} is the required reactivity for initial cold critical conditions. For scaling, $\hat{f}_{c,max}$ = 3308 pcm and $\hat{f}_{c,min}$ = 0 pcm. This objective should be minimimzed for both CSA and CSB.

Next, the QPTR objective for both cases is different. For CSA, a QPTR of 1 is desired. Therefore, the quadrant powers (P_1, P_2, P_3 and P_4) should all be equal. Mathematically, this objective can be written as :

$$\hat{f}_p(\vec{x}) = \sum_{i=1}^{4} \left| \frac{P_i(\vec{x})}{\sum_{j=1}^{4} P_j(\vec{x})} - \frac{1}{4} \right| \tag{9}$$

The quadrant powers are estimated with the neural network described in Section 2.2. In this case, $\hat{f}_{p,max}$ = 0.0345 and $\hat{f}_{p,min}$ = 0. This objective is minimized for CSA.

For CSB, a QPTR of 1.02 is desired where quadrant 1 has 25.5 % of the total reactor power while the

remaining three quadrants evenly split 74.5% power. Mathematically, this can be expressed as:

$$\hat{f}_p(\vec{x}) = \left| \frac{P_1(\vec{x})}{\sum_{j=1}^{4} P_j(\vec{x})} - 0.255 \right| + \sum_{i=2}^{4} \left| \frac{P_i(\vec{x})}{\sum_{j=1}^{4} P_j(\vec{x})} - \frac{0.745}{3} \right| \tag{10}$$

Here, $\hat{f}_{p,max} = 0.0345$ and $\hat{f}_{p,min} = 0$. For CSB, as with the previous objectives, this objective is minimized.

The final objective for CSA will be the travel distance of the farthest traveling drum. Here, it is desired that the travel distance is minimized. Mathematically, this can be formulated as:

$$\hat{f}_d(\vec{x}) = \|\vec{x}\|_\infty \tag{11}$$

The simple form of this equation is due to the fact that all drums are set to start at the inward-facing (or $\vec{x} = \vec{0}$) direction. In CSA, this objective will be minimized. This means that, $\hat{f}_{d,max} = 180°$ and $\hat{f}_{d,min} = 0°$.

The final objective for CSB will be the differential worth. This can be formulated as:

$$\hat{f}_w(\vec{x}) = \sum_{i=1}^{8} \left| \frac{d\rho(\vec{x})}{d\vec{x}_i} \right| \tag{12}$$

Here, $\frac{d\rho(\vec{x})}{d\vec{x}_i}$ is the differential worth of the i-th control drum and is calculated using the derivative form of the hybrid model. Here, $\hat{f}_{w,max} = 3800 \frac{\text{pcm}}{\text{rad}}$ and $\hat{f}_{w,min} = 0 \frac{\text{pcm}}{\text{rad}}$. $\hat{f}_{w,max}$ was calculated by running an independent, single objective optimization problem to estimate the maximum possible total drum worth. Unlike the previous objectives, this objective will be maximized in CSB.

3.3. NeuroEvolution Optimization with Reinforcement Learning (NEORL)

NEORL is a set of implementations of hybrid algorithms combining neural networks and evolutionary computation based on a wide range of machine learning and evolutionary intelligence architectures [46]. NEORL has been developed by some of the authors of this current study. NEORL offers algorithms for large-scale optimization problems relevant to operation and optimization research, engineering, business, and other disciplines.

In this work, we utilize various optimization algorithms from NEORL. First, we used the implementation of differential evolution (DE) and evoluationary strategies (ES) to represent evolutionary algorithms. Second, we utilize the implementation of the grey wolf optimizer (GWO), Harris hawks optimization (HHO), particle swarm optimization (PSO) and moth flame optimization (MFO) as examples of swarm algorithms. This work offers an opportunity to test a variety of NEORL algorithms in a practical application.

Aside from the optimization algorithms, we utilize the algorithm parameter tuning capabilities of NEORL, mainly grid search, to tune the parameters of the prescribed algorithms. Additionally, these methods are used to tune the neural network hyperparameters (e.g. number of nodes, number of layers, learning rate) for the quadrant power surrogate model described in Section 2.2.

Regarding the optimizer hyperparameters, for DE, we tune the population size, mutation weight, and crossover probability. For ES, we consider tuning number of individuals to survive to the next generation (μ), mutation probability, and crossover probability. For PSO, the cognitive speed constant, the social speed constant, number of particles in the swarm, and speed mechanism are tuned. It is worth highlighting that NEORL supports three different PSO variants which we consider in the tuning process. These are PSO with constriction coefficient [47], PSO with time-varying inertia weighting [48], and PSO with global local best inertia [49]. For the other swarm algorithms, we tune the population size.

4. Results and Discussion

4.1. Quadrant Power Model Results

Following both grid and random hyperparameter searches, the final results of the optimum DNN architecture are listed in Table 1. The DNN features 5 layers, ReLU activation, a constant learning rate of 9.00E-04, and mini-batch size of 8. The inputs to the DNN are the 8 drum angles, while the outputs are the four quadrant powers, which are then used to calculate QPTR. The training and validation losses for this DNN are plotted in Figure 3, showing very good convergence trend and negligible overfitting (i.e. training and validation losses are close). Lastly, in Table 2, we benchmark the DNN model against other machine learning methods in modeling the relative quadrant power. Obviously, the DNN model carries impressive metrics compared to random forests, decision trees, and linear regression, through achieving larger R^2 and smaller MAE and RMSE in the test set compared to other methods. Therefore, we can certainly use the current DNN model in the optimization process.

Table 1: DNN optimized hyperparameters for the quadrant power model

Item	Value
Number of layers	5
Number of nodes per layer	437, 258, 101, 75, 35
Activation function	ReLU
Learning rate	9.00E-04
Loss function	MAE
Batch size	8
Number of epochs	200
Training data	2404
Test data	600
Validation Split	0.2

Table 2: Comparison of testing metrics for different quadrant power models

Method	Test MAE	Test RMSE	Test R^2
Deep Neural Network (DNN)	4.62E-04	6.05E-04	0.98
Random Forest (RF)	1.68E-03	2.11E-03	0.75
Decision Trees (DT)	2.75E-03	3.57E-03	0.29
Linear Regression (LR)	3.41E-03	4.23E-03	0.00

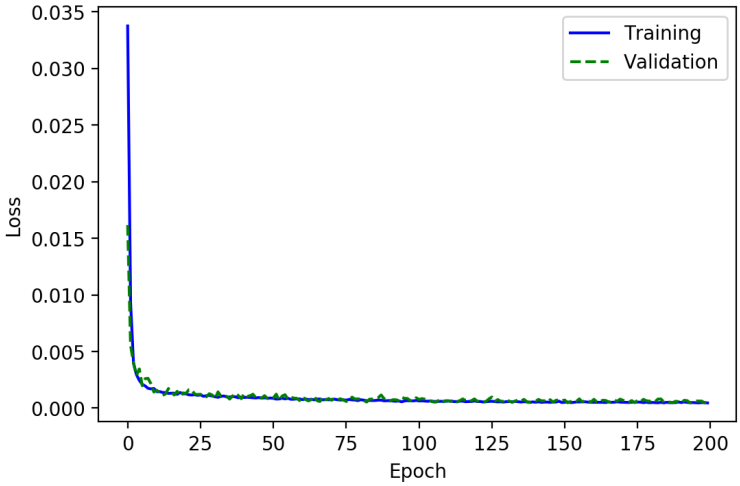

Figure 3: Convergence of the training and validation losses for the DNN quadrant power model.

4.2. Scalarization and Weight Selection

In Section 3.2, three objectives each for CSA and CSB were presented in mathematical terms. The concept of scalarization was also introduced which is a method to combine the three objectives into a single objective function by making some $F(\vec{x})$ which is a linear combination of the individual objectives. The formation of $F(\vec{x})$ requires weights to be assigned to each of the objectives to assert their relative importance. Although each of the objectives is scaled to fall roughly within the 0 to 1 range, differences in the distributions of these objectives within those ranges prevent weights being immediately assigned by some perceived importance of each parameter. Instead, the optima that a particular set of weights may yield need to be analyzed in order to determine the appropriateness of that set of weights. This can be done by running many independent optimizations and generating distributions of the objective for the solution set. Analysis of the solution set is application specific, some considerations when determining the appropriateness of the weights are given below:

- Objective Importance: The more important objectives (as determined by expert opinion) should be prioritized. The reported optima should yield satisfactory results for important objectives before other objectives.
- Objective Competition: Some objectives may directly compete with one another. Identifying interacting objectives may help focus the weight selection process by focusing on these interacting objectives.
- Optima Consistency: Complex optimization algorithms often do not yield identical optima. Solution consistency may be increased with certain sets of weights.
- Solution Set Modality: Perhaps a specific case of optima consistency, some sets of weights may cause the solutions for a single set of weights to group into distinct groups of optima. This can cause unpredictability in the reported optima from the algorithm and should typically be avoided.

