DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof. Reference herein to any social initiative (including but not limited to Diversity, Equity, and Inclusion (DEI); Community Benefits Plans (CBP); Justice 40; etc.) is made by the Author independent of any current requirement by the United States Government and does not constitute or imply endorsement, recommendation, or support by the United States Government or any agency thereof.

LA-UR-25-29285

Approved for public release; distribution is unlimited.

Title: MetaHeuristic Feature Selection for Energy Group Optimization and Analysis

Author(s): Rouse, Natalie Kealaula
Whewell, Benjamin Joseph
Gibson, Nathan Andrew

Intended for: Report

Issued: 2025-09-16

MHFS

MetaHeuristic Feature Selection for Energy Group Optimization and Analysis

Natalie Rouse
Mentors: Nathan Gibson, Benjamin Whewell
Los Alamos National Laboratory

August 5, 2025

Contents

Abstract

Energy discretization is a crucial component of deterministic neutron transport simulations. Metaheuristic (MH) optimizers are effective algorithms to determine group structures that maximize both solution accuracy and computational efficiency. This project establishes a framework for optimizing group structures for PARTISN simulations using the Python library MEALPY. Group structure optimization is formulated as a binary feature selection problem, and results are investigated with permutation and material importance techniques to determine physically relevant energy bounds. We conclude that MH optimizers find group structures that drastically improve flux calculations while preserving k-effective accuracy. Further, we find that individual energy bounds are not necessarily physically relevant, but rather specific energy ranges are.

1 Introduction

Deterministic neutron transport simulations, like PARTISN, require discretization in both energy and angle. Energy discretizations involve grouping cross-sections by their incident neutron energy. Fine group structures increase accuracy, but decrease computational efficiency. Coarse group structures increase efficiency, but are often far less accurate. However, more groups does not necessarily equate to greater accuracy; solution accuracy relies heavily on boundary placement.

Previous work on this topic explores finding effective group structures using various optimization techniques. Optimizers researched in the past include greedy algorithms like hierarchical agglomeration and division [1], as well as metaheuristic (MH) algorithms like simulated annealing coupled with machine learning techniques [6]. This project expands upon MH applications by widening access to over one hundred MH algorithms with a single implementation to generate group structures for general configurations of the user's choice. This is done by formulating group structure optimization as a feature selection problem and utilizing the MH Python library MEALPY [8] to solve it. The user is then able to tailor the optimization process to the problem they are solving.

This implementation casts group structure optimization as a binary feature selection problem for simplicity and compatibility with canned optimizers. The LANL618 group structure is considered a "close-enough" approximation to continuous energy simulations. Therefore, the formulation of this optimization problem is a feature selection of the LANL618 structure. Essentially, the goal is to minimize error between the LANL618 and a 15-22 group subset of the LANL618. Ideally, this yields a highly accurate low-group structure, compared to the LANL30.

Metaheuristics are used to solve this feature selection problem because it is a class of semi-stochastic, approximate algorithms. These characteristics of MH ease much of the computational burden associated with a combinatorial problem like feature selection. As described in Section 2, the MH algorithms employ both exploration (global search) and exploitation (local search) techniques, and the balance of these is critical to success in "solving" the group structure optimization problem. Finally, MH optimization is problem-independent, so this project can theoretically optimize group structure for any type of transport simulations, though this implementation was only tested on 1-dimensional metal fast configurations.

Analyzing optimization results also allowed us to investigate the physical reasoning behind energy bound importance. This is important because LANL's standard structures

work well for a wide range of configurations, but lack physical reasoning as to why. So, we also conducted feature importance studies with permutation importance simulations and material-specific optimization runs to further explore impactful energy bounds and ranges.

2 Methodology

2.1 Formulation as Binary Feature Selection

Group structure optimization can be formulated as binary feature selection when searching for a subset of another structure. The LANL618 is used as the reference structure because it is almost as accurate as continuous energy. In context, LANL618's bounds are considered the features, and its solutions are considered ground truth for error calculations. The optimized structures can then be viewed as the solution to the feature selection problem, where its energy bounds are a subset of the LANL618 bounds. For example, Figure 1 shows which features are selected for the LANL30 from the LANL618.

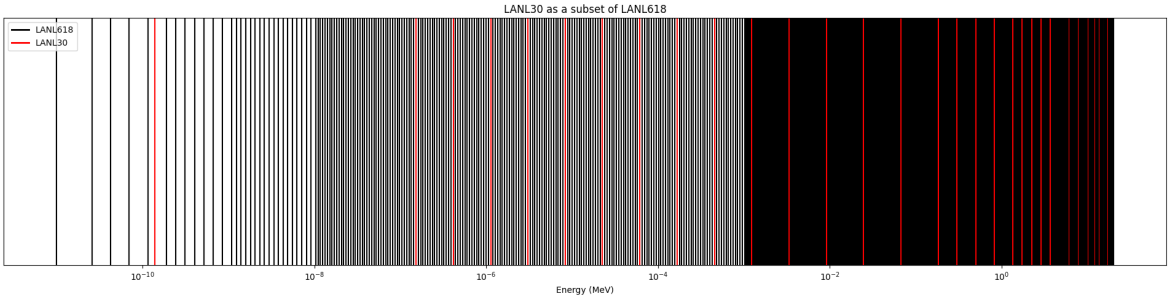

Figure 1: Features Selected from LANL618 in LANL30

In this implementation, each structure in the optimization process is considered a solution and represented by a 619-element binary vector, one element for each possible group boundary. The red in Figure 1 indicates "active" features, or boundaries, and thus the binary vector is referred to as the position of a structure, where ones are active bounds. Using binary vectors as solutions prevents round-off error in actual energy values while also forcing solutions to stay in a predefined search space, the LANL618. Furthermore, the binary implementation lends itself to compatibility with binary MH algorithms.

2.2 Implementation in MEALPY

Feature selection is combinatorial in nature, making it highly complex, or NP-hard (computational expensive). MH algorithms search both locally and globally, but not exhaustively, which makes them advantageous for this type of problem because their stochastic and approximate characteristics reduce computational complexity [5]. The balance between an MH algorithm's global and local search techniques is critical to optimization success. Too much exploration leads to computational inefficiency, while excessive exploitation causes the algorithm to get stuck in local minima.

Evaluating the entire LANL618 search space for the best 20 group subset tailored to a *single* test problem is equivalent to $\binom{619}{20} \approx 2 \times 10^{37}$ PARTISN evaluations. In contrast, using a metaheuristic optimizer to find an approximate best structure for 40

different problems with a standard 30 epochs and 50 agents takes only $40 \times 30 \times 50 = 6000$ PARTISN evaluations. For reference, an agent is a single candidate solution for the optimization problem and represents one possible position in the search space. As such, one agent equates to 40 PARTISN evaluations, since we are using a 40-component objective function. An epoch is one complete iteration of the optimizer, where all agents (50 in this case) are evaluated and updated based on the optimizer's specific algorithm. The minimal number of PARTISN evaluations make using metaheuristics a logical choice for determining generalized, accurate, low-group structures.

