Repopulation Frequency as a Driving Factor in Bias Correction in Asynchronous Parallel Evolutionary Neural Architecture Search

Abhishek Singh[1], Zimeng Lyu[2], and Travis Desell[1]

[1] Golisano College of Computing and Information Sciences, Rochester Institute of Technology, Rochester, NY 14623, USA
as3485@rit.edu
[2] Kean University, 1000 Morris Ave Union, NJ 07083
zimeng.lyu@kean.edu

Abstract. Asynchronous parallel evolutionary algorithms improve computational efficiency but introduce evaluation-time bias: architectures with shorter evaluation times accumulate disproportionate reproductive opportunity, steering the search toward compact but potentially suboptimal solutions. While correction strategies such as Harada's frequency-based parent selection and SWEET have been proposed, their behavior in island-based neuroevolution systems with periodic repopulation remains uncharacterized. We present a systematic ablation study comparing these bias correction strategies within EXAMM, a distributed asynchronous neuroevolution framework, across six island repopulation frequencies on three real-world time-series datasets: aviation flight recorder data, coal power plant sensor data and wind turbine sensor data. Each of the 24 experimental conditions is evaluated across 10 independent trials with a fixed budget of 10,000 genome evaluations, with statistical significance assessed via the Mann-Whitney U test against a no-repopulation vanilla baseline. Results show that repopulation frequency provides the strongest effect across the evaluated strategies.

Keywords: Asynchronous Evolutionary Algorithms · Neuroevolution · Neural Architecture Search · Time Series Forecasting

1 Introduction

Evolutionary algorithms offer a powerful gradient-free approach to neural architecture search, capable of exploring non-differentiable search spaces that resist traditional optimization [7, 22, 16]. For computationally expensive tasks such as neuroevolution of recurrent architectures for time-series prediction, distributed parallel evaluation is essential for scalability. Asynchronous parallel evolutionary algorithms (APEAs) improve throughput by dispatching new evaluations immediately upon any worker becoming free, eliminating the idle time of synchronous methods [18, 5]. However, this efficiency comes at a cost: architectures with shorter evaluation times re-enter the reproductive pool more frequently,

creating an implicit selection pressure toward fast-evaluating typically smaller and simpler architectures regardless of predictive quality. This *evaluation-time bias* was formally characterized by Scott and De Jong [18, 19], who showed it arises from an interaction between fitness and evaluation-time gradients and can steer search away from high-quality solutions.

Two complementary strategies address this bias. Harada [9] introduced frequency-based parent selection, which restricts reproduction to individuals with the lowest search-frequency counters, enforcing uniform reproductive opportunity. Scott et al. [17] proposed SWEET (*Selection WhilE EvaluaTing*), which expands the parent pool to include individuals currently undergoing evaluation, giving slow-evaluating architectures an earlier chance to propagate. Both methods have been validated in master-worker parallel EA architectures, where most recently Karns and Desell [11] replicated and extended these findings in a single-population setting; however their behavior in island-based neuroevolution where population dynamics are shaped by island topology, migration, and periodic extinction and repopulation has not been systematically studied.

EXA-GP [15] is a graph-based genetic programming system built on EXAMM (Evolutionary eXploration of Augmenting Memory Models), a distributed asynchronous neuroevolution framework with Lamarckian weight inheritance and periodic island extinction and repopulation [6, 13]. Unlike the master-worker settings in which Harada and SWEET were evaluated, EXAMM and EXA-GP's island topology introduces structural population management that interacts with evaluation-time bias in ways prior work does not capture: repopulation periodically resets accumulated selection history, raising the question of whether explicit bias correction provides measurable benefit at all. This paper investigates this interaction through a systematic ablation study. We compare Harada's method, SWEET, and their combination against a vanilla baseline across six island repopulation frequencies within EXA-GP, evaluated on three real-world time-series datasets. Our study addresses the following research questions:

- **RQ1:** Do evaluation-time bias correction strategies improve predictive performance in asynchronous island-based neuroevolution?
- **RQ2:** Does repopulation frequency dominate when bias correction methods produce measurable benefit?
- **RQ3:** How do bias correction strategies interact with island repopulation to influence the structural complexity of evolved architectures?

2 Related Work

2.1 Asynchronous Parallel Evolutionary Algorithms and Evaluation-Time Bias

Parallel evolutionary algorithms (PEAs) reduce computational cost by distributing fitness evaluations across multiple nodes [3, 10]. Asynchronous PEAs (APEAs) improve on synchronous methods by dispatching new offspring immediately upon any evaluation completing, eliminating idle time [5, 23]. However, this efficiency

introduces *evaluation-time bias*: individuals with shorter evaluation times are selected as parents more frequently regardless of fitness, biasing search toward fast-evaluating rather than high-quality solutions. Scott and De Jong [18, 19] formally characterized this phenomenon, showing it arises from interactions between fitness and evaluation-time gradients. Harada subsequently introduced a mathematical framework quantifying its influence on asynchronous evolutionary dynamics [8].

Island-based parallel EAs partition the population into semi-independent subpopulations with periodic migration between islands [21, 4]. Skolicki and De Jong demonstrated that migration interval is the dominant structural parameter governing island model performance, outweighing migration size and operator design – a finding that directly motivates our investigation of repopulation frequency as a dominant factor in island-based neuroevolution. Neuroevolution algorithms such as NEAT [22] and EXAMM [16] extend these principles to neural architecture search, introducing additional evaluation-time heterogeneity as architectures grow in complexity [7].

2.2 Bias Correction Strategies

Harada [9] proposed frequency-based parent selection, which restricts reproduction to individuals with the lowest search-frequency counters, enforcing uniform reproductive opportunity regardless of evaluation time. Scott et al. [17] introduced SWEET (*Selection WhilE EvaluaTing*), which expands the parent pool to include individuals currently under evaluation, giving slow-evaluating architectures an earlier chance to propagate before fitness is known. Karns and Desell [11] most recently replicated and extended these findings in a single-population master-worker setting. Lyu et al. [13] introduced island extinction and repopulation into EXAMM, demonstrating significant improvements over baseline island evolution, but did not investigate its interaction with evaluation-time bias correction the interaction the present work directly characterizes.