Being the first step in the optimization process, the best algorithm with optimally performing parameters has not yet been found. Fortunately, in this early step, the optimization algorithm does not change the mapping of the individual objectives into the space of $F(\vec{x})$. Therefore, the optimization algorithm used in this step should have minimal effect on the selected weight set. As such, ES with 150 generations is used for both CSA and CSB. There is no specific reason ES is used here, other algorithms could have been used as well.

The results from performing around 900 independent optimization routines for each of three separate sets of weights for CSA is given in Figure 4. In this figure, each row of plots corresponds to one of the objectives associated with CSA. Each column corresponds to a different set of weights where w_c is the weight assigned to the scaled f_c, w_p is assigned to f_p and w_d to f_d. Although more sets of weights were run, these three were selected for display as a platform for discussion. First, as the weights for a particular objective increase, the reported optima tends to favor that objective. For example, moving from left to right, w_c increases which causes the distribution of $\hat{f}_c$ to group more tightly near 0. However, this is not necessarily always true. Interactions between objectives can cause unpredictable behavior in the optima distributions. Note how between column 1 and column 3 w_p decreases while the mean of the distribution for $\hat{f}_p$ tends to decrease as well. This is counter-intuitive because, at first glance, the decreasing w_p should diminish the importance of this objective. However, the results show that the decrease in w_d weakens the competing objective $\hat{f}_d$ which allows for better $\hat{f}_p$ despite the decreased w_p.

From this figure, the set of weights were $w_c = 0.50, w_p = 0.40$ and $w_d = 0.10$ is preferred over the other weights. With this set of weights, both $\hat{f}_c$ and $\hat{f}_p$ tend to group nearer to 0 at the expense of $\hat{f}_d$. $\hat{f}_d$ is the objective associated with the travel distance of the drum which is of minimal importance compared to the other objectives. Furthermore, the slightly more favorable $\hat{f}_c$ objectives associated with column 3 comes with the large cost to $\hat{f}_p$ compared to column 2. In fact, there is a notably different distribution associated with $\hat{f}_p$ between these two columns. For further analysis of these two optima groups, Figure 5 is provided. The plot given in the left portion of this figure shows the strong competition between the $\hat{f}_d$ and $\hat{f}_p$ objectives. This can be seen from the negative correlation between these two objectives, a more favorable result in $\hat{f}_d$ tends to lead to a less favorable result in $\hat{f}_p$ and vice-versa. Regardless, it is clear that the $[w_c = 0.50, w_p = 0.40, w_d = 0.10]$ weight set results in more favorable optima for the $\hat{f}_p$ objective—the more important objective between the two shown in the plot. In the right portion of Figure 5, the mean optimal drum positions are shown. A drum angle of 0° corresponds to the drum position where the absorbing material is fully facing the core, all drum angles are contained in -180° to 180°. Therefore, the absolute value of the drum angle is indicative of how withdrawn the absorbing material is from the core. The standard deviations of the optima locations are also shown with error bars. Overall, the angle of drum 1 has the largest differences between the two weight sets. The weight set which places the lower importance on $\hat{f}_d$ favors optima with a less withdrawn control drum in position 1. Being the drum which tends to have the largest rotation angle, $\hat{f}_d$ will seek to minimize the rotation angle of this drum alone. These deviations are similar for each drum across both weight sets so there

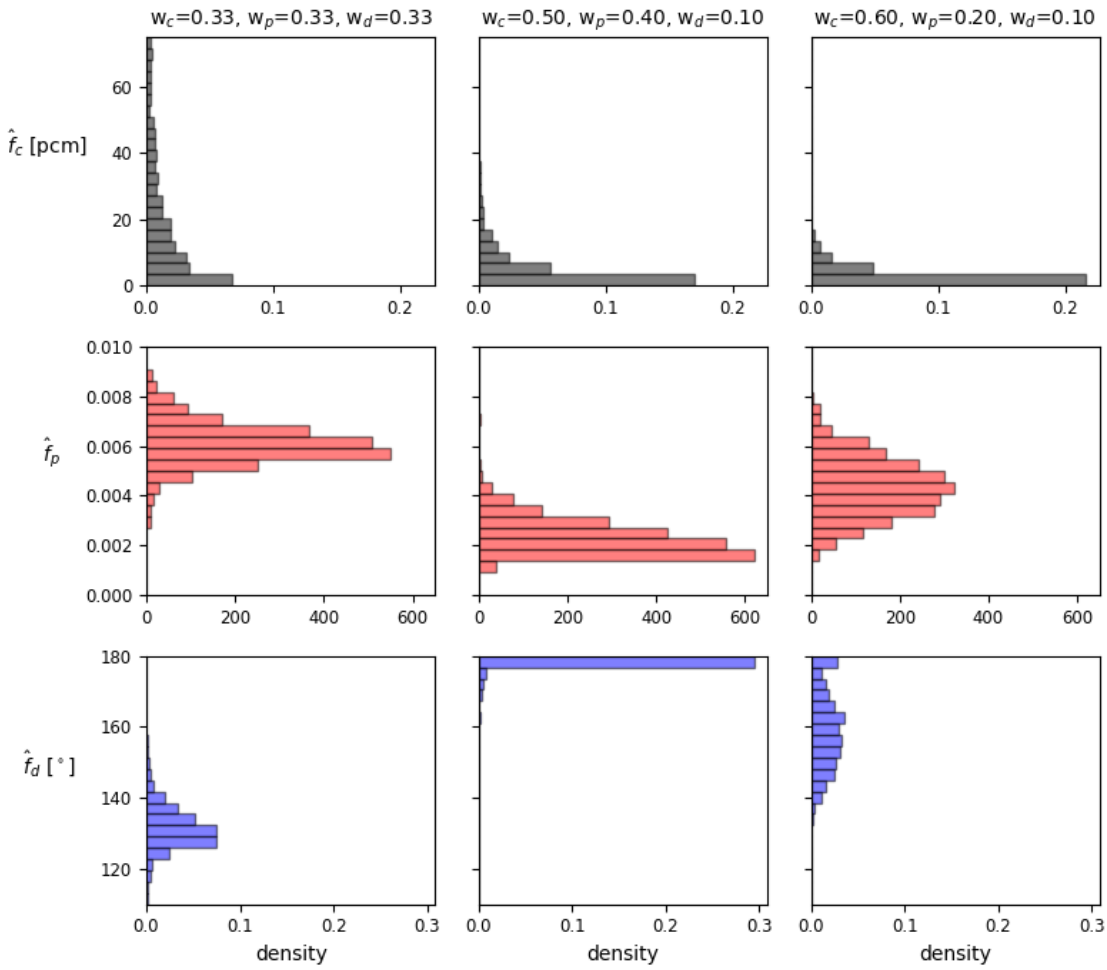

Figure 4: Objective histograms calculated with multiple independent optimization routines run for three weights for CSA.

is no clear benefit to be gained by selecting a particular set of weights for any advantage in the consistency of the optima results. Moving forward, $[w_c = 0.50, w_p = 0.40, w_d = 0.10]$ will be used for CSA due primarily to the favorable balance observed in the $\hat{f}_p$ and $\hat{f}_c$ objectives.