MEALPY optimizers are not developed specifically for binary feature selection, so a few adjustments must be made to the library's implementation. In this project, we created a custom child class of MEALPY's problem class to read and edit PARTISN input files, as well as to utilize transfer binary variables and a transfer function. `TransferBinaryVar` is a MEALPY-specific datatype built for networks and feature selection problems. This datatype performed the best out of MEALPY's other two applicable options, `BinaryVar` and `PermutationVar`. Furthermore, the use of a transfer function is necessary to map the continuous search space back to discrete solutions. We chose the vstf_04 transfer function based on a particle swarm study published in the Swarm and Evolutionary Computation journal [3]. The combination of the `TransferBinaryVar` and a transfer function allows almost any MEALPY algorithm to run on the binary feature selection problem we have formulated. Thus, over one hundred MH algorithms are now easily available for use in group structure optimization. In addition to transfer binary variables and transfer functions, this implementation is highly customizable by the user with the parameters shown in Table 1.

Parameter	**Description**	**Chosen Value**
Training Set	Problems optimized in objective function	40 metal fast configurations (test 001 of each experiment)
Testing Set	Problems used to validate accuracy	84 metal fast configurations (training set plus all experimental perturbations: 002, 003...)
Optimizer	MEALPY algorithm	BBOA, AEO, TWO, etc.
Run Length	Epochs, agents	30, 50
Transfer Function	Continuous to discrete search space mapper	vstf_04
Objective Function Weights	For each training problem	All ones
Comparison Group Structure	To validate accuracy with test set	LANL30, LANL70
Fast/Thermal Boundary Delineation	For error calculations	0.823 MeV
Group Number Penalty	Function defined to penalize group structures outside a given range	(15, 22]

Table 1: User-Defined Parameters

The same objective function and error calculations were used across all optimization trials. The group penalty in the fitness function either penalized groups above 25 or 22,

depending on the trial. Error calculations, as shown in Equations 2 and 3, include scalar flux, reaction rate, and k-effective errors, all split between fast and thermal energy zones at 0.823 MeV. These metrics are calculated, aggregated, and scaled for each test problem, then summed over all test problems. This is a simple sum, assuming all objective function weights from Table 1 are set equal to one. This makes up Equation 1, the fitness function (test problem summation not shown).

Fitness Function

$$\text{minimize } p + \sqrt{(\frac{1000}{3} e_{keff})^2 + \sum_{i,j} (e_{ij})^2} \tag{1}$$

$$\text{where } e_{keff} = \left| \frac{e_{keff}^{ref} - e_{keff}^{approx}}{e_{keff}^{ref}} \right| \tag{2}$$

$$\text{and } e_{ij} = \left| \frac{\overrightarrow{\Delta_i^{\text{ref}}} \cdot \overrightarrow{e_{ij}^{ref}} - \overrightarrow{\Delta_i^{\text{approx}}} \cdot \overrightarrow{e_{ij}^{approx}}}{\overrightarrow{\Delta_i^{\text{ref}}} \cdot \overrightarrow{e_{ij}^{ref}}} \right| \tag{3}$$

$$\forall i \in I, j \in J$$

Sets

$i \in I$: energy zones, $I = \{\text{fast, thermal}\}$

$j \in J$: error metrics, $J = \{\text{absorption reaction rate, neutron production reaction rate, total reaction rate, scalar flux}\}$

Parameters

$\overrightarrow{\Delta_i^{\text{ref}}}$ = reference structure's energy bin widths in the i^{th} energy zone

$\overrightarrow{\Delta_i^{\text{approx}}}$ = approximate structure's energy bin widths in the i^{th} energy zone

e_{ij} = total error for the j^{th} metric in the i^{th} energy zone

$\overrightarrow{e_{ij}^{ref}}$ = reference structure's j^{th} metric value for each energy bin in zone i

$\overrightarrow{e_{ij}^{approx}}$ = approximate structure's j^{th} metric value for each energy bin in zone i

p = group number penalty

3 Results

Initially, the goal of this project was to find a general 20-group structure that outperforms LANL30 and establish a new benchmark group structure, LANL20. However, the optimization results revealed that very different group structures often yield comparable errors. Therefore, finding a generalized LANL20 would require far more extensive training and testing sets to determine the *overall best* structure for a wide range of problems. Despite not establishing a LANL20, five optimization trials of roughly eight algorithms produced 36 group structures that are at least 60% as accurate as LANL30. Of these, 22 have $\geq$ 90% accuracy, 15 have $\geq$ 95% accuracy, and four have 100% accuracy. All optimized structures included 40 1-dimensional metal fast configurations in the objective function (training set) and were tested over 84 1-dimensional critical sphere metal

fast configurations. Accuracy with respect to LANL30 is calculated as the percentage of test problems where the optimized structure produces a lower overall error compared to LANL30. The four 100% accurate structures are explored in Section 3.1 and their algorithms detailed in Section 3.2.

3.1 Best Structures

The best structures were determined by their accuracy compared to both LANL30 and LANL70. The four 100% accurate structures are shown in Table 2, but only the structures that used 30 epochs will be shown in the majority of the following plots for consistency and readability. The medium lengths were varied to included subcritical $k_{\text{eff}} \approx 0.9$ and supercritical $k_{\text{eff}} \approx 1.1$ versions of the critical test problems, otherwise referred to as "perturbed problems". All structures still outperformed LANL30 on the perturbed problems, indicating structure robustness.

run group	epochs	agents	group penalty	algorithm	# groups	LANL 30	LANL 70	LANL 30 (%)	LANL 70 (%)
2	30	50	25	AEO	22	84	10	100%	11.90%
2	30	50	25	BBOA	24	84	3	100%	3.57%
5	25	60	22	AEO	22	84	1	100%	1.19%
3	30	50	22	TWO	20	84	0	100%	0.00%

Table 2: Optimized Structures that Perform 100% Better than LANL30 on Test Set

Comparing error between optimized structures and LANL30 reveals that optimized structures are far more accurate in calculating scalar flux and moderately more accurate in calculating k-effective. Reaction rate errors were consistent across structures, so they were not plotted in Figure 2. Flux error is the most visually obvious improvement, where all optimized structures have a flux error ≤ 0.4, compared to LANL30's ≥ 0.6 flux error. This highlights the strength of optimized structures in flux calculations, likely tied to the more effective bound placement in the resonance region. LANL30 bounds often split resonances, which causes LANL30 bounds to also intersect on flux dips in the same region. Optimized structures avoid these intersections, and thus improve upon LANL30's overall error even while utilizing less groups.