3 Methodology

3.1 EXAMM and EXA-GP

EXAMM (Evolutionary eXploration of Augmenting Memory Models) [16] is a distributed asynchronous neuroevolution framework that evolves recurrent neural network architectures for time-series prediction through a series of mutation and crossover operations, following in the tradition of topology-evolving neuroevolution algorithms such as NEAT [22]. EXAMM evolves progressively larger recurrent neural network architectures consisting of recurrent connections, simple neurons, and memory cells (LSTMs, GRUs, Delta-RNNs, MGUs, and UGRNNs), using Lamarckian weight inheritance [12] to significantly reduce the backpropagation required for each generated genome.

In this work we use EXA-GP [15], which converts EXAMM into a graph-based genetic programming algorithm by replacing its recurrent memory cells

with seven differentiable operations (sin, cos, tanh, sigmoid, inverse, sum, and product). EXA-GP has shown improved performance while providing more interpretable and efficient recurrent models.

Because all operations are differentiable, the error signal can be propagated backwards through the entire computational graph, enabling efficient optimization of all trainable weights and bias terms via backpropagation. EXA-GP retains all of EXAMM's distributed infrastructure, island-based population management, Lamarckian weight inheritance, and repopulation mechanisms, differing only in the node library available to the evolutionary process.

3.2 Island Specialization Strategy

EXAMM uses an island-based population model in which the total population is partitioned into K semi-independent islands, each maintaining a fixed-size population of genomes. In this work we use $K = 10$ islands each with population size $N = 10$, consistent with prior EXAMM work [20]. Genome generation rotates across islands in round-robin order, cycling through island indices $0, 1, \ldots, K-1$ and wrapping back to 0 after the last island. For each island, genome generation follows one of three paths depending on the island's current state:

1. **Initializing:** The island is seeded with mutated copies of a starting seed genome and immediately inserted into the population.
2. **Full:** New genomes are generated through a combination of mutation and crossover as described below.
3. **Repopulating:** The island is repopulated with mutated copies of the best genome from the best-performing island following an extinction event.

For fully populated islands, the reproduction operator is selected stochastically. Let $r \sim \mathcal{U}(0, 1)$. If $r < p_{\text{mut}}$, mutation is applied: a genome is selected from the current island and subjected to $n_{\text{mut}} = 3$ randomly chosen mutation operations. If $p_{\text{mut}} \leq r < p_{\text{intra}}$ or only one island is full, intra-island crossover is performed with both parents drawn from within the same island. Otherwise, inter-island crossover is performed with one parent drawn from the current island and a second parent drawn from a randomly chosen other fully populated island.

Island extinction and repopulation events are triggered when the total number of generated genomes g satisfies $g \equiv 0 \pmod{T}$, where T is the repopulation interval. At each trigger, the island with the worst best-genome fitness is entirely replaced and repopulated with mutated copies of the best genome from the best-performing island, resetting its accumulated selection history [13].

3.3 Bias Correction Strategies

We compare four parent selection strategies that vary in how they address evaluation-time bias, applied uniformly to both intra-island and inter-island

crossover. In all crossover operations, parents are ordered so that the genome with fewer enabled connections serves as the primary parent.

Vanilla (Baseline): Standard parent selection with no bias correction. A parent is selected uniformly at random from the fully evaluated genomes $\mathcal{P}$ in the island population:

$$p \sim \mathcal{U}(\mathcal{P}) \tag{1}$$

This preserves default asynchronous selection dynamics and serves as the global baseline for all statistical comparisons.

Harada: The frequency-based parent selection method of Harada [9] is integrated into EXAMM's island-based selection procedure. Each genome g_i maintains a search frequency counter $g_i.\texttt{freq}$, initialized to one at creation for seed genomes, inherited directly from the parent for genomes produced via mutation, or set to the mean frequency of both parents for genomes produced via crossover. When g_i is selected as a parent, its counter is incremented:

$$g_i.\texttt{freq} \leftarrow g_i.\texttt{freq} + 1 \tag{2}$$

Offspring inherit the mean search frequency of their two parents p_1, p_2:

$$g_{\text{child}}.\texttt{freq} = \frac{p_1.\texttt{freq} + p_2.\texttt{freq}}{2} \tag{3}$$

At each parent selection event the island population $\mathcal{P}$ is sorted in ascending order of search frequency, and parents are drawn exclusively from the lowest-frequency candidate pool:

$$\mathcal{P}' = \mathcal{P}_{\text{sorted}}[0 : \max(2,\ \lfloor |\mathcal{P}| \times r_s \rfloor)] \tag{4}$$

where r_s is the selection ratio. We set $r_s = 0.3$, restricting selection to the three least-explored individuals in each island of size 10. This ratio was chosen to account for EXAMM's small island population: at $r_s = 0.5$ only five individuals would form the pool, while $r_s = 0.3$ provides stronger bias correction under these constrained population dynamics.

SWEET: The Selection WhilE EvaluaTing strategy of Scott et al. [17] expands the parent pool to include genomes currently undergoing fitness evaluation. Let $\mathcal{E}$ denote the set of genomes currently being evaluated by worker processes for the island the child is being generated from. Three cases are handled:

$$\text{pool} = \begin{cases} \mathcal{P} & \text{if } |\mathcal{E}| = 0 \\ \mathcal{E} & \text{if } |\mathcal{E}| \geq 2 \\ \mathcal{P} \cup \mathcal{E} & \text{if } |\mathcal{E}| = 1 \end{cases} \tag{5}$$

Parents are drawn uniformly from the selected pool, giving slow-evaluating typically larger architectures an earlier opportunity to propagate their genetic material before fitness evaluation completes.

SWEET+Harada: Both strategies are combined by constructing a unified candidate pool:

$$\mathcal{C} = \mathcal{P} \cup \mathcal{E} \tag{6}$$

This combined pool is sorted in ascending order of search frequency. Unevaluated genomes naturally float toward the front of this ordering alongside infrequently selected evaluated genomes. The Harada selection ratio $r_s = 0.3$ is then applied to this combined pool:

$$\mathcal{C}' = \mathcal{C}_{\text{sorted}}[0 : \max(2,\ \lfloor |\mathcal{C}| \times r_s \rfloor)] \tag{7}$$

Two distinct parents are drawn from $\mathcal{C}'$, both have their search frequency incremented by one, and offspring inherit the mean frequency of their parents. As in SWEET, the parent with fewer enabled connections is the primary parent.