The results from performing around 900 independent optimization routines for each of three separate sets of weights for CSB is given in Figure 6. These distributions are more consistent across the different weight sets than those for CSA. The general shape of the distributions for a fixed objective do not change across weight sets. For $\hat{f}_c$, an exponential-shaped distribution forms as w_c is increased. For $\hat{f}_p$, the peak of the distribution roughly follows what is expected based on w_p. That is, lower w_p values lead to distributions with higher $\hat{f}_p$ values. Finally for $\hat{f}_w$, the width of the distribution increases as w_w is decreased. Although the peak of the distributions associated with this objective do tend to decrease as w_w decreases, the weight set with the smallest w_w still yields acceptable objectives. For all objectives, the mean of an objective distribution responds predictably to a change in the weight of the corresponding objective. As such, the selection of weights is more straightforward in this case. $[w_c = 0.55, w_p = 0.40, w_w = 0.05]$ is selected because it gives strong results for $\hat{f}_p$

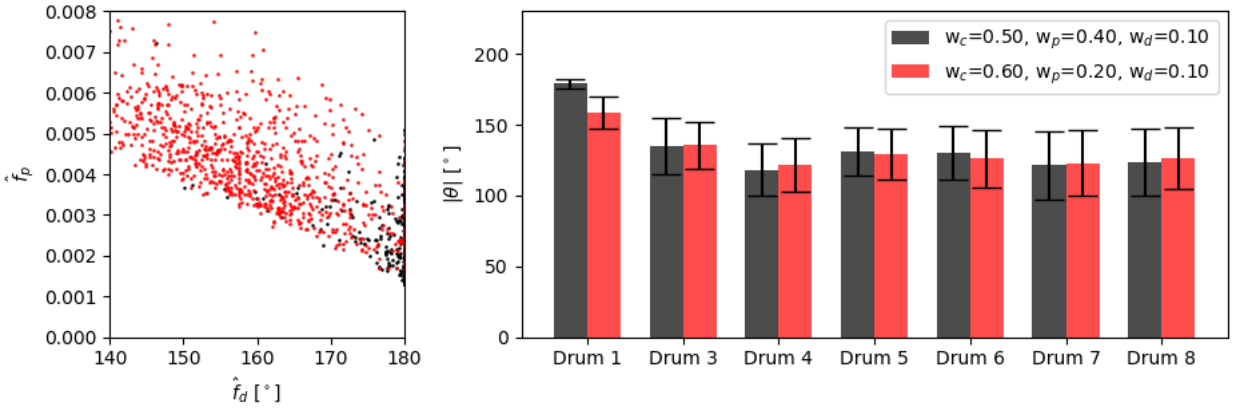

Figure 5: Optima distributions associated with two weight sets.

while having a minimal effect on $\hat{f}_c$. Moving forward, $[w_c = 0.55, w_p = 0.40, w_w = 0.05]$ will be used for CSB.

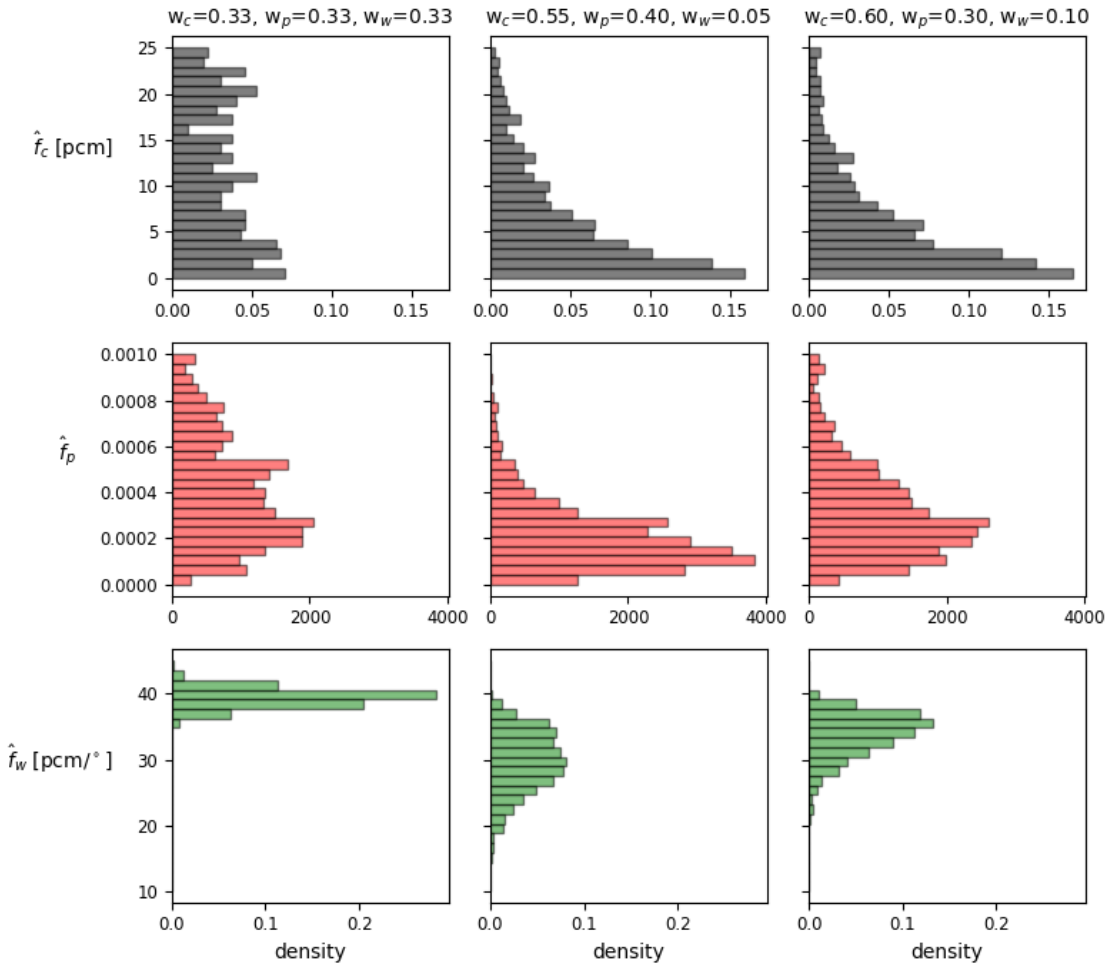

Figure 6: Objective histograms calculated with multiple independent optimization routines run for three weights for CSB.

4.3. Algorithm Parameter Selection

For both CSA and CSB, an optimal parameter search for six algorithms was performed. The results of this search are shown in Table 3 and Table 4. This parameter search was done using a simple grid-based procedure, minimal computational time was dedicated to this task as computational time is better spent focusing on the optimization problem itself, not on the optimization of the algorithm—provided it shows reasonably competitive performance. Also, some algorithms (e.g. ES, PSO, DE) have more parameters to tune than the others, which could highlight that parameter search difficulty is also algorithm-dependent. The number of tunable parameters defines the dimension of the search space for the tuning process, making the computational cost of tuning these parameters very expensive for large numbers of parameters. The reader should notice that the selection of these tunable parameters came from the authors' experience of their sensitivity on the algorithm performance.

Table 3: Parameters for six optimization algorithms for CSA.

Algorithm	Abbreviation	Parameter	Value
Evolution Strategies	ES	μ	25
		crossover probability	0.6
		mutation probability	0.3
Differential Evolution	DE	population size	20
		mutation weight	0.4
		crossover probability	0.3
Grey Wolf Optimization	GWO	population size	45
Harris Hawks Optimization	HHO	population size	55
Moth Flame Optimization	MFO	population size	55
Particle Swarm Optimization	PSO	number of particles	30
		cognitive speed constant	2.15
		social speed constant	2.15
		speed mechanism	time-varying inertia weighting

Table 4: Parameters for six optimization algorithms for CSB.

Algorithm	Abbreviation	Parameter	Value
Evolution Strategies	ES	μ	30
		crossover probability	0.7
		mutation probability	0.3
Differential Evolution	DE	population size	40
		mutation weight	0.9
		crossover probability	0.2
Grey Wolf Optimization	GWO	population size	30
Harris Hawks Optimization	HHO	population size	30
Moth Flame Optimization	MFO	population size	30
Particle Swarm Optimization	PSO	number of particles	50
		cognitive speed constant	2.15
		social speed constant	2.10
		speed mechanism	time-varying inertia weighting

4.4. Algorithm Objective Comparisons

Here, the search strategies of these six optimization algorithms will be compared to one another. Despite the identical purpose for these algorithms—that is to minimize $F(\vec{x})$—the approach that each algorithm uses can vary dramatically. A useful approach when analyzing these algorithms is qualitatively placing their behavior on a spectrum of exploration versus exploitation. Exploitative behavior in an algorithm will rely on knowledge of the objective function obtained in past function evaluations to form some guess for the location of the optima and aggressively pursue this speculated optima with new function evaluations. This behavior makes algorithms susceptible to local optima entrapment where the algorithm will prematurely converge to a local optima and miss the global optima of the function. Exploratory behavior in an algorithm will not actively seek out the optima of the objective function, instead, it will simply seek to further explore the input space. This behavior can potentially cause algorithms to never converge to an optima at all. It is the problem-specific balance of these two behaviors which makes a particular algorithm perform better than others.