AEO (Row 1, Table 2) outperforms LANL30 and all other structures on almost every test problem. This structure uses only 22 groups, but is able to calculate k-effective more accurately even on problems where LANL30 produces a high k-effective error, like PU-MET-FAST-027-001 and HEU-MET-FAST-022-001 (Figure 2). This establishes confidence in using MH algorithms to determine effective group structures. AEO, despite using 8 less groups than LANL30, performs better on several error metrics, not just a single one like k-effective.

Since group structures discretize cross-section by incident neutron energy, Figure 3 is an intuitive way to visualize the differences between LANL30 and optimized structures. Cross-sections for Plutonium-239, Uranium-235, Beryllium-9, and Iron-56 are shown on a log-log scale with vertical dotted lines representing group boundaries. LANL30 utilizes a log-spacing scheme, while the optimized structures appear to have more specialized bound placement. LANL30 boundaries intersect big resonances, but optimized structures generally avoid this, as you can see in the Be and Fe cross-sections. Further, optimized

structures have far less bounds in the ^{235}U and ^{239}Pu resonance regions, but better error. This points to either more physically relevant placement, or more cancellation of error when compared to LANL30.

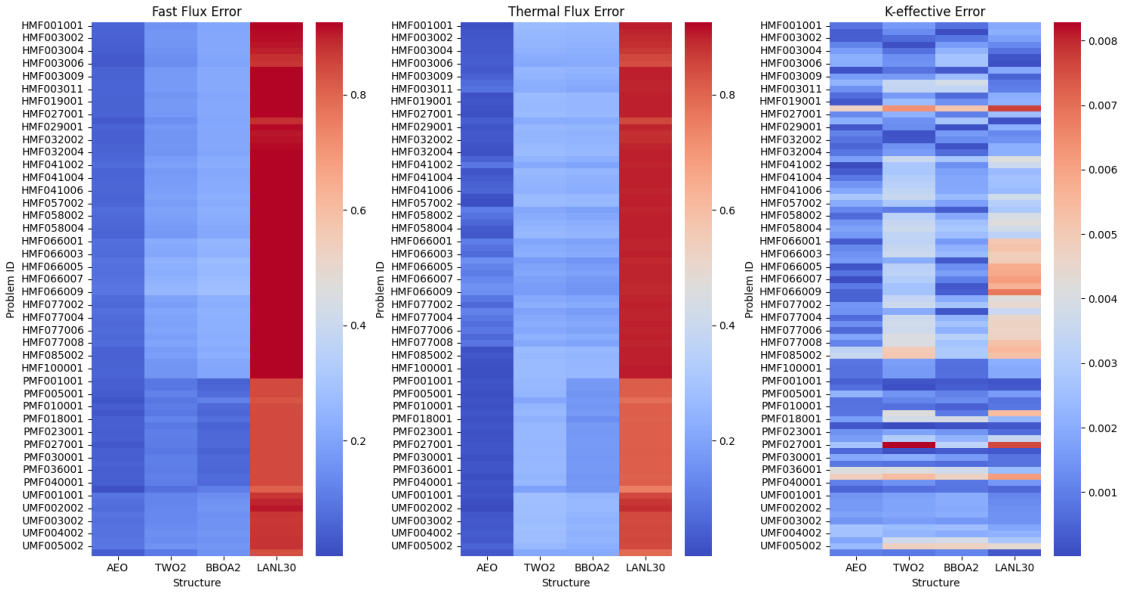

Figure 2: Fast and Thermal Scalar Flux Error by Test Problem and Group Structure

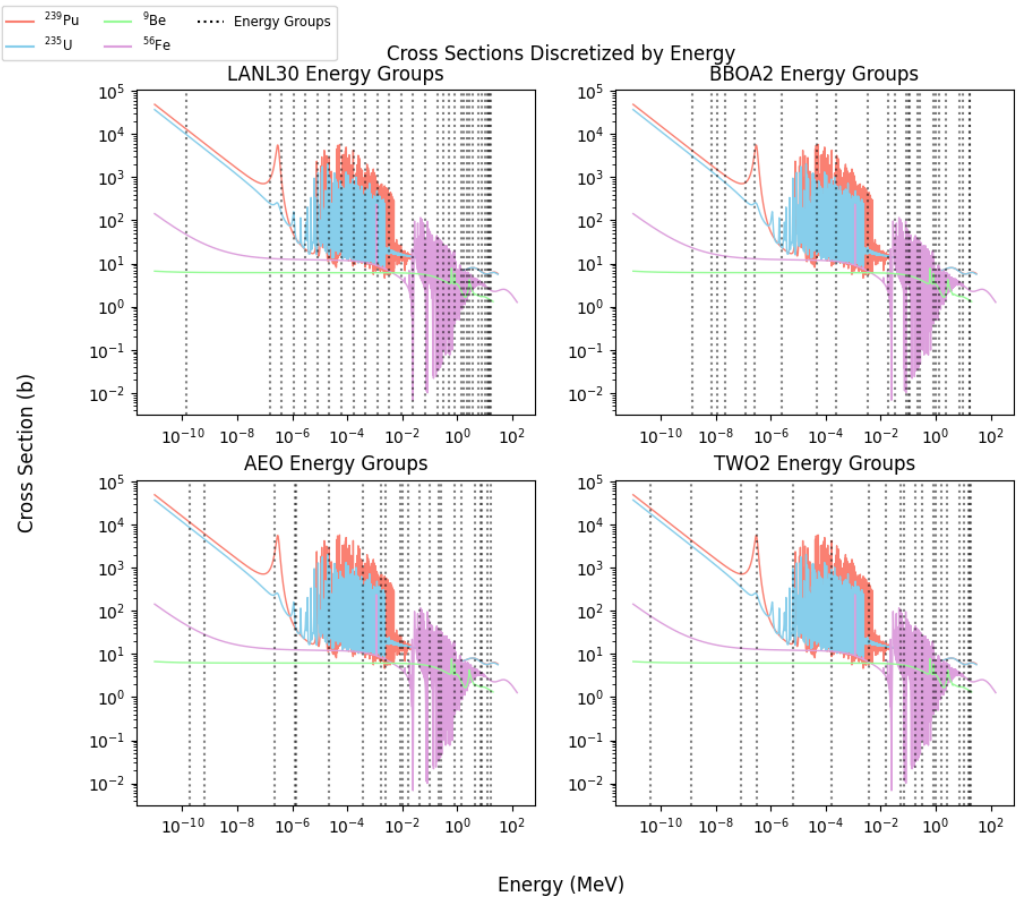

Figure 3: Best Structure to Cross-Section Comparison with LANL30.