4 Experimental Setup

4.1 Experimental Design

We compare the four parent selection strategies – Vanilla (baseline), Harada, SWEET, and SWEET+Harada – each evaluated under six island repopulation frequency settings $T \in \{\infty, 50, 150, 500, 1000, 2000\}$, yielding $4 \times 6 = 24$ experimental conditions in total. Each condition is run for 10 independent trials with a fixed budget of $G = 10{,}000$ genome evaluations per trial across three real-world time-series datasets: Aviation Flight Recorder, Coal Power Plant, and Wind Turbine [15, 14], using a fixed training sequence length of $L = 50$ time steps.

The six repopulation frequencies were chosen to span the full range of structural intervention: $T = \infty$ preserves default asynchronous island dynamics with no extinction events, $T = 50$ triggers extinction after every 50 genome evaluations across all islands, and intermediate values $T \in \{150, 500, 1000, 2000\}$ characterize the transition between these regimes. This design allows us to isolate the effect of repopulation frequency from the effect of bias correction strategy and to identify the conditions under which each strategy produces measurable benefit.

Two performance metrics are recorded per trial. Predictive performance is measured by best validation mean squared error (MSE) achieved during the evolutionary run, and structural complexity is measured by total enabled connections $W = |\text{Enabled Edges}| + |\text{Enabled Recurrent Edges}|$. All 24 conditions are compared against Vanilla-no-repop as the global baseline via two-sided Mann-Whitney U test ($\alpha = 0.05$). All MSE values reported in Tables 1–2 reflect best validation MSE recorded during the evolutionary run; held-out test evaluation is presented qualitatively for the best-performing genome per dataset in Appendix A.

4.2 Computing Environment

All experiments were conducted on the SPORC high-performance computing cluster at Rochester Institute of Technology [2], using the SLURM workload manager and its tier3 partition. Each experimental condition was executed as a distributed MPI job consisting of one master process and dataset-dependent

worker processes: 22 workers for Aviation, 16 workers for Coal, and 16 workers for Wind. The master process maintains the island population, generates new genomes, and manages repopulation events, while worker processes asynchronously request genomes, perform backpropagation training, and report fitness results back to the master. This asynchronous master-worker architecture ensures that no worker idles waiting for other evaluations to complete, consistent with the APEA design studied in this work.

5 Data Sets

Three real-world multivariate time-series datasets from the EXAMM GitHub repository[3] were used for evaluation, all exhibiting non-seasonal characteristics with interdependent parameter recordings.

Coal-Fired Burner Data (Coal) Sensor recordings from a coal-fired burner system across 12 parameters including airflow, temperature, and combustion measurements. *Supp_ Fuel_ Flow* is the forecasting target.

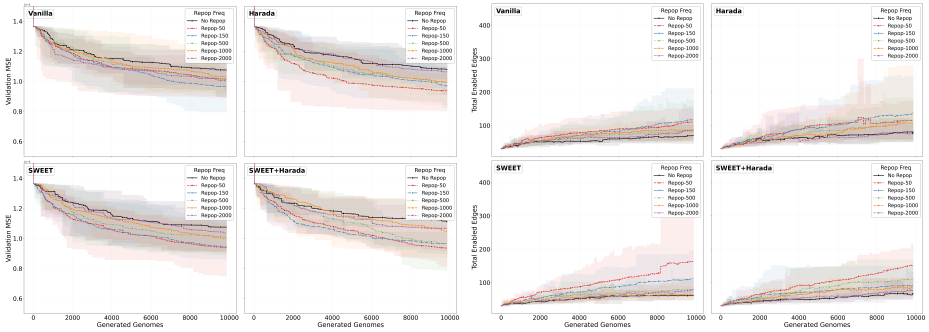
Aviation Flight Recorder Data (Aviation) Flight recordings from Cessna 172 aircraft sourced from the NGAFID [1], comprising per-second data across 31 sensor parameters with identifying information removed. *E1_ CHT1* (Engine 1 Cylinder Head Temperature) is the forecasting target.

Wind Turbine Data (Wind) Ten-minute interval recordings from ENGIE's La Haute Borne windfarm (2017–2020) across 22 parameters. Turbine R80711 was used for training and validation; generalization was evaluated on unseen turbines R80721, R80736, and R80790. *Cm_ avg* (mean torque) is the forecasting target.

6 Results

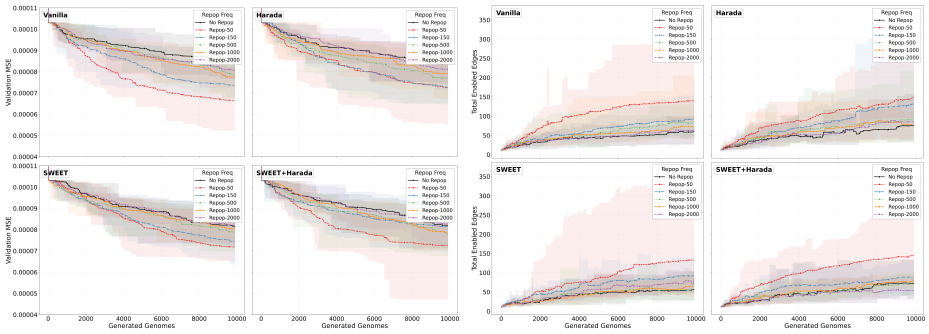
We present results across two primary metrics – validation mean squared error (MSE) and total enabled edges (for structural complexity) – for all 24 experimental conditions on the Aviation Flight Recorder (Aviation), Wind Turbine (Wind), and Coal Power Plant (Coal) datasets. Figures 1-3 show MSE convergence and architecture complexity trajectories across 10 repeats of each experiment. Statistical results are presented in Tables 1-2.