Figure 7 and Figure 9 show some statistics surrounding the objective function solutions over each generation for the six algorithms for CSA and CSB, respectively. In the context of evolutionary and swarm optimization algorithms the fitness of an individual in a generation is equal to the objective function evaluated at the location of that individual. The generation average (gen. ave.) fitness is shown with the green line for each algorithm. This quantity is defined as the average fitness of all individuals in an algorithm in a generation. The 1-σ bound around the generation average (gen. 1-σ) fitness is also shown. This is calculated by taking the standard deviation of the fitness values of all individuals in a population. The maximum (gen. max) and minimum (gen. min) fitness for each generation are shown as well. Each of these algorithms was allotted 10,000 function evaluations to perform the optimization, the total number of generations run is adjusted accordingly.

Separately, Table 5 and Table 6 are given to characterize the reported optima from these algorithms applied to CSA and CSB, respectively. Due to the randomness associated with these algorithms, the reported optima can be different for each run of the optimization algorithm. As such, 50 independent optimization routines were performed for each of the six algorithms. The only difference between the 50 runs is the random seed which changes the initial population guess every run. Both the mean and standard deviations in the final best fitness are reported.

For the results associated with CSA seen in Figure 7, there is very different behavior between the six algorithms. On the extremes, MFO exhibits the most exploitative behavior out of all algorithms. Around generation 50, the heterogeneity in the population quickly decreases which is indicative of convergence to an optima. ES also demonstrates exploitative behavior because the mean fitness of the population drifts close to the minimum population fitness relatively early on in the optimization process. However, the mutation operator consistently perturbs selected individuals to the end of the problem preventing the maximum fitness of the population to converge to its mean. This maintains some degree of exploratory behavior for this algorithm late into the optimization process, despite the predominantly exploitative behavior demonstrated in this context. On the other side, DE and PSO maintain diversity in their populations until the end of the

optimization process. As compared to the other algorithms, the large generation 1-σ and generation average seen in these algorithms demonstrates consistent exploratory behavior. Moving on to HHO, this algorithm shows a significant shift in its behavior which occurs around generation 100. This shift is characteristic of the HHO algorithm where, as described by [29], a transfer from an exploration to an exploitation phase generally occurs as a function of generation number. In the exploration phase, a large degree of heterogeneity is maintained in the population that is rapidly lost when transitioning to the exploitation phase. Finally, GWO demonstrates a seemingly linear reduction in its population heterogeneity in that the population average, 1-σ and maximum all decrease linearly with respect to generation number. Again, this is a characteristic of GWO that is captured in a parameter denoted as a in [28]. Mathematically well-described in the aforementioned paper, a is essentially the allowable proximity of individuals in a population to the predicted optima. This parameter is decreased linearly as a function of generation, hence the roughly linear decrease in population diversity as a function of generation.

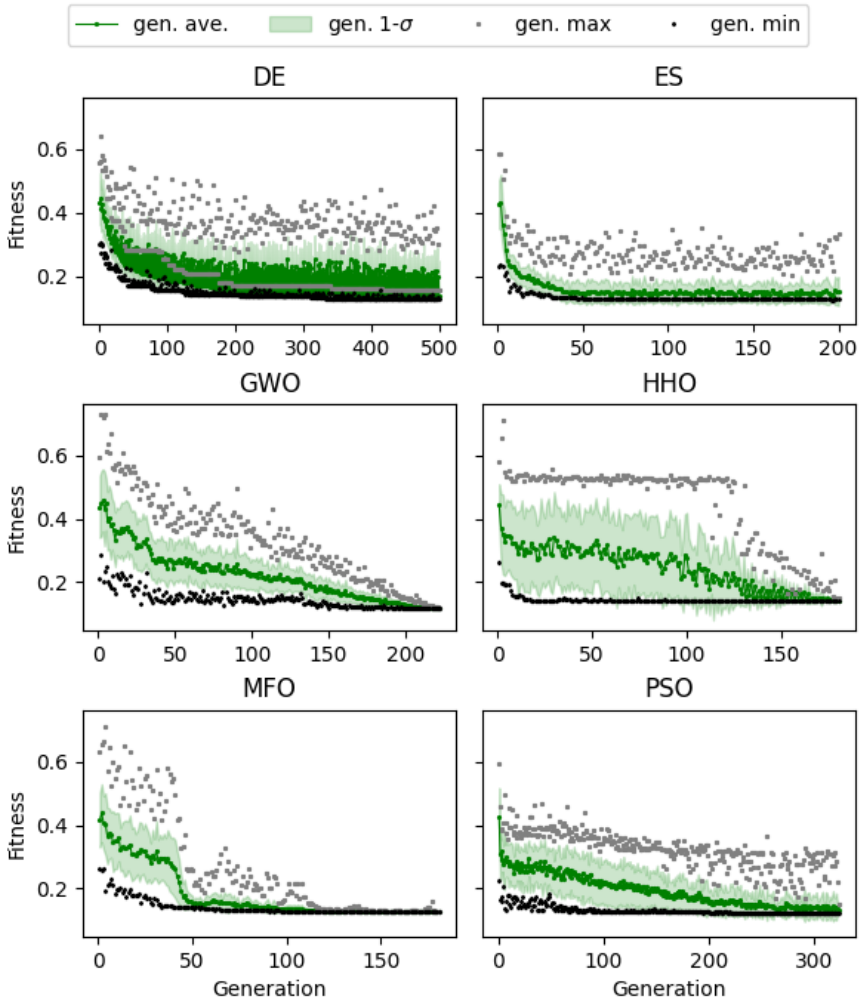

Figure 7: Generation characteristics for optimization algorithms for CSA.

Now, the behavior of the reported optima for CSA will be discussed as shown in Table 5. The rankings

for best mean optima is MFO as best, then DE, ES, GWO, PSO and finally HHO. Interestingly, this ranking is similar if standard deviation is ranked: DE, MFO, ES, PSO, GWO, then HHO. These results suggest that there are clear better choices for optimization algorithms for CSA because DE and MFO give the best optima with the most consistency. Furthermore, ES, DE and PSO required more effort to obtain tuned parameters as shown in Section 4.3 because they had more parameters to tune. The cost of obtaining these tuned algorithm parameters can be very high. From these results, there is no clear advantage that algorithms containing more tuneable parameters have on those with less. In fact, the best performing algorithm—MFO—only has a single parameter. Therefore, these results suggest that for this application, the extra computational effort required to tune parameters for ES, DE and PSO may not be justified as they offered no clear performance advantage over the other algorithms with only a single parameter (GWO, HHO, MFO). Instead, this computational effort should be spent on running more generations for the actual optimization problem. Another interesting point of discussion is how some algorithms tend to prioritize different objectives despite the scalarized objective function F being the same for all algorithms. For example, MFO consistently reports optima where $\hat{f}_c$ is well-below those shown by any other algorithm. To a lesser extent, PSO demonstrates similar behavior for $\hat{f}_c$. Both of these algorithms also consistently show the worst value for $\hat{f}_d$ where every optimization routine resulted in the worst-possible value for this objective at 180°. This may be the result of how the characteristics of the search space interact with the search behavior of the algorithm. As an example of differing search behaviors, the GWO algorithm tends to have individuals move in a grid-like pattern around the predicted optima mimicking the circling behavior of wolves while the HHO algorithm has individuals travel directly at the predicted optima mimicking a hawk's dive towards prey.

Table 5: Statistics from 50 independent optimization routines performed with six algorithms for CSA.