3.2 Best Algorithms

The best algorithms were the optimizers that were consistently successful in finding a competitive structure. Augmented Ecosystem Optimizer (AEO), Brown-Bear Optimization Algorithm (BBOA), and Tug-of-War Optimizer (TWO) all found a structure that is 100% as accurate as LANL30. AEO mimics energy flow across Earth through living organisms, utilizing three main processes, production, consumption, and decomposition [7]. BBOA is inspired by the communication practices of brown bears, namely pedal scent marking and sniffing behaviors [4]. Finally, TWO is the most self-explanatory, given it is based on a game of tug-of-war between multiple opposing teams, or agents, and positions are updated according to Newtonian laws of mechanics [2].

Aside from the above algorithms, the Flower Pollination, Particle Swarm, and Grey-Wolfe Whale Optimizers also consistently produced structures with $\geq$ 90% accuracy. Most of these minimize the number of parameters necessary, with only two requiring parameters. These parameters are highly algorithm-specific and generally control exploration/exploitation and the position-update process. We found that parameter-free optimization algorithms balance exploration and exploitation the most reliably because they are general algorithms and do not require parameter tuning. This allows for efficient convergence to a near-optimal solution. Fitness for the best structures converge efficiently to roughly the same value for any optimizer, shown in Figure 4.

In contrast, using algorithms like Simulated Annealing (SA), Differential Evolution (DE), and Harmony Search (HS) with their default parameters caused them to get stuck in local minima. For example, SA got stuck testing structures with 1-2 groups, while DE tested only structures with 100-200 groups, and HS tested structures around 500 groups. This increases the computational burden because the algorithms run PARTISN on finer group structures, which are inefficient to simulate. Testing fine group structures in late epochs despite the group number penalty implies that exploration remains high for too long, so these algorithms would require specialized parameter-tuning.

Even some algorithms that have success, like TWO, are inefficient due to their exploration balance. Figure 5 shows how BBOA and AEO start with high exploration and quickly switch to high exploitation. On the other hand, TWO stochastically alternates between increasing and decreasing exploration and exploitation, though the former always stays higher than the latter. For this reason, TWO takes far longer to converge (complete all 30 epochs) compared to other algorithms because it spends a lot of computation time exploring structures with a high number of groups.

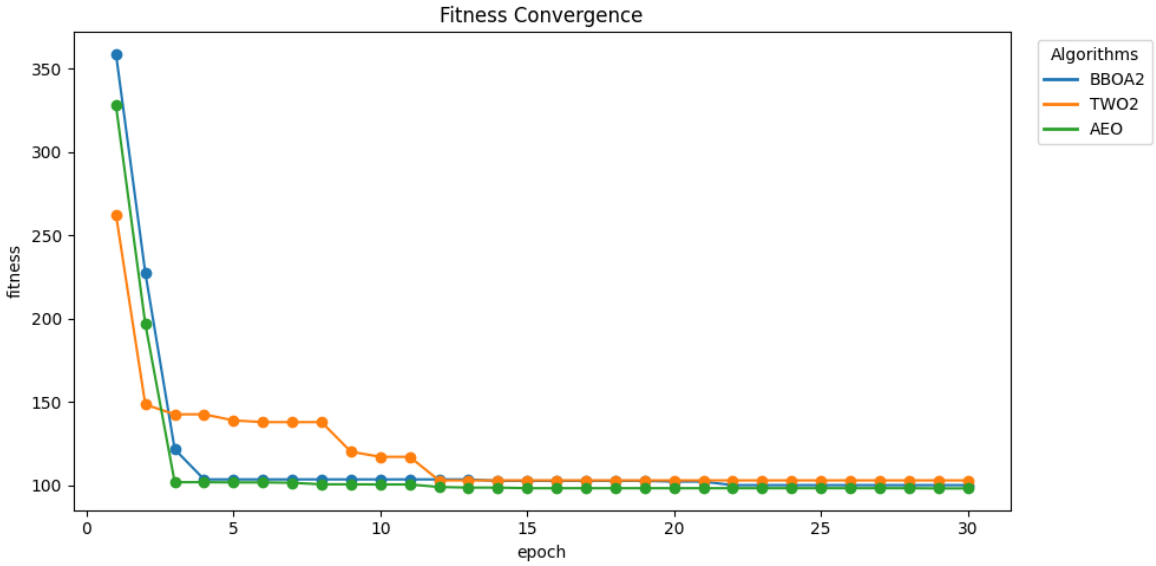

Figure 4: Fitness Convergence of Top 3 Structures

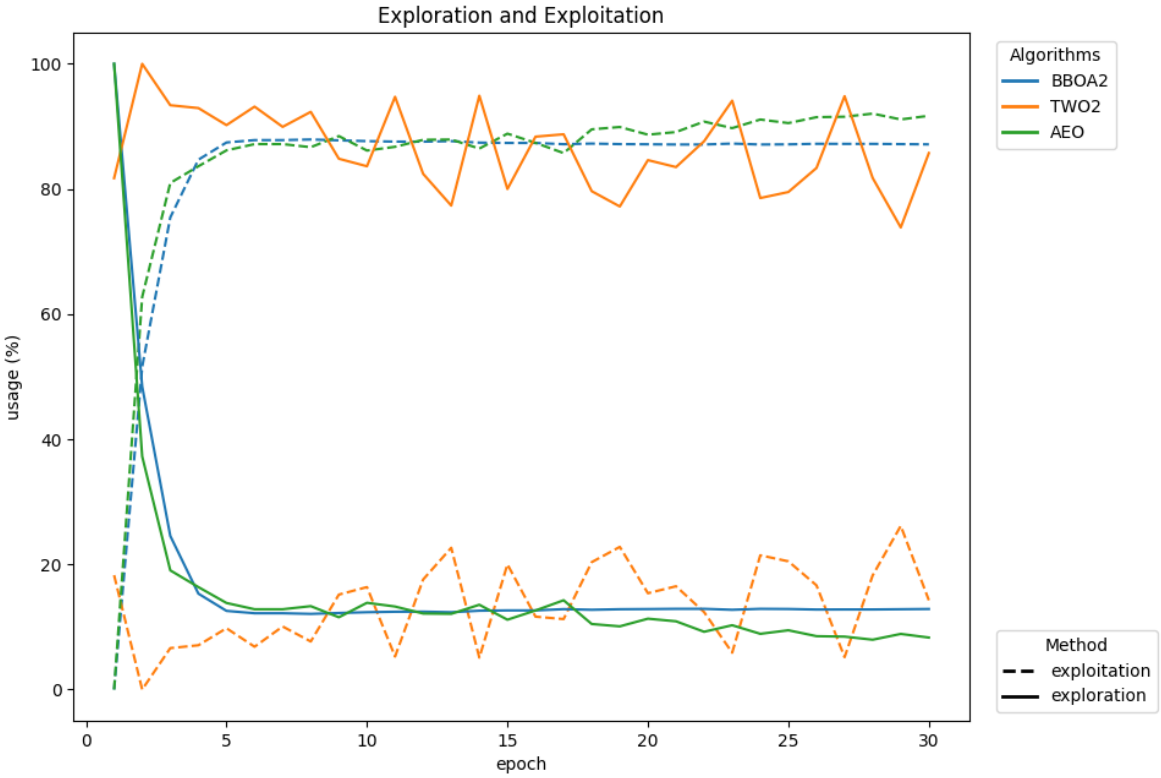

Figure 5: Exploration vs. Exploitation for Top 3 Structures

4 Analysis

The next segment of this project focuses on feature importance and determining physical reasoning behind the effectiveness of certain energy bounds. We conduct permutation

importance simulations as well as material-specific optimization runs to identify patterns in results and structure behavior.