[3] https://github.com/travisdesell/exact/tree/main/datasets



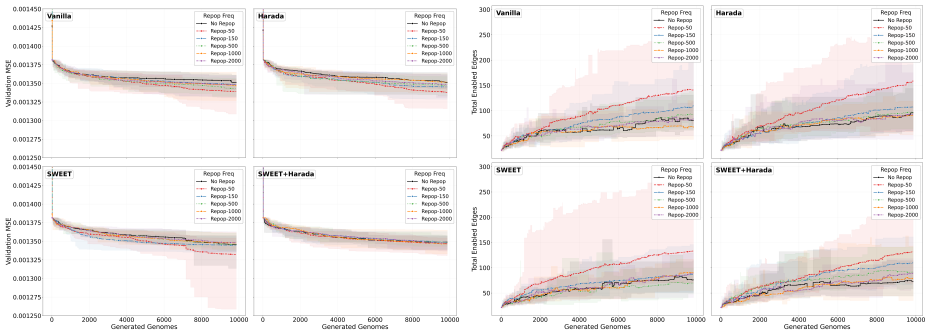
(a) Validation MSE (b) Total Enabled Edges

Fig. 1: Aviation Flight Recorder dataset. MSE convergence (left) and architectural complexity (right) across all four bias correction strategies and six repopulation frequencies. Shaded regions indicate the min-max range across 10 independent trials.



(a) Validation MSE (b) Total Enabled Edges

Fig. 2: Coal Power Plant dataset. MSE convergence (left) and architectural complexity (right). The repopulation frequency effect on MSE is more pronounced than in the Aviation dataset, with Repop-50 achieving substantially lower MSE across all four strategy panels.



(a) Validation MSE (b) Total Enabled Edges

Fig. 3: Wind Turbine dataset. MSE convergence (left) and architectural complexity (right). Differences between repopulation frequencies are more subtle compared to the Aviation and Coal datasets.

6.1 Predictive Performance (MSE)

Repopulation frequency is the dominant factor determining final validation MSE across all three datasets: Repop-50 consistently achieves the lowest average MSE while no-repopulation produces the worst, and no bias correction strategy produces a statistically significant improvement without repopulation (all $p > 0.05$, Table 1). On Coal, Vanilla-repop-50 achieves a 23% MSE reduction over baseline (6.589×10^{-5}, $p = 0.0006$). On Aviation, Harada-repop-50 achieves the best average MSE (9.340×10^{-7}, $p = 0.0046$) with SWEET+Harada-repop-50 the best p-value ($p = 0.0036$), indicating modest additive benefit from combining bias correction with frequent repopulation. On Wind, differences are smaller (range: 1.258-1.365 $\times 10^{-3}$) with SWEET-repop-50 achieving the best results ($p = 0.0257$) but weaker overall significance.

6.2 Architecture Complexity (Enabled Edges)

More frequent repopulation consistently produces larger architectures across all datasets (Table 2). On Coal, Repop-50 increases average edges from 59 to 133-155 ($p < 0.001$), diminishing toward baseline at repop-2000. Aviation follows the same trend with higher variance (SWEET-repop-50 max: 453 edges). Harada produces the largest networks overall while SWEET trends toward more compact architectures, with elevated min-max variance under frequent repopulation reflecting disruption of accumulated selection history at each extinction event.

Table 1: MSE results for all datasets. P-values via Mann-Whitney U test vs V-no-repop (baseline). ↓/↑ = lower/higher than baseline. **Bold** = best in column. V=Vanilla, H=Harada, S=SWEET, SH=SWEET+Harada.