Algorithm	Statistic	F	$\hat{f}_c$ [pcm]	$\hat{f}_p$	$\hat{f}_d$ [°]
ES	Mean	0.1286	5.156	2.434×10^{-3}	179.6
	Std.	8.185×10^{-3}	4.722	6.964×10^{-4}	1.049
DE	Mean	0.1259	18.06	2.016×10^{-3}	179.2
	Std.	5.622×10^{-3}	20.62	5.348×10^{-4}	2.054
GWO	Mean	0.1368	2.150	3.143×10^{-3}	180.0
	Std.	1.383×10^{-2}	5.343	1.176×10^{-3}	7.018×10^{-2}
HHO	Mean	0.1495	14.82	4.900×10^{-3}	161.5
	Std.	1.604×10^{-2}	50.15	1.455×10^{-3}	16.81
MFO	Mean	0.1253	0.2202	2.175×10^{-3}	180.0
	Std.	7.725×10^{-3}	0.3730	6.672×10^{-4}	1.154×10^{-2}
PSO	Mean	0.1395	0.7930	3.397×10^{-3}	180.0
	Std.	1.117×10^{-2}	2.437	9.646×10^{-4}	0.000

Next, the convergence of each objective for MFO will be analyzed for CSA. MFO is chosen for analysis here because it yielded the best mean F value out of all the methods chosen. The behavior of each objective as a function of $F(\vec{x})$ evaluations for a single optimization routine is shown in Figure 8. Early on, the optimization routine finds the $\hat{f}_d$ = 180° that was observed in Table 5. $\hat{f}_c$ and $\hat{f}_p$ are continually updated during the remainder of the optimization process as new optima for F are found. Moving forward through

the optimization, the best F is monotonic in that the best value for F is only replaced when a new, lower value for F is found. On the other hand, the trends associated with other objectives do not necessarily require monotonicity. An improvement in the running optima can result in an increase in one objective provided that the overall value of F decreases. This is the reason for the nonmonotonic behavior in $\hat{f}_c$ throughout the optimization process. From this figure, it is clear that all objectives converge to reasonable values.

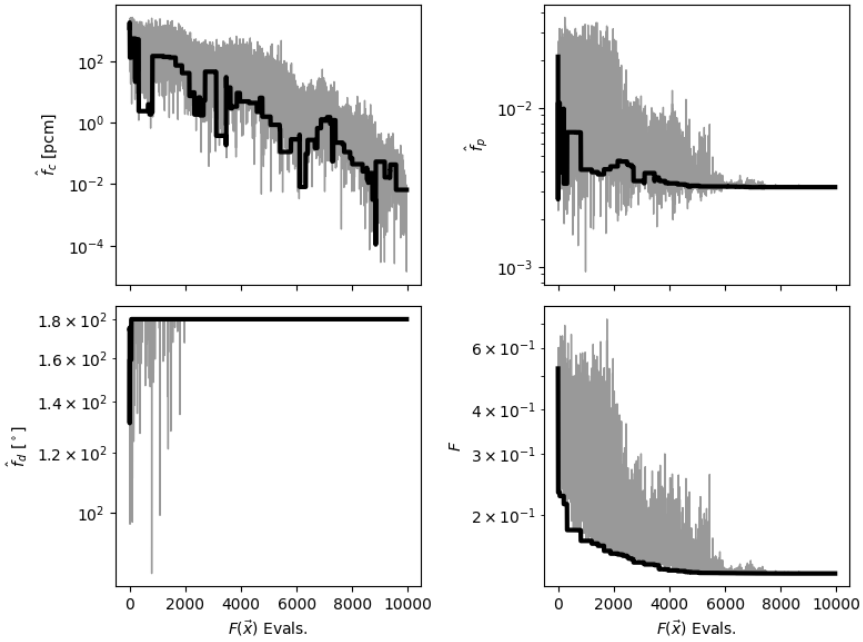

Figure 8: Objective convergence for MFO optimization performed on CSA. Black indicates characteristics of the running best optima, gray indicates areas still being explored by the optimization algorithm.

Now, on to the results associated with CSB. Figure 9 shows the population characteristics as a function of generation number for the six algorithms. In general, algorithms seem to demonstrate similar behavior as they did for CSA. More specifically, ES shows the same homogeneous populations, with the population mean and maximum fitnesses lying much closer to the minimum fitness for the majority of the optimization process. MFO again demonstrates little exploration because the population becomes homogeneous very early in the optimization process. GWO and HHO demonstrate similar behavior as their counterparts shown in CSA, with the linear transition from exploration to exploitation seen in the case of GWO and the sharp transition for HHO. DE is the most exploratory of all the algorithms, maintaining population heterogeneity until the end of the optimization routine. However, PSO demonstrates stronger exploratory characteristics then it did for CSA. The generation 1-σ stays relatively large despite the generation average coming close to the generation minimum at the end of the optimization process.

Moving on to the results shown in Table 6 the algorithms ranked for the best mean F to worst mean F are: ES, MFO, GWO, PSO, HHO, DE. As demonstration of the no free lunch theorem [50], the performance of these different optimization algorithms is application specific. These rankings are different from the rankings in CSA. In general, the no free lunch theorem states that no single optimization algorithm performs better

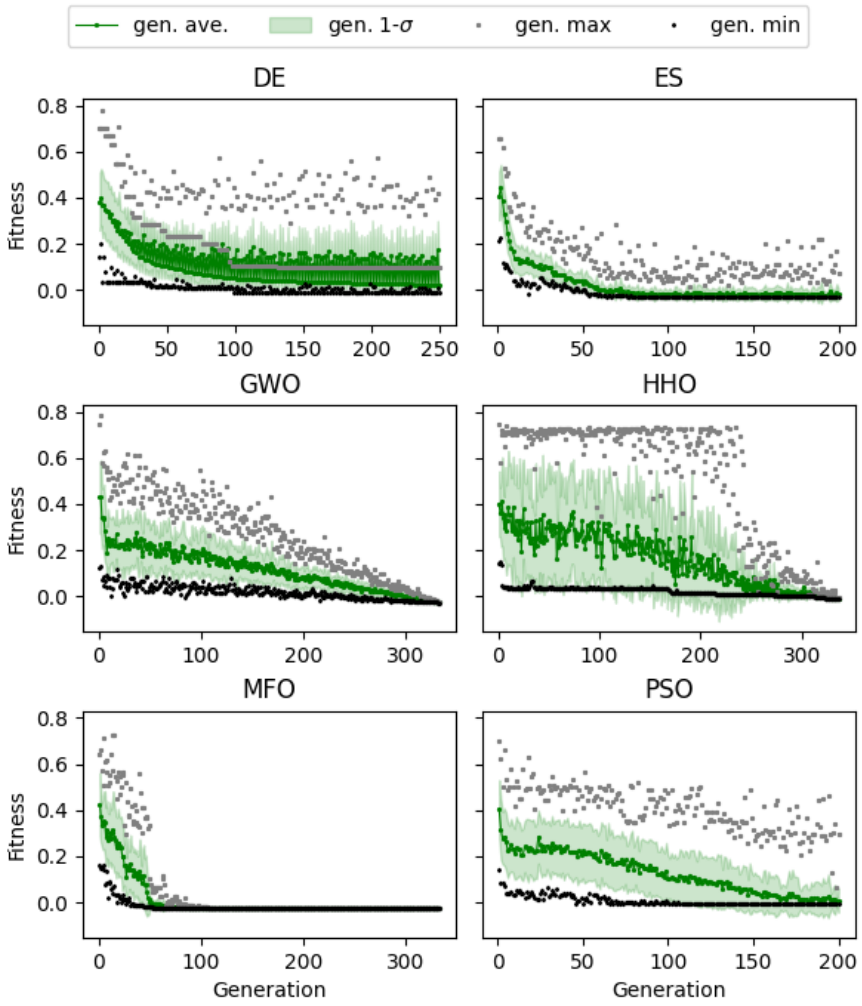

Figure 9: Generation characteristics for optimization algorithms for CSB.

than all others for all possible optimization problems. Here, this means that the optimization algorithms that perform well on CSA are not necessarily the ones that perform the best on CSB. That being said, MFO performs well in both cases. As such, with more exploration, it may be possible to identify an optimization algorithm which performs better for a broader set of optimization problems concerning control drum position optimization. As was done for CSA, the rankings for the spread in reported optima (from least to most spread) are ES, PSO, MFO, DE, GWO, HHO. Again, this ranking is moderately similar to the ranking for the mean F, indicating that the algorithms which report the best optima are also the most consistent, making ES the clear choice. Furthermore, the prioritization of $\hat{f}_c$ demonstrated by PSO and MFO is apparent in CSB. This was observed in CSA as well.

