Agentic Discovery of Multi-Channel Bioacoustic Association Algorithms

Paramjyoti Mohapatra[1],[*] Irina Tolkova[1], Akanksha Sarkar[1], Atharva Sehgal[2], Daniel P. Salisbury[1], Andrew L. Von Duyke[3], Léa Bouffaut[1], Holger Klinck[1], Jennifer J. Sun[1]
[1]Cornell University [2]University of Texas at Austin [3]North Slope Borough

Abstract

Multi-channel call association is an understudied yet essential problem in bioacoustics, serving as a critical building block for localization and density estimation. The complexity of the task stems from the fact that a robust algorithm would involve integrating spectral and temporal features across receivers. As a result, conservation researchers currently rely on time-consuming manual annotation. The challenge of searching this large, multi-constraint algorithmic space makes it an ideal testbed for agentic AI, which has recently shown a strong ability to discover novel algorithms. In this work, we investigate the effectiveness of agentic approaches for discovering generalizable algorithms for multi-channel association. We formulate multi-channel association as an agentic program synthesis problem, where candidate algorithms are discovered through iterative empirical evaluation and feedback. To address the scarcity of labeled real-world data, we rely on physics-inspired simulation and introduce a regime-scheduled evaluation and feedback strategy with warm restarts that expose candidate programs to synthetic conditions of increasing difficulty. The resulting constraint-aware algorithms generalize to unseen simulation regimes and transfer effectively to real whale call datasets, improving association accuracy.

1 Introduction

In a time of global biodiversity loss, marine mammals also face persistent threats – from ship strikes to climate-induced prey shifts (Avila et al., 2018; Nelms et al., 2021). Understanding the spatiotemporal distributions of threatened species is essential for effective conservation interventions (Sequeira et al., 2025). Passive acoustic monitoring has been a central methodology for studying the distribution, abundance, and behavior of marine mammals (Fleishman et al., 2023; Marques et al., 2013).

Over the past two decades, machine learning has transformed acoustic monitoring by enabling automated detection and species-level classification of animal vocalizations, substantially expanding the spatial and temporal scale of ecological analysis. However, most machine learning approaches in bioacoustics have focused on single-channel audio (Stowell, 2022). In contrast, many of the most critical conservation tasks—such as localizing and tracking individual animals—require multi-channel acoustic arrays (Rhinehart et al., 2020; Wijers et al., 2021). This introduces a significantly harder technical challenge: *multi-channel call association*. To localize a call, one must first determine which detections, recorded across a sparse array of receivers, originated from the same source. This is a complex combinatorial problem that requires jointly reasoning over learned signal similarity (e.g., spectrogram structure) and hard physical constraints such as time-difference-of-arrival (TDoA). In practice, reliable association is further complicated by noise, varying source densities, clock drift, and environmental effects.

Despite its importance, automated multi-channel association remains difficult to scale for practical use, forcing researchers to rely on time-consuming manual annotation. One class of approaches applies signal-processing heuristics tailored to specific array geometries or vocalization types, which are often brittle and difficult to generalize. Another class uses standard deep learning techniques such as metric learning, but these are typically designed for single-channel tasks and struggle to

[*]Corresponding author: pm656@cornell.edu

incorporate the hard physical constraints required for multi-channel association. Moreover, such methods depend on large-scale, manually associated datasets for each array configuration, which are largely unavailable in bioacoustics. A general and automated framework for multi-channel call association would therefore substantially streamline bioacoustic analysis.

While domain experts understand the relevant physical constraints, designing algorithms that effectively combine these constraints with spectral similarity in noisy, data-scarce settings is non-trivial. Recent work on agentic AI (Jiang et al., 2025; Toledo et al., 2025), powered by large language models (LLMs), has shown promise in addressing such challenges by iteratively searching over spaces of executable programs guided by empirical objectives and feedback. However, a key bottleneck in applying agentic program synthesis to scientific domains is evaluation: large-scale real-world datasets with ground-truth associations are not available, making it difficult to provide reliable feedback during search. Unlike prior agentic algorithm discovery work, which evaluates candidate programs under a fixed environment, we argue that in constraint-heavy scientific domains the structure of the feedback itself must evolve during search. Fixed simulators can implicitly bias discovery toward brittle heuristics that exploit artifacts of a single regime. We therefore treat simulator scheduling as a first-class design component of the program synthesis loop.