4.1 Permutation Importance

Permutation importance is a technique for determining feature importance by holding the model, or structure in this case, constant, and changing one feature at a time to calculate the impact of that permutation on the model performance. Permutation importance was measured for both adding and removing features (bounds) one at a time. Impact is calculated as permutation error (new structure) subtracted from baseline error (original structure). So, positive impacts indicate better performance because permutation error is lower than the baseline. Negative impacts indicate worse performance because permutation error is higher than the baseline. Permutation impacts are calculated for each structure, each *relevant* boundary (Sections 4.1.1 and 4.1.2 explain further), and each test problem. Therefore, there are 84 data points for each relevant bound. Boxplots visualize permutation impacts without averaging these data points while also highlighting outlier test problems. Each bound removal and bound addition boxplot shows the spread of permutation impacts for each relevant bound and colors outliers by test problem name. Visualizations depict LANL30 permutation impacts because this is a standardized structure.

4.1.1 Bound Removal

Bound removal impacts calculate the permutation impact of removing one bound from the structure being analyzed at a time. In detail, the impact at position 381 in Figure 6 was found by calculating the error between LANL30 and LANL618, removing the energy bound at 0.184 MeV, recalculating the error between the permuted structure and LANL618, and subtracting the permuted error from the original error. This process was repeated for each bound in the LANL30. For each structure, between 21-31 impacts are calculated because that is the maximum number of removable bounds. Impact plots for each 100% accurate optimized structure were created and analyzed but omitted for brevity.

The two figures below show the bound removal and perturbation impacts for LANL30. Figure 6 depicts five nonzero bound removal impacts, with negative impacts at 2.48e-02, 6.76e-02, 1.84e-01, and 3.03e-01 MeV and a positive impact at 5.00e-01 MeV. So, removing the bound at 5.00e-01 MeV actually decreases the error and improves the solution for the majority of test problems.

To test whether individual bounds are impactful, we conducted a perturbation of the structure and then retested permutation impact. This is visualized in Figure 7. The blue boxplots show the original permutation impacts. Boxplots clustered around zero (in orange and purple) show the impact of shifting one of the five impactful bounds to the left and right, respectively. This illustrates that small perturbations side-to-side do not effect simulation accuracy. The green and pink boxplots show the impact of then removing the shifted bound, whose data align with the original permutation impact. This proves that permutation impact depends on the energy range, not the individual bound. So, the underlying physics likely depend on energy ranges, not specific LANL618 boundaries. Therefore, feature importance analysis will rely on sets of features, not single features.