For deeper analysis of the best performing algorithm, Figure 10 is given which shows the progression of each objective associated with an optima as a function of number of $F(\vec{x})$ evaluations in a ES optimization calculation. As with Figure 8, the black lines show the objectives associated with the running best optima. The black line must be monotonically decreasing for F but can increase for particular objectives if another

Table 6: Statistics from 50 independent optimization routines performed with six algorithms for CSB.

Algorithm	Statistic	F	$\hat{f}_c$ [pcm]	$\hat{f}_p$	$\hat{f}_w$ [pcm/°]
ES	Mean	-2.736 $\times 10^{-2}$	5.113	1.400 $\times 10^{-4}$	29.02
	Std.	5.015 $\times 10^{-3}$	6.512	9.268 $\times 10^{-5}$	44.97
DE	Mean	-1.032 $\times 10^{-2}$	25.07	7.155 $\times 10^{-4}$	24.91
	Std.	6.651 $\times 10^{-3}$	21.99	3.634 $\times 10^{-4}$	5.296
GWO	Mean	-2.391 $\times 10^{-2}$	2.831	2.016 $\times 10^{-4}$	26.34
	Std.	1.274 $\times 10^{-2}$	5.052	6.685 $\times 10^{-4}$	4.655
HHO	Mean	-1.601 $\times 10^{-2}$	4.817	6.991 $\times 10^{-4}$	27.50
	Std.	1.355 $\times 10^{-2}$	9.367	8.154 $\times 10^{-4}$	5.364
MFO	Mean	-2.404 $\times 10^{-2}$	1.302 $\times 10^{-4}$	6.991 $\times 10^{-6}$	22.98
	Std.	5.591 $\times 10^{-3}$	4.868 $\times 10^{-4}$	4.260 $\times 10^{-5}$	5.304
PSO	Mean	-1.948 $\times 10^{-2}$	0.3613	2.666 $\times 10^{-5}$	19.01
	Std.	5.310 $\times 10^{-3}$	0.8164	9.344 $\times 10^{-5}$	4.814

objective is reduced for a new best optima. The results shown here indicate that the objectives may be less competitive in CSB than those in CSA, in general, the trends are more monotonic. This is also seen in Section 4.2 where weight selection for CSB was more straightforward than it was for CSA. Also, F seems to stabilize earlier in the algorithm as compared to CSA. This was verified with multiple independent runs. This may suggest an easier optimization problem overall. Interestingly, exploration in the $\hat{f}_c$ and $\hat{f}_p$ is observed until the end of the algorithm by the thickness of the gray area associated with the plot showing this objective into the end of the optimization process. From around 4000 function evaluation onward, the explored domain seems to fail to achieve the strong $\hat{f}_p$ objective found early in the routine. This can be seen by the gap between the black line and grey area in the plot for this objective. This may suggest that the algorithm struggled to find domains where F was optimized in a way that led to strong $\hat{f}_p$ objectives. The final reported optima have reasonable values for each objective.

4.5. Optima Verification

For completeness, the validity of the surrogate models should be evaluated at the reported optima. One important aspect of any surrogate model is that the accuracy of the model can vary in different parts of its domain. Often in optimization, highly specific parts of the input domain are sought out because of their desirable characteristics which may result in domains where the surrogate models are less accurate than expected if the model error is evaluated on the global scale. As such, two final optimization calculations were performed for CSA and CSB using MFO and ES, respectively. Due to the nature of the Monte Carlo calculation method associated with the Serpent model, there are uncertainties associated with the final results. At the optima reported for CSA, the surrogate model overpredicted criticality by 74 pcm ± 13 pcm. This misprediction is reasonably within the understood model errors described in [40]. The QPTR reported by the surrogate model is 1.011, the QPTR reported by Serpent is 1.010 with negligible uncertainty from the Monte Carlo calculation method. Again, this misprediction is reasonably within the known model errors described in Table 2. From these results, the use surrogate models did not result in any unpredictable behavior from CSA. At the optima reported by CSB, the surrogate model underpredicted criticality by 54 pcm ± 13 pcm,

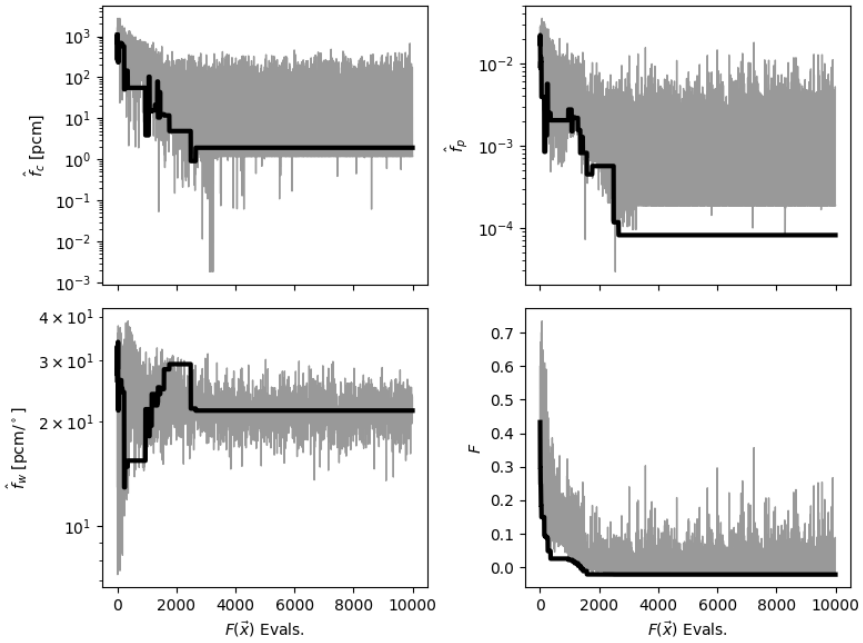

Figure 10: Objective convergence for ES optimization performed on CSB. Black indicates characteristics of the running best optima, gray indicates areas still being explored by the optimization algorithm.

which certainly lies in the expected model errors. The surrogate model predicted a QPTR of 1.020, the QPTR reported by the surrogate model is 1.019 with negligible uncertainty from the Monte Carlo calculation method. Again, this error is small. Finally, the differential worth predicted by the surrogate model was verified using a central difference approximation using two different simulations from the Serpent code. The differential worth reported by the surrogate model is 29.24. pcm/°, the differential worth approximated using Serpent is 31.85 pcm/°± 0.55 pcm/°. These values show strong agreement between the surrogates and the high-fidelity Monte Carlo Serpent calculations. Overall, the results from the surrogate model predictions at the optima agree very well with those from the Serpent model.

5. Conclusions

In this paper, six optimization algorithms are demonstrated on two nuclear engineering problems specific to the operation of microreactors. They both involve optimizing control drum configurations to yield favorable operating conditions. The optimization problem referred to as CSA is based on a scenario where one control drum becomes inoperable. In this scenario, three objectives are identified for safe operation of the reactor: (1) criticality, (2) even power distribution quantified by the QPTR and (3) minimal control drum travel distance. The next optimization problem referred to as CSB centers around normal operation where the three objectives are: (1) criticality, (2) target power distribution quantified by QPTR and (3) maximum control drum worth. For both cases, scalarization is used to assimilate the three objectives into a single objective function F. Following this, the six optimization algorithms applied to this objective function are:

- Evolution Strategies (ES) [25]
- Differential Evolution (DE) [26]
- Grey Wolf Optimization (GWO) [28]
- Harris Hawks Optimization (HHO) [29]
- Moth Flame Optimization (MFO) [30]
- Particle Swarm Optimization (PSO) [27]

For both CSA and CSB, all algorithms demonstrated some ability to reasonably traverse the search space and report optima. For CSA, the MFO algorithm performed the best, although all algorithms reported reasonable optima. For CSB, the ES algorithm performed the best. Furthermore, it was observed in both cases that, in general, the algorithms which provided the best mean optima are also those which had the least spread of the reported optima. This makes it easier to identify the best algorithm for a particular problem because the algorithm which has the potential to report the best optima is also the most consistent. Finally, the optima found by CSA and CSB were run through the original Serpent model used to train the surrogate models to verify results. Strong agreement was observed between the objectives predicted by the surrogate models and those calculated using Serpent.