In scientific domains such as bioacoustics, scalable evaluation and feedback can be done using synthetic data generated by physics-inspired simulators. However, simulation design introduces a fundamental tension. Simulations that are too simplistic may reward brittle heuristics that exploit artifacts of synthetic data and fail to generalize, while overly complex simulations can overwhelm the search process, producing noisy or uninformative feedback.

In this work, we address this limitation by introducing a *regime-scheduled simulation orchestrator* for agentic program synthesis. Easy initial configurations help give useful feedback signals to build up a base algorithm that can be optimized further to tackle more challenging regime configurations in the later stages. The idea here is similar to that of curriculum learning in machine learning (Bengio et al., 2009).

To summarize, our contributions are as follows:

- We formulate multi-channel association as an agentic program synthesis problem, enabling automated discovery of heuristic algorithms (including adaptive preprocessing) for a core bioacoustics task without relying on large labeled real-world datasets during search.
- We introduce a regime-scheduled simulation orchestrator that evaluates candidate programs under a sequence of synthetic conditions of increasing difficulty, addressing key limitations of fixed simulator configurations in agentic program synthesis.
- Through controlled ablations, we analyze how warm starts and regime scheduling affect the diversity and stability of discovered algorithms.
- We show that algorithms discovered using regime-scheduled simulation generalize to unseen synthetic conditions and transfer meaningfully to held-out synthetic datasets and real-world bioacoustic datasets.

2 Related Work

Computational Bioacoustics & Source Association Computational bioacoustics has traditionally relied on classical signal processing and array analysis methods derived from naval sonar research (Au & Hastings, 2008). Over the past decade, deep learning has become dominant for detection and species-level classification of animal vocalizations, demonstrating strong performance across both low-frequency baleen whales and high-frequency toothed whales (Shiu et al., 2020; Kirsebom et al., 2020; Schall et al., 2024; Frasier, 2021).

Despite these advances, most methods operate on single-channel audio, while multi-channel association remains comparatively underexplored. Acoustic arrays are widely used for localization and density estimation (Sousa-Lima et al., 2013; Fleishman et al., 2023), where accurate cross-receiver call association is a prerequisite. Prior approaches typically enforce physical feasibility through TDoA rejection or combine temporal constraints with cross-correlation-based similarity measures in rule-based or metric-driven pipelines (Helble et al., 2015; Nosal, 2013; Lapp et al., 2023; Hendricks et al., 2019; Baggenstoss, 2011). However, these techniques often degrade under high call density or require extensive manual tuning, and manual association remains common in practice (Tenorio-Hallé et al., 2022). Once association is resolved, localization is typically performed via multilateration using least-squares, Bayesian, or likelihood-based formulations (Urazghildiiev & Clark, 2013; Rideout et al., 2013).

AI Agents for Science and Algorithm Discovery Recent progress in large language models has enabled agentic frameworks that iteratively generate, execute, and refine code, reducing the manual effort required to build scientific software. Such systems have demonstrated utility across domains including biology, mathematics, and materials science (Huang et al., 2025; Novikov et al., 2025; Zhang et al., 2024), forming part of a broader movement in AI for Science (Wang et al., 2023). Unlike standard predictive tasks, multi-channel bioacoustic association is a constrained symbolic problem that requires balancing learned similarity measures with hard physical constraints, making it well-suited to search-based agentic approaches.