Dataset	Experiment	Min	Avg	Max	Std	p
Aviation	V-no-repop	8.976e-07	1.075e-06	1.198e-06	8.909e-08	1.0000
Aviation	H-no-repop	9.965e-07↑	1.077e-06↑	1.211e-06↑	6.545e-08↓	0.9097
Aviation	S-no-repop	9.120e-07↑	1.075e-06↑	1.247e-06↑	1.130e-07↑	0.9698
Aviation	SH-no-repop	1.053e-06↑	1.110e-06↑	1.220e-06↑	5.634e-08↓	0.4274
Aviation	V-50	8.887e-07↓	1.015e-06↓	1.120e-06↓	7.129e-08↓	0.1212
Aviation	H-50	8.008e-07↓	**9.340e-07**↓	1.098e-06↓	8.640e-08↓	0.0046
Aviation	S-50	**7.413e-07**↓	9.360e-07↓	1.111e-06↓	1.263e-07↑	0.0140
Aviation	SH-50	8.097e-07↓	9.356e-07↓	1.076e-06↓	8.026e-08↓	0.0036
Aviation	V-150	7.885e-07↓	9.645e-07↓	1.202e-06↑	1.342e-07↑	0.0640
Aviation	H-150	8.264e-07↓	9.682e-07↓	1.090e-06↓	8.594e-08↓	0.0173
Aviation	S-150	8.562e-07↓	9.438e-07↓	1.158e-06↓	9.321e-08↑	0.0113
Aviation	SH-150	8.992e-07↑	9.640e-07↓	**1.041e-06**↓	**4.258e-08**↓	0.0073
Aviation	V-500	9.011e-07↑	1.012e-06↓	1.089e-06↓	6.485e-08↓	0.0890
Aviation	H-500	9.010e-07↑	9.918e-07↓	1.212e-06↑	9.281e-08↑	0.0539
Aviation	S-500	8.897e-07↓	1.009e-06↓	1.128e-06↓	8.863e-08↓	0.1212
Aviation	SH-500	7.859e-07↓	9.658e-07↓	1.108e-06↓	8.967e-08↑	0.0211
Aviation	V-1000	9.233e-07↑	1.032e-06↓	1.169e-06↓	7.297e-08↓	0.3075
Aviation	H-1000	8.425e-07↓	9.943e-07↓	1.046e-06↓	5.653e-08↓	0.0312
Aviation	S-1000	9.001e-07↑	1.002e-06↓	1.088e-06↓	6.387e-08↓	0.0640
Aviation	SH-1000	8.706e-07↓	1.047e-06↓	1.231e-06↑	1.042e-07↑	0.4274
Aviation	V-2000	9.388e-07↑	9.970e-07↓	1.080e-06↓	**4.258e-08**↓	0.0211
Aviation	H-2000	8.956e-07↓	1.061e-06↓	1.215e-06↑	9.782e-08↑	0.6232
Aviation	S-2000	9.163e-07↑	1.036e-06↓	1.154e-06↓	7.710e-08↓	0.2730
Aviation	SH-2000	9.159e-07↑	1.061e-06↓	1.213e-06↑	9.475e-08↑	0.6232
Coal	V-no-repop	7.667e-05	8.541e-05	9.402e-05	6.306e-06	1.0000
Coal	H-no-repop	7.468e-05↓	8.345e-05↓	9.213e-05↓	5.894e-06↓	0.4727
Coal	S-no-repop	7.192e-05↓	8.154e-05↓	9.203e-05↓	6.443e-06↑	0.2413
Coal	SH-no-repop	7.131e-05↓	8.193e-05↓	9.049e-05↓	6.170e-06↓	0.3075
Coal	V-50	5.240e-05↓	**6.589e-05**↓	8.212e-05↓	8.288e-06↑	0.0006
Coal	H-50	5.519e-05↓	7.260e-05↓	9.377e-05↓	1.259e-05↑	0.0211
Coal	S-50	6.522e-05↓	7.172e-05↓	8.940e-05↓	7.500e-06↑	0.0013
Coal	SH-50	**4.718e-05**↓	7.243e-05↓	8.257e-05↓	1.147e-05↑	0.0073
Coal	V-150	6.697e-05↓	7.335e-05↓	8.446e-05↓	5.986e-06↓	0.0017
Coal	H-150	6.500e-05↓	7.219e-05↓	**8.170e-05**↓	5.743e-06↓	0.0013
Coal	S-150	6.348e-05↓	7.404e-05↓	8.804e-05↓	7.186e-06↑	0.0028
Coal	SH-150	6.985e-05↓	8.134e-05↓	9.258e-05↓	8.577e-06↑	0.1859
Coal	V-500	6.633e-05↓	7.844e-05↓	8.705e-05↓	7.628e-06↑	0.0890
Coal	H-500	6.446e-05↓	7.672e-05↓	8.416e-05↓	6.910e-06↑	0.0257
Coal	S-500	7.011e-05↓	7.877e-05↓	9.109e-05↓	7.113e-06↑	0.0376
Coal	SH-500	6.769e-05↓	7.802e-05↓	8.599e-05↓	6.930e-06↑	0.0640
Coal	V-1000	6.766e-05↓	7.749e-05↓	8.529e-05↓	6.004e-06↓	0.0173
Coal	H-1000	6.729e-05↓	7.915e-05↓	8.887e-05↓	6.918e-06↑	0.0640
Coal	S-1000	7.412e-05↓	8.029e-05↓	8.420e-05↓	**3.864e-06**↓	0.0757
Coal	SH-1000	7.050e-05↓	7.811e-05↓	8.932e-05↓	6.343e-06↑	0.0173
Coal	V-2000	7.261e-05↓	8.064e-05↓	9.091e-05↓	6.403e-06↑	0.1859
Coal	H-2000	7.160e-05↓	8.056e-05↓	8.987e-05↓	5.910e-06↓	0.1405
Coal	S-2000	7.266e-05↓	8.179e-05↓	8.848e-05↓	4.787e-06↓	0.2123
Coal	SH-2000	7.285e-05↓	8.246e-05↓	8.924e-05↓	4.925e-06↓	0.3075
Wind	V-no-repop	1.341e-03	1.351e-03	1.361e-03	6.733e-06	1.0000
Wind	H-no-repop	1.342e-03↑	1.351e-03↑	1.365e-03↑	6.981e-06↑	0.9097
Wind	S-no-repop	1.313e-03↓	1.345e-03↓	1.356e-03↓	1.201e-05↑	0.3075
Wind	SH-no-repop	1.338e-03↓	1.349e-03↓	1.357e-03↓	5.305e-06↓	0.4727
Wind	V-50	1.309e-03↓	1.339e-03↓	1.356e-03↓	1.286e-05↑	0.0173
Wind	H-50	1.316e-03↓	1.337e-03↓	**1.347e-03**↓	9.432e-06↑	0.0036
Wind	S-50	**1.258e-03**↓	**1.332e-03**↓	1.354e-03↓	2.833e-05↑	0.0257
Wind	SH-50	1.331e-03↓	1.346e-03↓	1.355e-03↓	7.776e-06↑	0.2123
Wind	V-150	1.332e-03↓	1.348e-03↓	1.361e-03↓	8.119e-06↑	0.6232
Wind	H-150	1.329e-03↓	1.345e-03↓	1.362e-03↑	1.337e-05↑	0.5205
Wind	S-150	1.335e-03↓	1.344e-03↓	1.357e-03↓	6.479e-06↓	0.0890
Wind	SH-150	1.339e-03↓	1.348e-03↓	1.363e-03↑	8.211e-06↑	0.3847
Wind	V-500	1.332e-03↓	1.343e-03↓	1.357e-03↓	6.622e-06↓	0.0376
Wind	H-500	1.333e-03↓	1.347e-03↓	1.359e-03↓	7.679e-06↑	0.2730
Wind	S-500	1.337e-03↓	1.346e-03↓	1.361e-03↓	7.174e-06↑	0.0890
Wind	SH-500	1.343e-03↑	1.348e-03↓	1.358e-03↓	**3.823e-06**↓	0.2730
Wind	V-1000	1.326e-03↓	1.346e-03↓	1.365e-03↑	1.259e-05↑	0.3256
Wind	H-1000	1.338e-03↓	1.352e-03↑	1.363e-03↑	8.191e-06↑	0.7913
Wind	S-1000	1.339e-03↓	1.348e-03↓	1.361e-03↓	6.977e-06↑	0.2567
Wind	SH-1000	1.333e-03↓	1.346e-03↓	1.365e-03↑	8.853e-06↑	0.1212
Wind	V-2000	1.330e-03↓	1.348e-03↓	1.360e-03↓	7.516e-06↑	0.2730
Wind	H-2000	1.331e-03↓	1.349e-03↓	1.360e-03↓	1.111e-05↑	0.8501
Wind	S-2000	1.336e-03↓	1.348e-03↓	1.363e-03↑	7.780e-06↑	0.4274
Wind	SH-2000	1.337e-03↓	1.347e-03↓	1.357e-03↓	5.505e-06↓	0.1859

Table 2: Total Enabled Edges for all datasets. P-values via Mann-Whitney U test vs V-no-repop (baseline). ↓/↑ = lower/higher than baseline. **Bold** = best in column. V=Vanilla, H=Harada, S=SWEET, SH=SWEET+Harada.