6. CRediT Statement

Dean Price: Conceptualization, Methodology, Software, Formal Analysis, Investigation, Data Curation, Writing - Original Draft, Visualization. **Majdi Radaideh:** Conceptualization, Methodology, Software, Formal Analysis, Investigation, Writing - Original Draft, Supervision. **Brendan Kochunas:** Conceptualization, Resources, Writing - Review & Editing, Supervision, Project Administration, Funding Acquisition.

7. Acknowledgments

This material is based upon work supported under an Integrated University Program Graduate Fellowship. This work was supported by funding received from the DOE Office of Nuclear Energy's Nuclear Energy University Program under contract number DE-NE0008887 as well.

8. Data Availability

The code and data needed to reproduce the results found in this paper can be found in the public Github repository at this link: https://github.com/deanrp2/MicroControl.

References

[1] J. Vujić, R. M. Bergmann, R. Škoda, M. Miletić, Small modular reactors: Simpler, safer, cheaper?, Energy 45 (1) (2012) 288–295.

[2] R. H. Stewart, T. S. Palmer, B. DuPont, A survey of multi-objective optimization methods and their applications for nuclear scientists and engineers, Progress in Nuclear Energy (2021) 103830.

[3] J. Pevey, B. Hiscox, A. Williams, O. Chvala, V. Sobes, J. W. Hines, Gradient-informed design optimization of select nuclear systems, Nuclear Science and Engineering (2021) 1–13.

[4] J. L. Pevey, C. Salyer, O. Chvala, V. Sobes, J. W. Hines, Multi-objective design optimization of a fast spectrum nuclear experiment facility using artificial intelligence, Annals of Nuclear Energy 162 (2021) 108476.

[5] X. Li, J. Cheng, D. Zhang, Z. Dai, Design and optimization of passive residual heat removal system for lead-bismuth reactor svbr-100, International Journal of Energy Research 45 (8) (2021) 12124–12146.

[6] A. Naserbegi, M. Aghaie, A. Zolfaghari, Implementation of grey wolf optimization (gwo) algorithm to multi-objective loading pattern optimization of a pwr reactor, Annals of Nuclear Energy 148 (2020) 107703.

[7] G. T. Phan, Q. B. Do, Q. H. Ngo, T. A. Tran, H.-N. Tran, Application of differential evolution algorithm for fuel loading optimization of the dnrr research reactor, Nuclear Engineering and Design 362 (2020) 110582.

[8] V.-P. Tran, G. T. Phan, V.-K. Hoang, P. N. V. Ha, A. Yamamoto, H.-N. Tran, Evolutionary simulated annealing for fuel loading optimization of vver-1000 reactor, Annals of Nuclear Energy 151 (2021) 107938.

[9] A. Kumar, P. V. Tsvetkov, A new approach to nuclear reactor design optimization using genetic algorithms and regression analysis, Annals of Nuclear Energy 85 (2015) 27–35.

[10] M. I. Radaideh, I. Wolverton, J. Joseph, J. J. Tusar, U. Otgonbaatar, N. Roy, B. Forget, K. Shirvan, Physics-informed reinforcement learning optimization of nuclear assembly design, Nuclear Engineering and Design 372 (2021) 110966.

[11] M. I. Radaideh, B. Forget, K. Shirvan, Large-scale design optimisation of boiling water reactor bundles with neuroevolution, Annals of Nuclear Energy 160 (2021) 108355.

[12] Y. Song, Z. Zhang, J. Mao, C. Lu, S. Tang, F. Xiao, H. Lyu, Research on fast intelligence multi-objective optimization method of nuclear reactor radiation shielding, Annals of Nuclear Energy 149 (2020) 107771.

[13] D. Clayton, R. Wood, The role of instrumentation and control technology in enabling deployment of small modular reactors, in: Proceeding of the Seventh American Nuclear Society International Topical Meeting on Nuclear Plant Instrumentation, Control, and Human-Machine Interface Technologies, 2010.

[14] B. Mignacca, G. Locatelli, Economics and finance of small modular reactors: A systematic review and research agenda, Renewable and Sustainable Energy Reviews 118 (2020) 109519.

[15] G. Wang, J. Wu, B. Zeng, Z. Xu, W. Wu, X. Ma, State-space model predictive control method for core

power control in pressurized water reactor nuclear power stations, Nuclear Engineering and Technology 49 (1) (2017) 134–140.

[16] S. J. Qin, T. A. Badgwell, A survey of industrial model predictive control technology, Control engineering practice 11 (7) (2003) 733–764.

[17] A. Rubaai, M. J. Castro-Sitiriche, A. R. Ofoli, Dsp-based laboratory implementation of hybrid fuzzy-pid controller using genetic optimization for high-performance motor drives, IEEE transactions on industry applications 44 (6) (2008) 1977–1986.

[18] A. Rubaai, M. J. Castro-Sitiriche, A. R. Ofoli, Design and implementation of parallel fuzzy pid controller for high-performance brushless motor drives: an integrated environment for rapid control prototyping, IEEE Transactions on Industry applications 44 (4) (2008) 1090–1098.

[19] C. Liu, J.-F. Peng, F.-Y. Zhao, C. Li, Design and optimization of fuzzy-pid controller for the nuclear reactor power control, Nuclear Engineering and Design 239 (11) (2009) 2311–2316.

[20] S. Ramachandran, M. Jayalal, A. Riyas, R. Jehadeesan, K. Devan, Application of genetic algorithm for optimization of control rods positioning in a fast breeder reactor core, Nuclear Engineering and Design 361 (2020) 110541.

[21] M. Waintraub, R. Schirru, C. M. Pereira, Multiprocessor modeling of parallel particle swarm optimization applied to nuclear engineering problems, Progress in Nuclear Energy 51 (6-7) (2009) 680–688.

[22] C. Liu, C. Yan, J. Wang, Hybrid particle swarm optimization algorithm and its application in nuclear engineering, Annals of Nuclear Energy 64 (2014) 276–286.

[23] J. C. Bansal, P. K. Singh, N. R. Pal, Evolutionary and swarm intelligence algorithms, Springer, 2019.

[24] S. Sivanandam, S. Deepa, Genetic algorithms, in: Introduction to genetic algorithms, Springer, 2008, pp. 15–37.

[25] H.-G. Beyer, H.-P. Schwefel, Evolution strategies–a comprehensive introduction, Natural computing 1 (1) (2002) 3–52.

[26] R. Storn, K. Price, Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces, Journal of global optimization 11 (4) (1997) 341–359.

[27] J. Kennedy, R. Eberhart, Particle swarm optimization, in: Proceedings of ICNN'95-international conference on neural networks, Vol. 4, IEEE, 1995, pp. 1942–1948.

[28] S. Mirjalili, S. M. Mirjalili, A. Lewis, Grey wolf optimizer, Advances in engineering software 69 (2014) 46–61.

[29] A. A. Heidari, S. Mirjalili, H. Faris, I. Aljarah, M. Mafarja, H. Chen, Harris hawks optimization: Algorithm and applications, Future generation computer systems 97 (2019) 849–872.

[30] S. Mirjalili, Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm, Knowledge-based systems 89 (2015) 228–249.

[31] C. Filippone, K. Jordan, The holos reactor: A distributable power generator with transportable subcritical power modules.

[32] N. E. Stauff, C. Lee, A. Wells, C. Filippone, Design optimization of the holos-quad micro-reactor concept, in: EPJ Web of Conferences, Vol. 247, EDP Sciences, 2021, p. 01005.

[33] N. Stauff, C. Lee, P. Shriwise, Y. Miao, R. Hu, P. Vegendla, T. Fei, Neutronic design and analysis of the holos-quad concept, Tech. rep., Argonne National Lab.(ANL), Argonne, IL (United States) (2019).

[34] C. Lee, C. Filippone, L. Zou, N. Stauff, Core design of the holos-quad micro reactor, in: Transactions of the American Nuclear Society - Volume 123, AMNS, 2020.

[35] R. Hernandez, M. Todosow, N. R. Brown, Micro heat pipe nuclear reactor concepts: Analysis of fuel cycle performance and environmental impacts, Annals of Nuclear Energy 126 (2019) 419–426.

[36] M. F. Khandaq, A. W. Harto, A. Agung, Conceptual core design study for indonesian space reactor (isr), Progress in Nuclear Energy 118 (2020) 103109.