Parallel advances in LLM code generation increasingly treat program synthesis as an iterative decision process guided by execution feedback and structured search rather than single-pass decoding (Chen, 2021; Roziere et al., 2023; Snell et al., 2024). Tree and population-based strategies—including beam search, Monte-Carlo tree search, and evolutionary refinement—have shown consistent gains on coding and reasoning benchmarks by maintaining and improving multiple candidate programs (Yao et al., 2023; Wang et al., 2022; Hazra et al., 2024).

3 METHOD

Problem Setup We assume detections are given as input and focus exclusively on the association problem, excluding upstream detection or segmentation. Each detection includes spectrogram data together with metadata such as time of arrival, receiver location, and receiver identity. The task is to partition detections into groups corresponding to individual sources.

Effective association depends not only on grouping logic but also on adaptive preprocessing and representation of raw observations, particularly under varying noise conditions. We therefore treat preprocessing, feature extraction, and constrained grouping as a single algorithmic pipeline to be discovered. Given a candidate algorithm, predicted groupings are evaluated against ground truth using task-specific objective Hungarian-matched precision, recall, and F1 scores, while enforcing domain constraints such as physical feasibility or channel uniqueness. The exact problem description provided to the agent can be found in C.

3.1 AGENTIC ALGORITHM GENERATION FRAMEWORK

Iterative Synthesis Loop Algorithm generation proceeds as an iterative propose–evaluate–refine process. At each step, the agent proposes a candidate `associate_calls` program conditioned on the task description and previously evaluated candidates. The candidate is executed on simulated instances and assigned a score which is then used to guide subsequent edits.

Search Policy For the base search policy, we adopt a UCT-based tree search over candidate programs, building on Jiang et al. (2025). Nodes correspond to candidate programs and edges correspond to edit operations. Selection balances exploitation of high-scoring candidates with exploration of under-visited branches via a UCT criterion. More details about the policy can be found in B in the appendix.

Evaluation Objective Each candidate program implements `associate_calls(detections)` and outputs a grouping for all detections in an instance. We compare predicted groupings to ground truth using Hungarian-matched, weighted F1. Alongside F1, we compute structural diagnostics (e.g., singleton fraction, duplicate-channel rate, mean group size, precision, recall, and soft TDoA violation frequency) to characterize physical plausibility and failure modes. These diagnostics are provided as textual feedback during the synthesis loop.

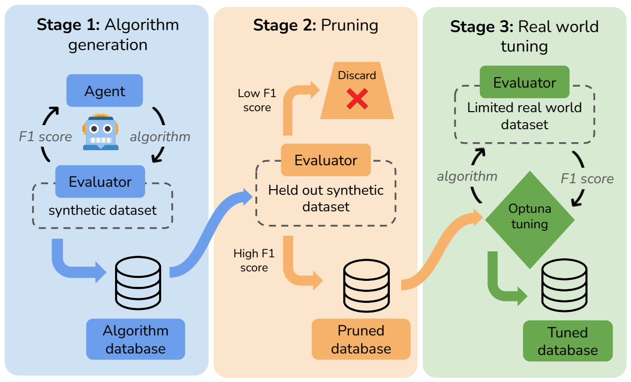


Figure 1: Three-stage agentic synthesis pipeline consisting of algorithm generation on synthetic data, pruning using evaluation on held-out synthetic data, and light parameter tuning using Optuna on target data. The first stage generates a large corpus of programs for our task, the second stage prunes out programs that fail to generalize to unseen synthetic setting, while the third stage tunes the parameters on a small subset of our target set.

3.2 Simulation-Based Evaluation

Simulator We use a physics-based simulator to generate synthetic multi-channel association problems with known ground-truth groupings. Each simulated instance consists of a set of detections with associated metadata (receiver identity, time of arrival, and spectrograms), generated by sampling latent source locations and simulating their propagation across an acoustic array.

The simulator exposes several controllable parameters that affect task difficulty while preserving the underlying physical structure of the problem. These include: (i) the number of calls (source density), (ii) the number of receivers, (iii) Odds of a call being picked up by a receiver (Det. Noise where larger values corresponds to wider detection radius), and (iv) spectrogram corruption via additive noise and masking. By varying these parameters, the simulator produces association problems with differing levels of ambiguity and noise.