Dataset	Experiment	Min	Avg	Max	Std	p
Aviation	V-no-repop	**46.0**	69.8	102.0	17.76	1.0000
Aviation	H-no-repop	51.0↑	82.2↑	133.0↑	28.80↑	0.4274
Aviation	S-no-repop	48.0↑	**61.4**↓	**76.0**↓	**9.24**↓	0.3640
Aviation	SH-no-repop	54.0↑	68.6↓	100.0↓	16.15↓	0.8499
Aviation	V-50	52.0↑	110.2↑	149.0↑	32.06↑	0.0073
Aviation	H-50	61.0↑	115.7↑	173.0↑	37.32↑	0.0058
Aviation	S-50	70.0↑	168.2↑	453.0↑	115.10↑	0.0036
Aviation	SH-50	64.0↑	153.2↑	216.0↑	43.74↑	0.0010
Aviation	V-150	62.0↑	116.2↑	211.0↑	47.42↑	0.0113
Aviation	H-150	58.0↑	138.6↑	248.0↑	64.39↑	0.0155
Aviation	S-150	72.0↑	114.3↑	195.0↑	42.70↑	0.0065
Aviation	SH-150	67.0↑	92.2↑	150.0↑	23.42↑	0.0342
Aviation	V-500	50.0↑	83.0↑	176.0↑	35.01↑	0.3843
Aviation	H-500	51.0↑	114.0↑	183.0↑	46.26↑	0.0539
Aviation	S-500	59.0↑	80.1↑	109.0↑	17.75↓	0.1979
Aviation	SH-500	62.0↑	109.1↑	169.0↑	34.88↑	0.0081
Aviation	V-1000	65.0↑	87.5↑	121.0↑	19.46↑	0.0586
Aviation	H-1000	53.0↑	106.6↑	279.0↑	63.67↑	0.0492
Aviation	S-1000	49.0↑	63.8↓	90.0↓	11.65↓	0.5201
Aviation	SH-1000	67.0↑	89.3↑	116.0↑	20.69↑	0.0450
Aviation	V-2000	59.0↑	83.6↑	155.0↑	27.48↑	0.2261
Aviation	H-2000	56.0↑	75.6↑	107.0↑	19.93↑	0.4494
Aviation	S-2000	59.0↑	78.7↑	111.0↑	16.06↓	0.2413
Aviation	SH-2000	**46.0**↓	76.6↑	132.0↑	30.89↑	0.8205
Coal	V-no-repop	**26.0**	58.9	99.0	23.26	1.0000
Coal	H-no-repop	34.0↑	77.6↑	157.0↑	39.79↑	0.3447
Coal	S-no-repop	29.0↑	56.1↓	104.0↑	22.94↓	0.7335
Coal	SH-no-repop	32.0↑	73.9↑	132.0↑	32.46↑	0.3071
Coal	V-50	92.0↑	142.6↑	285.0↑	54.85↑	0.0002
Coal	H-50	95.0↑	155.3↑	370.0↑	80.50↑	0.0003
Coal	S-50	46.0↑	133.8↑	369.0↑	94.25↑	0.0113
Coal	SH-50	73.0↑	146.7↑	280.0↑	58.84↑	0.0007
Coal	V-150	44.0↑	91.0↑	157.0↑	32.81↑	0.0342
Coal	H-150	80.0↑	132.5↑	314.0↑	69.88↑	0.0004
Coal	S-150	47.0↑	94.8↑	168.0↑	36.38↑	0.0310
Coal	SH-150	42.0↑	88.1↑	132.0↑	29.47↑	0.0311
Coal	V-500	52.0↑	82.2↑	128.0↑	29.69↑	0.1303
Coal	H-500	34.0↑	94.4↑	218.0↑	50.59↑	0.0492
Coal	S-500	33.0↑	64.5↑	96.0↓	21.58↓	0.5452
Coal	SH-500	44.0↑	71.4↑	98.0↓	16.90↓	0.1981
Coal	V-1000	40.0↑	72.9↑	205.0↑	48.89↑	0.9097
Coal	H-1000	39.0↑	80.7↑	112.0↑	18.66↓	0.0173
Coal	S-1000	38.0↑	66.7↑	109.0↑	25.14↑	0.4960
Coal	SH-1000	60.0↑	76.4↑	96.0↓	**10.82**↓	0.0637
Coal	V-2000	28.0↑	67.3↑	121.0↑	26.58↑	0.5705
Coal	H-2000	39.0↑	86.9↑	144.0↑	33.21↑	0.0962
Coal	S-2000	52.0↑	76.5↑	100.0↑	18.18↓	0.1211
Coal	SH-2000	29.0↑	**53.6**↓	**78.0**↓	14.62↓	0.6499
Wind	V-no-repop	61.0	80.1	128.0	19.39	1.0000
Wind	H-no-repop	60.0↓	95.7↑	144.0↑	26.70↑	0.1506
Wind	S-no-repop	51.0↓	76.6↓	116.0↓	19.48↑	0.7336
Wind	SH-no-repop	44.0↓	72.9↓	114.0↓	20.00↑	0.4960
Wind	V-50	73.0↑	141.4↑	249.0↑	68.24↑	0.0064
Wind	H-50	58.0↓	160.8↑	294.0↑	74.56↑	0.0190
Wind	S-50	72.0↑	134.8↑	278.0↑	72.62↑	0.0081
Wind	SH-50	79.0↑	136.4↑	255.0↑	54.29↑	0.0032
Wind	V-150	68.0↑	110.7↑	198.0↑	39.81↑	0.0342
Wind	H-150	60.0↓	108.1↑	194.0↑	49.44↑	0.3071
Wind	S-150	53.0↓	86.4↑	146.0↑	25.90↑	0.4960
Wind	SH-150	70.0↑	109.7↑	161.0↑	33.11↑	0.0538
Wind	V-500	63.0↑	93.0↑	138.0↑	23.75↑	0.1205
Wind	H-500	64.0↑	97.1↑	156.0↑	27.31↑	0.1037
Wind	S-500	48.0↓	**69.5**↓	**99.0**↓	19.86↑	0.2261
Wind	SH-500	58.0↓	91.5↑	141.0↑	25.22↑	0.2563
Wind	V-1000	43.0↓	71.9↓	104.0↓	18.10↓	0.4044
Wind	H-1000	52.0↓	90.2↑	120.0↓	24.98↑	0.3638
Wind	S-1000	64.0↑	92.0↑	117.0↓	**15.38**↓	0.0582
Wind	SH-1000	**35.0**↓	79.0↓	143.0↑	29.62↑	1.0000
Wind	V-2000	50.0↓	83.4↑	118.0↓	23.55↑	0.5964
Wind	H-2000	43.0↓	88.8↑	175.0↑	37.25↑	0.7335
Wind	S-2000	41.0↓	89.9↑	129.0↑	23.80↑	0.1301
Wind	SH-2000	59.0↓	91.2↑	125.0↓	20.03↑	0.1506