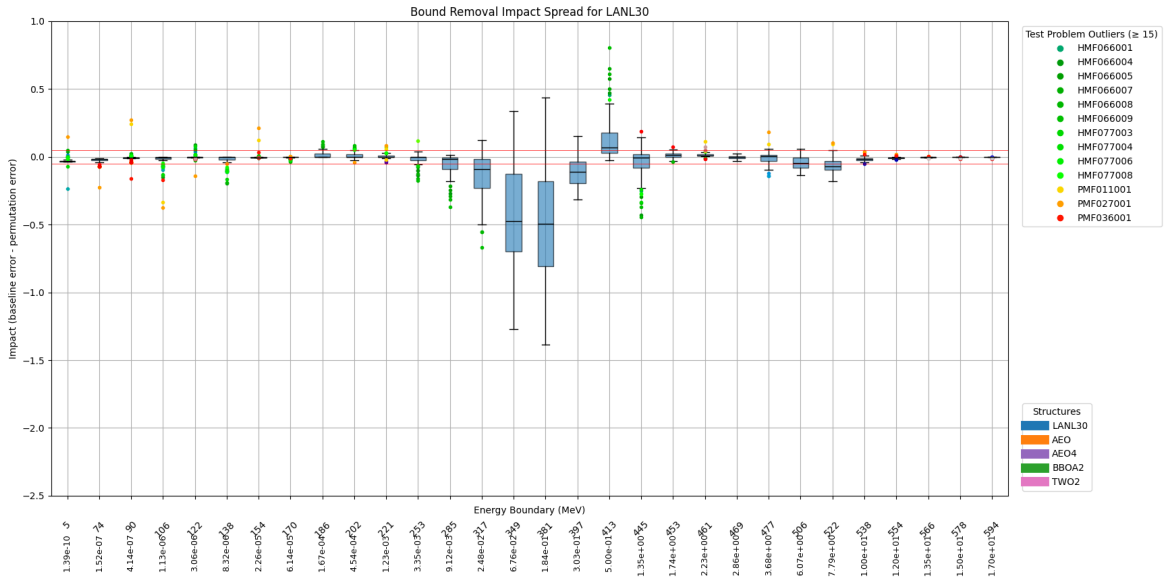

Figure 6: LANL30 Bound Removal Impacts

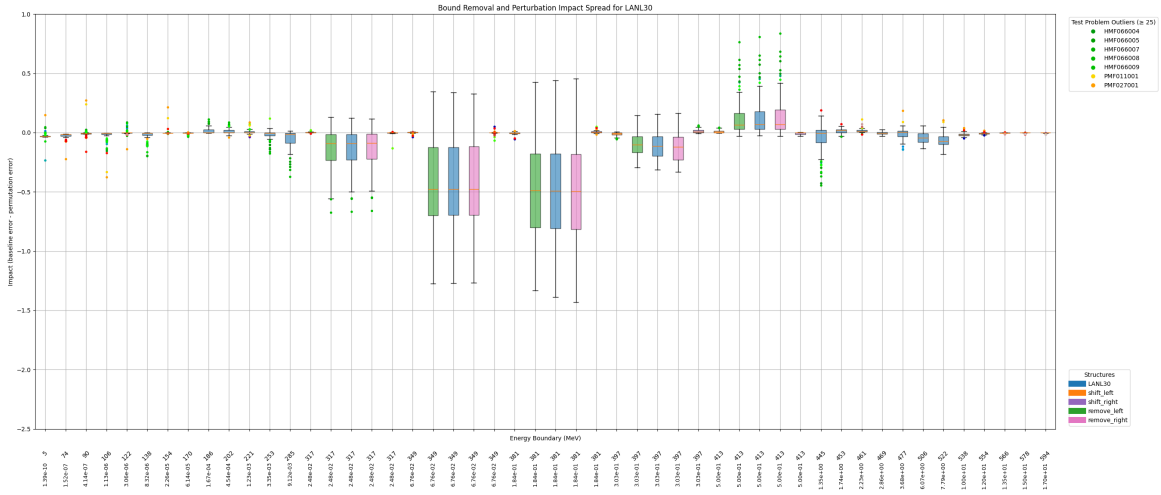

Figure 7: LANL30 Bound Removal Impact Perturbation

4.1.2 Bound Addition

Bound addition impacts are the reverse of bound removal. Since all structures in this project are subsets of the LANL618, bound addition impacts are calculated by adding one bound at a time (from LANL618) where the structure does not already have one. The bound addition boxplots, like Figure 8, display far more data than the bound removal plots because there are more bounds available to add than remove. Trends are easier to visualize with the larger amount of data, and the importance of energy ranges as opposed to individual bound becomes even more clear. The vertical gray lines depict the structure's original bounds, and we see impact peaks around the midpoint between adjacent bounds. This is because adding bounds far from existing bounds yields more information than adding bounds directly next to existing bounds.

Again, bound addition impacts were calculated, analyzed and visualized for all 100% accurate structures, but plots were omitted for brevity. From there, we determined

impactful energy ranges by recording if a bound's interquartile range is entirely above or below zero and the whiskers extend beyond ±0.05. Energy ranges were only calculated for bound addition impacts, since there is significantly less data available for bound removal. Figure 9 depicts impactful ranges for each structure, with green indicating positive impacts and red indicating negative impacts. The cross-section overlay tells us that adding bounds in the uranium and plutonium resonance regions is harmful for two out of five structures, but adding bounds in the iron and beryllium resonance regions is harmful for all five structures.

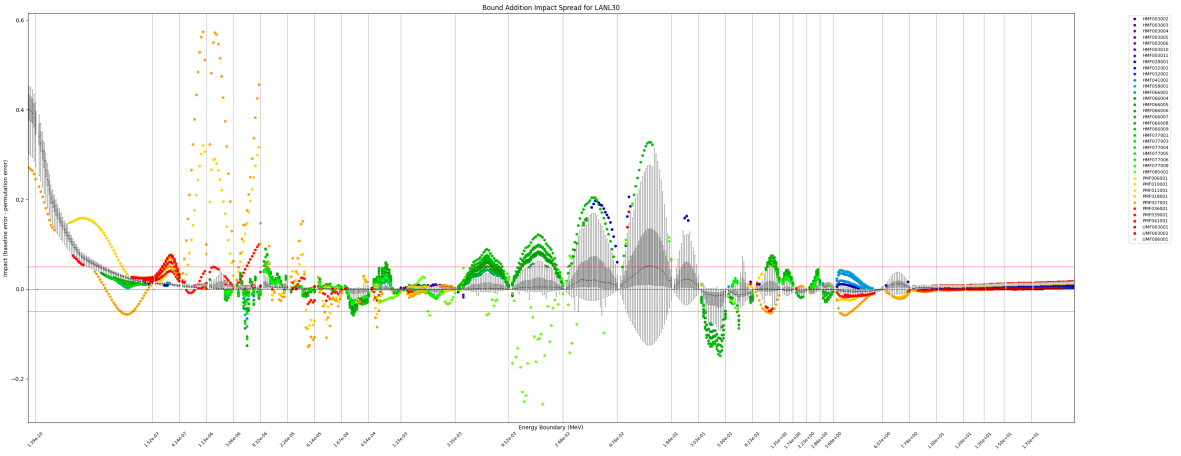

Figure 8: LANL30 Bound Addition Impacts

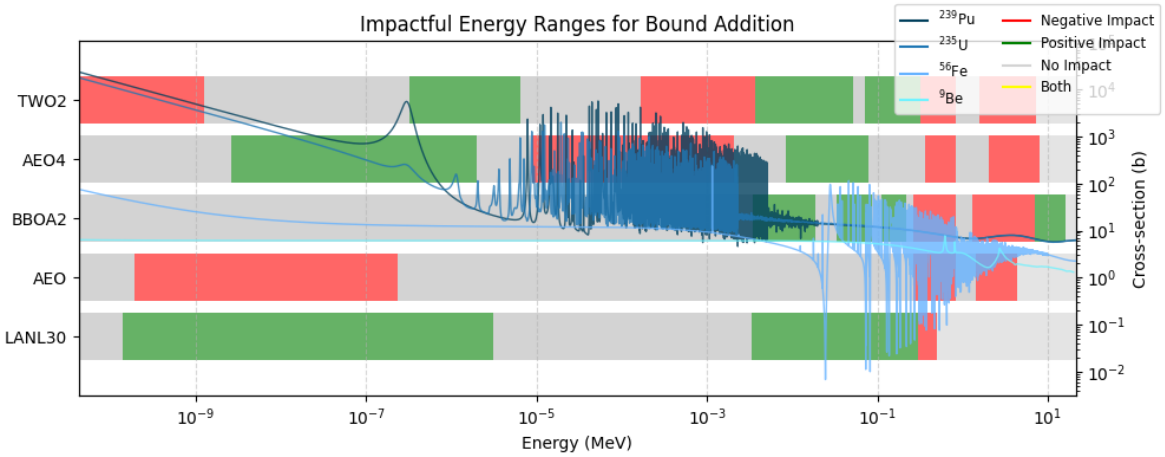

Figure 9: Bound Addition Impact Ranges Relative to Cross-Sections

4.2 Material Importance

The outliers in Figures 6 and 8 highlight the abnormal behavior of certain test problems. The bound addition impacts plot clearly visualizes how some plutonium problems behave abnormally at low energies and some uranium problems behave abnormally in the middle of the shown energy range. Since only a subset of problems associated with these fuel types are outliers, further investigation into their underlying materials was necessary.