Scheduled Simulation Regimes Rather than evaluating candidate programs under a single fixed simulator configuration, we define a sequence of pre-specified simulation regimes with increasing difficulty. Each regime corresponds to a distinct setting of simulator parameters, summarized in Table 9 in the Appendix. Early regimes emphasize simpler scenarios with lower density and noise, while later regimes progressively increase ambiguity through higher call density, varying receiver geometries, and stronger spectrogram corruption.

For each regime, we ensure that the algorithm is allowed to evolve for at least 20 and at most 60 iterations. Between 20 and 60 iterations, an algorithm is allowed to progress to next regime with if it has achieved a specified threshold objective score – this ensures that we don't waste compute to over optimize on regimes where the algorithm already has good performance.

3.3 Warm Starts

When transitioning between successive simulation regimes, we employ a warm-start strategy. Rather than restarting synthesis from scratch, each new regime is initialized using the best-performing algorithm discovered in the previous regime and the problem description is modified to include a description of the best performing algorithm from all previous stages. This allows the search process to incrementally refine existing algorithmic structure as task difficulty increases. Based on previous works, we hypothesize that a combination of warm starts and schedule regimes can facilitate stable convergence and help converge to better solutions (Bengio et al., 2009).

4 Experiments

Our experiments are designed to study i) how well do algorithms generated using simulated data generalize to unseen data; ii) what level of fine tuning might be needed post algorithm discovery

E1	Single regime / no warm start / F1-only	**E2**	Single regime / warm start / F1-only
E3	Scheduled regimes / warm start / F1-only	**E4**	Scheduled regimes / warm start / composite
E5	Combined regime / no warm start / F1-only		

Table 1: Glossary of experimental configurations. Refer§4.1 for further discussions.

for effective sim to real transfer; iii) how scheduled simulation regimes and warm-start strategies influence the quality, robustness, and diversity of algorithms discovered during agentic program synthesis. In particular, we want to analyze how algorithms generated transfer to i) Synthetic held out sets; ii) Real World Datasets. We focus first on controlled synthetic evaluation, and subsequently assess real-world transfer.

Baselines We compare the accuracy (weighted F1) of the generated algorithms against the following:

B1 (DL Embedding + Clustering); **B2** (Greedy TDoA based clustering); **B3** (GPT-5.2, 1-shot algorithm extraction followed by Optuna finetuning).

4.1 Experimental Configurations

We consider five synthesis configurations to evaluate the impact of the following factors: (i) single regime versus scheduled simulation regimes, (ii) warm-started versus no warm-started synthesis, (iii) objective function design for algorithm ranking (F1 vs Composite score consisting of F1, precision, and singleton fraction). Namely, configurations E1-E5 helps us study those. All configurations use the same base search policy (3.1) and identical LLMs for code generation, feedback, and summarization (Gemini 2.5 Pro).

The configurations are presented in Table 1. Each configuration is run for three independent synthesis runs with different random seeds to capture the stochasticity of the search process.

4.2 Synthetic Evolution and Held-Out Evaluation

During synthesis, candidate algorithms are evaluated exclusively on synthetic problem instances from the current simulation regime (analogous to a training set). To avoid selection bias and overfitting to the training simulator distribution, algorithm selection is performed using a separate held-out synthetic dataset (analogous to a test set) that is not used during synthesis.

The held-out synthetic set is generated by sampling new random seeds and simulator realizations, with parameter values disjoint from those seen during synthesis.

Each synthesis run produces a collection of candidate algorithms evaluated over time. In particular, for the experiments E2, E3, and E4 with warm starts, we sample the best algorithm from the specified regimes (totaling 11 for these experiments). For E1, E5, and E6 with no levels or warm starts, we sample the top algorithm from 11 uniform bins from the entire run. We report the average F1 score of these top-5 algorithms for each run.