7 Discussion

7.1 Repopulation as the Primary Driver

The most consistent finding across all three datasets is that island extinction repopulation frequency, rather than parent selection bias correction, is the primary driver of predictive performance improvement. Both Harada's frequency-based selection and SWEET were specifically designed to address evaluation-time bias in asynchronous evolutionary systems, yet neither produces measurable MSE improvement when applied without repopulation. We attribute this to the structural level at which bias operates in EXAMM's island model: bias manifests at the island level, with some islands accumulating high-quality genomes while others stagnate and consume computational budget without contributing useful diversity. Harada and SWEET operate as within-island interventions and cannot address this island-level divergence as they are the right solution applied at the wrong level of the system.

To understand why T dominates performance: at every T genome evaluations the worst-performing island is entirely destroyed and repopulated with mutated copies of the best genome from the best-performing island [13]. At $T = 50$, this reset fires across all $K = 10$ islands after every 50 evaluations, meaning each island accumulates at most $T/K = 5$ new genomes before the worst island is replaced – which is fewer than half an island generation for bias to compound [18, 19] before extinction intervenes. At $T = \infty$, bias compounds across the full 10,000 evaluation budget, explaining why no-repopulation conditions consistently produce the worst MSE regardless of bias correction strategy. This parallels Skolicki and De Jong's finding that migration interval dominates island model performance [21] where in both cases a renewal frequency parameter outweighs within-island operator design.

7.2 Repopulation Tradeoff: MSE, Complexity, and Dataset Interactions

Frequent repopulation improves MSE by seeding underperforming islands before they converge prematurely, with diminishing returns beyond repop-500 indicating repopulation must be frequent enough to prevent inter-event stagnation. However, repop-50 simultaneously drives a 2-3$\times$ increase in average enabled edges across all datasets (Tables 1-2), from a baseline of 59-80 to 133-168 at repop-50. This occurs because best-performing islands carry larger architectures than those that emerge under evaluation-time bias alone. The seeded complexity propagates through subsequent mutation and crossover, creating a positive feedback loop between repopulation events and architecture growth. This also explains why SWEET, which trends toward more compact architectures, shows elevated min-max variance under repop-50. For deployment-constrained applications, repop-500 to repop-1000 offers a better balance between MSE improvement and architectural compactness.

Strategy effectiveness varies by dataset. On Aviation, Harada and SWEET +Harada provide a modest advantage at repop-50 and repop-150, suggesting within-island frequency correction adds value in more complex search spaces. On Wind, SWEET consistently outperforms other strategies, consistent with Scott et al.'s finding that SWEET is most effective when evaluation time and solution quality are positively correlated [17]. On Coal, Vanilla-repop-50 achieves the best MSE, with bias correction introducing higher variance rather than lower error, suggesting repopulation alone fully compensates for evaluation-time bias on this dataset. Prediction visualizations (Appendix **??**[4]) confirm these patterns qualitatively: Aviation predictions track smoothly across all three test flights, Wind generalizes to unseen turbines with symmetric zero-centered error, and Coal requires clipped visualization to reveal the model's step-function tracking behavior.

7.3 Comparison with Prior Work

These findings diverge from Harada [9], Scott et al. [17], and Karns and Desell [11], all of whom validated bias correction in single-population master-worker architectures where evaluation-time bias accumulates continuously. In EXAMM's island model, periodic extinction and repopulation create a structural buffer that partially substitutes for explicit bias correction an interaction absent in prior single-population settings. The additive benefit of SWEET+ Harada at repop-50 on Aviation ($p = 0.0036$) suggests bias correction retains complementary value when combined with frequent structural repopulation, but cannot substitute for it. Whether this interaction strengthens with larger island populations remains an open question for future work.

8 Conclusion

This work investigates the interaction between evaluation-time bias correction strategies (Harada frequency-based selection, SWEET, and their combination) and island extinction repopulation frequency in the EXA-GP neuroevolution framework across three real-world time-series datasets: Aviation Flight Recorder, Coal Power Plant, and Wind Turbine data.

Our results establish three primary findings. First, repopulation frequency is the dominant factor in the studied setting, consistently outweighing explicit bias correction strategies: frequent repopulation (repop-50 to repop-150) produces consistent and statistically significant MSE improvements across all datasets, while bias correction strategies applied without repopulation produce no significant improvement over the baseline. Second, frequent repopulation introduces a complexity cost. Architectures evolved under repop-50 are 2–3× larger than those evolved without repopulation, with higher variance across trials, suggesting that repop-500 to repop-1000 offers a more practical balance between predictive performance and architectural compactness. Third, the effectiveness of

[4] The Appendix is available online at https://doi.org/10.5281/zenodo.19646111.

specific bias correction strategies is dataset-dependent: SWEET provides the largest gains on Wind, Harada and SWEET+Harada provide modest gains on Aviation, and no bias correction strategy meaningfully outperforms Vanilla on Coal when repopulation is held constant.