[37] S. A. Hatton, M. S. El-Genk, Sectored compact space reactor (score) concepts with a supplementary lunar regolith reflector, Progress in Nuclear Energy 51 (1) (2009) 93–108.

[38] C. J. Stanley, F. M. Marshall, Advanced test reactor: A national scientific user facility, in: International Conference on Nuclear Engineering, Vol. 48175, 2008, pp. 367–372.

[39] D. M. Nichols, M. A. Reichenberger, A. D. Maile, M. R. Holtz, D. S. McGregor, Simulated performance of the micro-pocket fission detector in the advanced test reactor critical facility, Nuclear Science and Engineering (2021) 1–9.

[40] D. Price, A perturbation-based hybrid methodology for control drum worth prediction applied to the holos-quad microreactor concept, Annals of Nuclear Energy, Accepted.

[41] J. Leppänen, M. Pusa, T. Viitanen, V. Valtavirta, T. Kaltiaisenaho, The serpent monte carlo code: Status, development and applications in 2013, in: SNA+ MC 2013-Joint International Conference on Supercomputing in Nuclear Applications+ Monte Carlo, EDP Sciences, 2014, p. 06021.

[42] U. N. R. Commission, et al., Westinghouse technology systems manual, Retrieved from website: pbadupws. nrc. gov/docs/ML1122/ML11223A214. pdf.

[43] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, X. Zheng, TensorFlow: Large-scale machine learning on heterogeneous systems, software available from tensorflow.org (2015).

[44] D. P. Kingma, J. Ba, Adam: A method for stochastic optimization, arXiv preprint arXiv:1412.6980.

[45] N. Gunantara, A review of multi-objective optimization: Methods and its applications, Cogent Engineering 5 (1) (2018) 1502242.

[46] M. I. Radaideh, K. Du, P. Seurin, D. Seyler, X. Gu, H. Wang, K. Shirvan, Neorl: Neuroevolution optimization with reinforcement learning, arXiv preprint arXiv:2112.07057.

[47] M. Clerc, J. Kennedy, The particle swarm-explosion, stability, and convergence in a multidimensional complex space, IEEE transactions on Evolutionary Computation 6 (1) (2002) 58–73.

[48] Y. Shi, R. Eberhart, A modified particle swarm optimizer, in: 1998 IEEE international conference on evolutionary computation proceedings. IEEE world congress on computational intelligence (Cat. No. 98TH8360), IEEE, 1998, pp. 69–73.

[49] M. S. Arumugam, M. Rao, On the performance of the particle swarm optimization algorithm with various inertia weight variants for computing optimal control of a class of hybrid systems, Discrete Dynamics in Nature and Society 2006.

[50] Y.-C. Ho, D. L. Pepyne, Simple explanation of the no-free-lunch theorem and its implications, Journal of optimization theory and applications 115 (3) (2002) 549–570.

Appendix A. Derivation of Differential Worth Model

To begin, the equations relating to the integral worth model will be presented. These are well-described in [] but will be briefly reviewed here. To begin, there is the integral worth model which can be described by the equation given in Equation A.1.

$$\Delta\rho = \sum_{n}^{N} \tilde{\zeta}^{(n)}(\boldsymbol{\gamma}) \left(\int_{\Theta_{A}^{(n)} \cap \Theta_{A_0'}^{(n)}} j^{-(n)}(\beta)^2 d\beta - \int_{\Theta_{A'}^{(n)} \cap \Theta_{A_0}^{(n)}} j^{-(n)}(\beta)^2 d\beta \right) \tag{A.1}$$

Here, each term in the summation represents the contribution of the n-th drum of N total drums to the change in reactivity. In the case of the current design iteration for the HOLOS-Quad concept, $N = 8$. $\tilde{\zeta}^{(n)}(\boldsymbol{\gamma})$ is a linear function with respect to $\boldsymbol{\gamma}$ which captures the tendency for power shifts to occur in the core. $\boldsymbol{\gamma}$ is a vector of albedo variables which can be calculated according to Equation A.3. Each element in $\boldsymbol{\gamma}$ is the albedo of each control drums.

$$\boldsymbol{\gamma} = \begin{bmatrix} \gamma^{(0)} & \gamma^{(1)} & \dots & \gamma^{(N)} \end{bmatrix}^T \tag{A.2}$$

$$\gamma = \int_{\Theta_{A'}} j^{-}(\beta) d\beta \tag{A.3}$$

$j^{-(n)}$ is a normalized flux shape function which describes the shape of the inward current on the surface of the n-th control drum. $\Theta_{A_0}^{(n)}$ and $\Theta_{A_0'}^{(n)}$ are domains of angular coordinate variables which describe the surface of the n-th control drum in the unperturbed configuration which span the absorbing surface and reflecting surface respectively. The same can be said about $\Theta_{A}^{(n)}$ and $\Theta_{A'}^{(n)}$ except for the perturbed configuration.

The explicit form of $\tilde{\zeta}^{(n)}(\boldsymbol{\gamma})$ used in this study is given in Equation A.4. Here, $\boldsymbol{\alpha}^{(n)}$ is a vector of fitted coefficients which must be determined using a training set.

$$\tilde{\zeta}^{(n)}(\boldsymbol{\gamma}) = \boldsymbol{\gamma}^{*T} \boldsymbol{\alpha}^{(n)} \tag{A.4}$$

where

$$\boldsymbol{\gamma}^{*} = \begin{bmatrix} 1 & \gamma^{(1)} & \gamma^{(2)} & \dots & \gamma^{(N)} \end{bmatrix}^T \tag{A.5}$$

$$\boldsymbol{\alpha}^{(n)} = \begin{bmatrix} \alpha_0^{(n)} & \alpha_1^{(n)} & \alpha_2^{(n)} & \dots & \alpha_N^{(n)} \end{bmatrix}^T \tag{A.6}$$

The full expression for the differential worth model is given as:

$$\frac{d\Delta\rho}{d\theta^{(k)}} = \sum_{n,n\neq k}^{N} \alpha_k^{(n)} \frac{d}{d\theta^{(k)}}\left[\gamma^{(k)}\right]\left(\int_{\Theta_A^{(n)}\cap\Theta_{A_0'}^{(n)}} j^{-(n)}(\beta)^2 d\beta - \int_{\Theta_{A'}^{(n)}\cap\Theta_{A_0}^{(n)}} j^{-(n)}(\beta)^2 d\beta\right) \tag{A.7}$$

$$+\alpha_k^{(k)} \frac{d}{d\theta^{(k)}}\left[\gamma^{(k)}\right]\left(\int_{\Theta_A^{(k)}\cap\Theta_{A_0'}^{(k)}} j^{-(k)}(\beta)^2 d\beta - \int_{\Theta_{A'}^{(k)}\cap\Theta_{A_0}^{(k)}} j^{-(k)}(\beta)^2 d\beta\right) \tag{A.8}$$

$$+\tilde{\zeta}^{(k)}(\boldsymbol{\gamma}) W^{(k)} \tag{A.9}$$

First, for $\frac{d}{d\theta^{(k)}}\left[\gamma^{(n)}\right]$ when $n = k$:

$$\frac{d}{d\theta^{(k)}}\left[\gamma^{(n)}\right] = -j^{-(n)}\left(\theta^{(n)} + \frac{\alpha^{(n)}}{2}\right) + j^{-(n)}\left(\theta^{(n)} - \frac{\alpha^{(n)}}{2}\right) \quad n = k \tag{A.10}$$

here, $\theta^{(n)}$ is the drum rotation angle of drum n and $\alpha^{(n)}$ is the coating angle of drum n. In the case where $n \neq k$:

$$\frac{d}{d\theta^{(k)}}\left[\gamma^{(n)}\right] = 0 \quad n \neq k \tag{A.11}$$

For $W^{(k)}$,

$$W^{(k)} = \frac{d}{d\theta^{(k)}}\left[\int_{\Theta_A^{(k)}\cap\Theta_{A_0'}^{(k)}} j^{-(k)}(\beta)^2 d\beta - \int_{\Theta_{A'}^{(k)}\cap\Theta_{A_0}^{(k)}} j^{-(k)}(\beta)^2 d\beta\right] \tag{A.12}$$

$$W^{(k)} = j^{-(k)}\left(\theta^{(k)} + \alpha^{(k)}/2\right)^2 - j^{-(k)}\left(\theta^{(k)} - \alpha^{(k)}/2\right)^2 \tag{A.13}$$