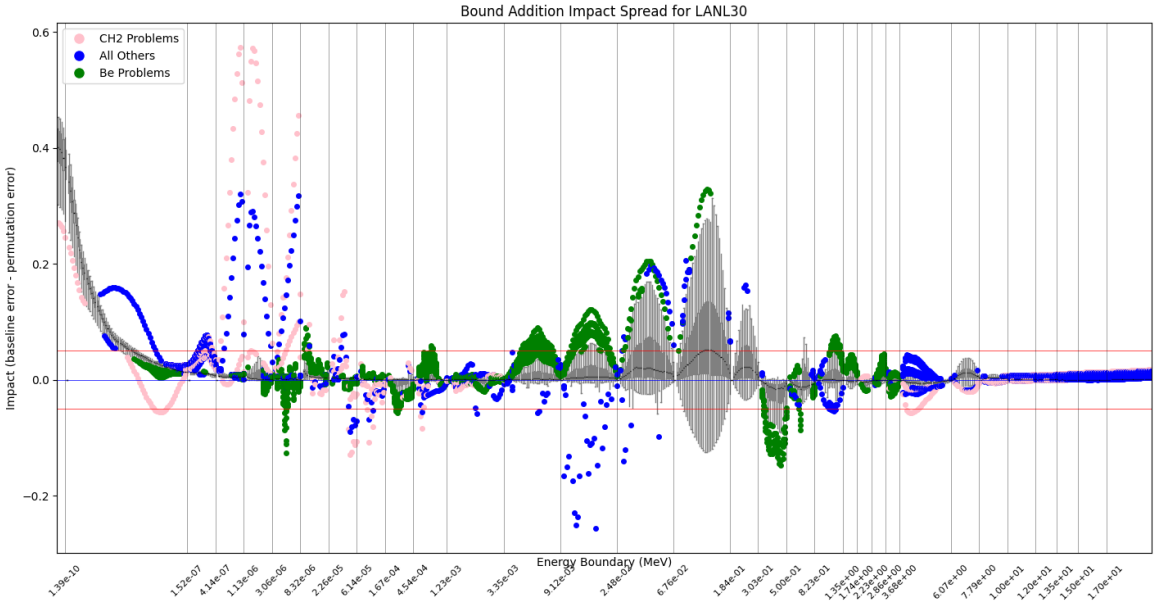

Figure 10: Material-Specific Outliers

Figure 10 color codes the outliers from Figure 8 by problems containing beryllium or high density polyethylene (HDPE). Iron outliers were not colored because there is considerable overlap between problems containing beryllium and iron. Further, each structure's outliers were aggregated and used to calculate the overall and structure-specific percentage of outliers that contained beryllium, poly, and iron, as shown in Table 3. The results showed that about half of the configurations with unusual permutation impacts contain beryllium and over a quarter contain iron. As we have already analyzed the iron and beryllium cross sections in relation to the top three structures, the next step is to visualize flux.

	Be %	Fe %	CH_2 %
LANL30	45.71%	26.08%	11.97%
AEO	57.75%	27.07%	8.59%
BBOA2	38.17%	34.44%	13.62%
TWO2	40.13%	28.24%	14.92%
AEO4	54.25%	27.22%	8.26%
All Structures	47.97%	28.66%	11.19%

Table 3: Outlier Percentages by Material and Structure

Figures 11 and 12 show the scalar flux calculated with the LANL618, along with the top three optimized group structures and LANL30. Figure 11 shows all experiments associated with the HEU-MET-FAST-066 configuration, which contains a beryllium reflector. Figure 12 shows all plutonium-fueled configurations that contain iron. The red vertical line indicates the fast/thermal energy range delineation used in error calculations. Similarly to the cross-section plots, we see LANL30 bounds intersecting flux dips, as well as the flux maximum. Optimized structures minimize flux error by utilizing bounds between flux dips, just as they do in the resonance region for cross-sections.