These findings suggest that in island-based neuroevolution systems, structural population management mechanisms such as extinction-based repopulation may subsume much of the benefit that explicit evaluation-time bias correction provides in single-population architectures. Future work should investigate adaptive repopulation schedules that dynamically adjust the repopulation frequency, T, based on island fitness stagnation signals, avoiding the fixed-interval tradeoff between exploration cost and convergence speed. A natural extension is to examine whether the complexity growth induced by frequent repopulation translates to improved generalization on held-out test data. Appendix A provides qualitative test-set predictions for the single best genome per dataset, confirming that evolved models generalize beyond the validation partition; however, systematic test-set evaluation across all 24 conditions remains future work, and larger architectures may overfit despite lower validation error. Finally, extending these experiments to larger island populations (e.g., $K = 20$, $N = 20$) would test whether within-island bias correction plays a more decisive role when islands have sufficient capacity for bias to compound meaningfully between repopulation events.

Acknowledgements

This work has been partially supported by a gift from Lenovo, and by the U.S. National Science Foundation under Grant Number #2225354. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of Lenovo or the National Science Foundation. The authors also thank RIT's Research Computing team for their assistance and support [2].

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. National general aviation flight information database (NGAFID). https://ngafid.org
2. Research computing, rochester institute of technology. https://www.rit.edu/researchcomputing
3. Alba, E., Luque, G., Nesmachnow, S.: Parallel metaheuristics: recent advances and new trends. Int. Trans. Oper. Res. **20**(1), 1–48 (2013). https://doi.org/10.1111/j.1475-3995.2012.00862.x
4. Alba, E., Tomassini, M.: Parallelism and evolutionary algorithms. IEEE transactions on evolutionary computation **6**(5), 443–462 (2002)

5. Depolli, M., Trobec, R., Filipič, B.: Asynchronous master-slave parallelization of differential evolution for multi-objective optimization. Evol. Comput. **21**(2), 261–291 (2013). https://doi.org/10.1162/EVCO_a_00076
6. Desell, T., ElSaid, A., Ororbia, A.G.: An empirical exploration of deep recurrent connections using neuro-evolution. In: The 23rd International Conference on the Applications of Evolutionary Computation (EvoStar: EvoApps) (2020)
7. Galván, E., Mooney, P.: Neuroevolution in deep neural networks: Current trends and future challenges. IEEE Transactions on Artificial Intelligence **2**(6), 476–493 (2021)
8. Harada, T.: Mathematical model of asynchronous parallel evolutionary algorithm to analyze influence of evaluation time bias. In: 2019 IEEE Asia-Pacific Conference on Computer Science and Data Engineering (CSDE). pp. 1–8 (2019)
9. Harada, T.: A frequency-based parent selection for reducing the effect of evaluation time bias in asynchronous parallel multi-objective evolutionary algorithms. Natural Computing **24**, 211–225 (2025). https://doi.org/10.1007/s11047-022-09940-z
10. Harada, T., Alba, E.: Parallel genetic algorithms: a useful survey. ACM Comput. Surv. (2020). https://doi.org/10.1145/3400031
11. Karns, J., Desell, T.: Evaluation time bias in asynchronous evolutionary algorithms: A replication study and a novel mitigation strategy. In: Proceedings of the Genetic and Evolutionary Computation Conference. pp. 13–21 (2025)
12. Lyu, Z., ElSaid, A., Karns, J., Mkaouer, M., Desell, T.: An experimental study of weight initialization and Lamarckian inheritance on neuroevolution. In: The 24th International Conference on the Applications of Evolutionary Computation (EvoStar: EvoApps) (2021)
13. Lyu, Z., Karns, J., ElSaid, A., Mkaouer, M., Desell, T.: Improving distributed neuroevolution using island extinction and repopulation. In: The 24th International Conference on the Applications of Evolutionary Computation (EvoStar: EvoApps) (2021)
14. Murphy, J., Desell, T.: Minimizing the EXA-GP graph-based genetic programming algorithm for interpretable time series forecasting. In: Proceedings of the Genetic and Evolutionary Computation Conference Companion (GECCO '24 Companion) (2024). https://doi.org/10.1145/3638530.3664173
15. Murphy, J., Kar, D., Karns, J., Desell, T.: EXA-GP: Unifying graph-based genetic programming and neuroevolution for explainable time series forecasting. In: Proceedings of the Genetic and Evolutionary Computation Conference Companion (GECCO '24 Companion) (2024). https://doi.org/10.1145/3638530.3654349
16. Ororbia, A., ElSaid, A., Desell, T.: Investigating recurrent neural network memory structures using neuro-evolution. In: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '19). pp. 446–455 (2019). https://doi.org/10.1145/3321707.3321795
17. Scott, E.O., Coletti, M., Schuman, C.D., Kay, B., Kulkarni, S.R., Parsa, M., Gunaratne, C., De Jong, K.A.: Avoiding excess computation in asynchronous evolutionary algorithms. Expert Systems **40**(5), e13100 (2023). https://doi.org/10.1111/exsy.13100
18. Scott, E.O., De Jong, K.A.: Understanding simple asynchronous evolutionary algorithms. In: Proc. FOGA XIII. pp. 85–98 (2015). https://doi.org/10.1145/2725494.2725509
19. Scott, E.O., De Jong, K.A.: Evaluation-time bias in quasi-generational and steady-state asynchronous evolutionary algorithms. In: Proc. GECCO. pp. 845–852 (2016). https://doi.org/10.1145/2908812.2908934

20. Singh, A., Desell, T.: Biologically-inspired homeostasis for neuroevolution: Alternating growth and pruning phases. In: Applications of Evolutionary Computation (EvoApplications 2026). Lecture Notes in Computer Science, vol. 16524. Springer, Cham (2026), in press. LNCS 16524, Chap. 32
21. Skolicki, Z., De Jong, K.: The influence of migration sizes and intervals on island models. In: Proceedings of the 7th annual conference on Genetic and evolutionary computation. pp. 1295–1302 (2005)
22. Stanley, K.O., Miikkulainen, R.: Evolving neural networks through augmenting topologies. Evolutionary computation **10**(2), 99–127 (2002)
23. Wessing, S., Rudolph, G., Menges, D.A.: Comparing asynchronous and synchronous parallelization of the SMS-EMOA. In: Parallel Problem Solving from Nature- PPSN XIV. pp. 558–567 (2016)