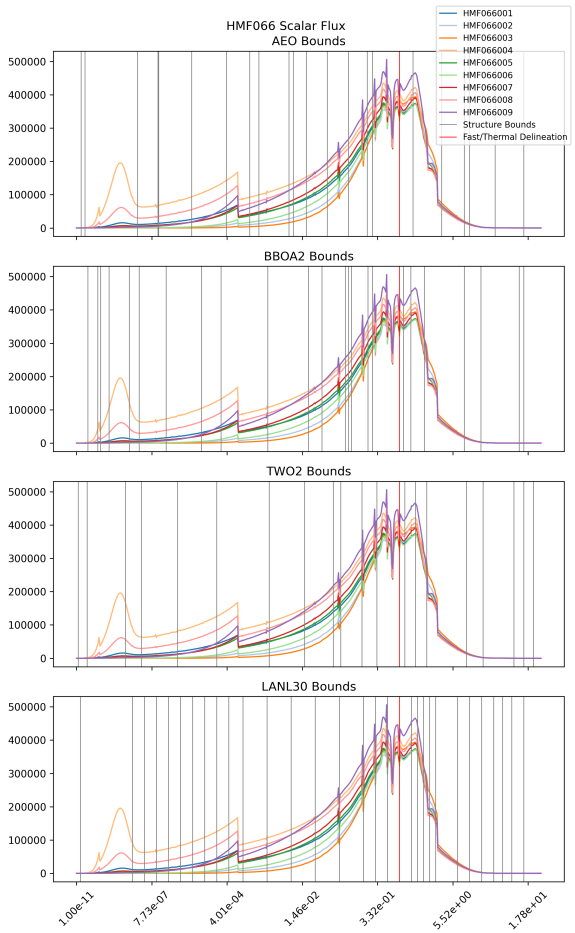

Figure 11: Beryllium Flux with Group Structure Overlay

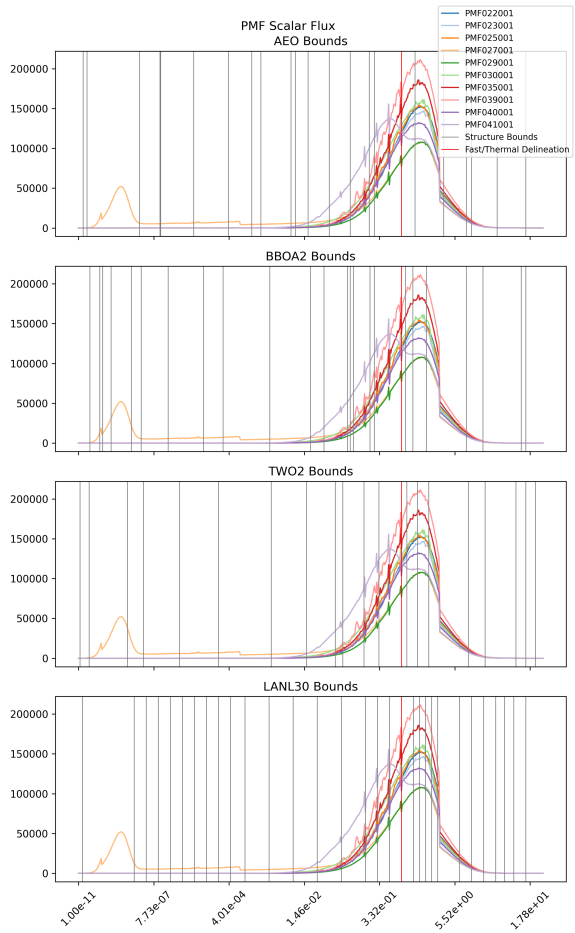

Figure 12: Iron Flux with Group Structure Overlay

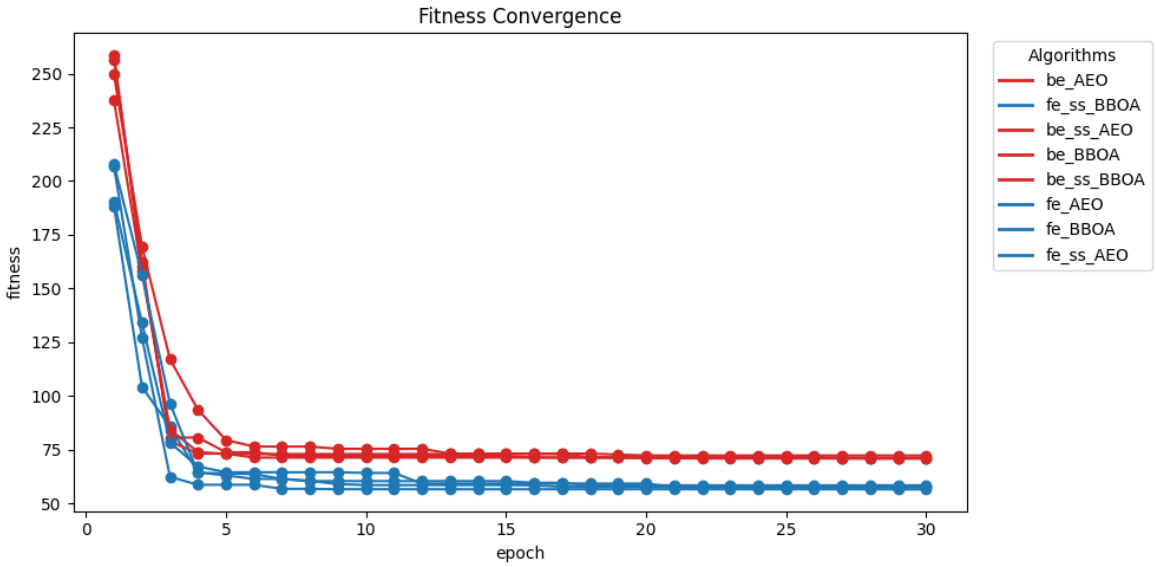

Figure 13: Fitness Convergence Comparison by Material

The final observation we made regarding structural behavior and material differences is a deviation in fitness convergence. Running several different MH algorithms on the same

training problem set often yields comparable best fitness scores, as shown in Figure 4. However, when the same algorithms train on different datasets, one for iron and one for beryllium in this case, we see convergence to different fitness levels. Further, running the algorithms on self-shielded data does not affect fitness convergence or error calculations as we expected it to. Instead, the use of self-shielded data simply reinforced the notion that structures optimized for material have a minimum fitness that differs from other materials. Figure 13 shows that all beryllium-based algorithms converge to roughly 75, while all iron-based algorithms reach a fitness level of 60.

5 Conclusion and Future Work

This project successfully connected energy group structure optimization for PARTISN simulations to the Python library of metaheuristic algorithms, MEALPY. Now, users can optimize group structures for any set of experimental configurations by solving a feature selection problem with the MH algorithm of their choice. This implementation allows the user to explore a variety of MH algorithms, rather than just the most common ones used in past research. The user can also define a set of parameters that change the length, continuous/discrete mapping, objective function weighting, and error calculations of the algorithm, so it is highly customizable to the user's needs.

Five optimization runs of about eight algorithms yielded four out of 36 group structures that outperform LANL30 on all test problems with only 20-24 groups. Using MH optimizers consistently improves scalar flux errors in both the fast and thermal regions, while also keeping the k-effective error sufficiently low. The decrease in error likely stems from more effective bound placement, i.e., minimizing the number of bounds that intersect big resonances and flux dips.

Upon further investigation, we found that individual bounds do not represent important underlying physics, but energy ranges do. We also determined that material composition, specifically the presence of beryllium, iron, or poly, causes problems to behave differently under the same structures. Permutation importance was the statistical technique used to draw these conclusions, given it was developed for binary feature selection problems like the one we are studying here.

This project has several limitations and drawbacks. Many are associated with the time in which the project was completed, as we use limited optimization trials (5 runs of 8 algorithms) and did not finalize efficient parameter-tuning methods. In the future, we could expand the algorithms tested to more parameter-heavy optimizers and spend computational resources tuning them for the best performance. This could yield even more robust results than already achieved.

Other future work could include expanding sample data from the optimization process to then use as inputs to an ML model that has feature importance capabilities. This was an initial approach taken until we realized that there was not sufficient generated data to train an ML model. With a trained ML model, we could analyze feature importance in a more statistically significant way, like SHAP values or accumulated local effects analysis, rather than a simple permutation importance. An ML model would allow for investigation into mulitcollinearity, which would be useful in determining individual bound importance. With permutation importance, we were unable to quantify feature correlations, which could give insight into whether MH algorithms are overfitting to training problems.

Finally, there are several more materials that likely have an effect on structure per-

formance that can be investigated in a similar way to iron and beryllium. Similarly, work on different configurations to find a completely general structure would widen the impact of this work. For example, the addition of solution or thermal configurations and self-shielded data would likely alter results drastically. Overall, the use of MEALPY in group optimization yielded promising preliminary results for continued work in the future.

References

[1] Jessica Berry and Thomas Saller. Hierarchical division and clustering of group structures. Report LA-UR-22-29528, Los Alamos National Laboratory, 2022. Approved for public release; distribution unlimited.

[2] A. Kaveh. *Tug of War Optimization*, pages 451–487. Springer International Publishing, Cham, 2017.

[3] Seyedali Mirjalili and Andrew Lewis. S-shaped versus v-shaped transfer functions for binary particle swarm optimization. *Swarm and Evolutionary Computation*, 9:1–14, 2013.

[4] Tapan Prakash, Praveen Prakash Singh, Vinay Singh, and Sri Singh. *A Novel Brown-bear Optimization Algorithm for Solving Economic Dispatch Problem*, pages 137–164. IEEE River Publishers, 06 2022.

[5] Octavio Ramos-Figueroa, Marcela Quiroz-Castellanos, Efrén Mezura-Montes, and Oliver Schütze. Metaheuristics to solve grouping problems: A review and a case study. *Swarm and Evolutionary Computation*, 53:100643, 2020.

[6] Thomas G. Saller, Vishnu Nair, Andrew Till, and Nathan Gibson. Using a random forest model to choose optimized group structures. *Nuclear Science and Engineering*, 197(8):2117–2135, 2023.

[7] Nguyen Van Thieu, Surajit Deb Barma, To Van Lam, Ozgur Kisi, and Amai Mahesha. Groundwater level modeling using augmented artificial ecosystem optimization. *Journal of Hydrology*, 617:129034, 2023.

[8] Nguyen Van Thieu and Seyedali Mirjalili. Mealpy: An open-source library for latest meta-heuristic algorithms in python. *Journal of Systems Architecture*, 139:102871, 2023.