This protocol allows us to compare not only peak performance but also the consistency with which strong algorithms are discovered under different evaluation strategies while also demonstrating trends in performance over iterations. The results for this dataset can be found in 2.

4.3 Real-World Evaluation

Following synthetic evaluation and selection, we evaluate the selected algorithms on real-world multi-channel whale call dataset with manually annotated associations collected using the pipeline described in (Garcia et al., 2025). The dataset consists of 405 source whale calls of which 33 are used for Optuna tuning (Akiba et al., 2019) and 372 for validating algorithm performance. Optuna tuning is done for 50 iterations for each algorithm. Furthermore, since the original dataset's source calls were fairly sparse over time, we also use a variant of this dataset by artificially compressing the times to a duration of 400s (60 calls / minute) for validation. The results on this dataset can be found in tables 4, 5, 6, and 7. A more detailed description of this dataset can be found in D in the Appendix.

Table 2: F1 Scores using baselines and No Synthetic Data Configuration (E0)

Dataset	**B1**	**B2**	**B3**
Sparse Real World Dataset	0.34	0.67	0.72 ± 0.2
Dense Real World Dataset	0.34	0.45	0.47 ± 0.17

Table 3: Held-out synthetic F1 scores (Top-5) and iteration counts for each experiment (E1–E5), reported across three independent runs.

	Run 1		**Run 2**		**Run 3**	
Experiment	**F1**	**# Iterations**	**F1**	**# Iterations**	**F1**	**# Iterations**
Config. E1	0.904 ± 0.005	350	0.526 ± 0.192	350	0.442 ± 0.004	350
Config. E2	0.758 ± 0.080	660	0.940 ± 0.014	226	0.410 ± 0.020	660
Config. E3	0.940 ± 0.013	380	0.744 ± 0.017	620	0.930 ± 0.019	380
Config. E4	0.950 ± 0.006	225	0.930 ± 0.013	398	0.932 ± 0.011	348
Config. E5	0.396 ± 0.005	350	0.490 ± 0.190	350	0.380 ± 0	350

5 Discussion

Do the generated algorithms generalize to real world settings? Our results indicate that algorithms discovered via agentic program synthesis and selection in simulated enviroments exhibit partial zero-shot transfer to real-world datasets, but consistently benefit from lightweight parameter adaptation on target sets. Applying Optuna fine-tuning yields an average improvement of approximately $+0.10$ F1 on the sparse real-world dataset and $+0.04$ F1 on the dense real-world dataset across 9 runs.

Notably, certain configurations show substantially larger gains. For instance, E3 Run 1 improves by $+0.29$ F1 on the sparse dataset and $+0.12$ F1 on the dense dataset suggesting that while the structural heuristics discovered by the agent are often sound, their performance is sensitive to parameter calibration under domain shift.

These findings suggest that agentic program synthesis appears to be effective at identifying robust algorithmic structures when exposed to a varied challenging regimes while small amounts of target-domain tuning significantly enhances practical performance.

Are synthetic held out sets accurate indicators of performance on real world sets? We observe a strong positive correlation between held-out synthetic F1 and average post-tuning real-world F1 (averaged on sparse and dense real world sets) as seen in 2. This indicates that synthetic validation performance serves as a reliable ranking signal for selecting candidate algorithms prior to real-world tuning.

Do warm starts and the scheduled regime help? While the experiment with warm starts and schedule regimes (E3) achieved the best F1 on average, such performance metrics given a small number of runs are highly dependent on the initial solutions proposed by the LLM and hence will not be a suitable indicator of a good configuration/search policy. In order to quantify which setting is the most conducive to algorithm search in our predefined configurations we make note of the following:

Table 4: Performance on sparse real-world whale call datasets.

Algorithm	**Run 1**	**Run 2**	**Run 3**
E1 (Top 5)	0.90 ± 0.01	0.64 ± 0.02	0.756 ± 0.009
E2 (Top 5)	0.92 ± 0.11	0.654 ± 0.036	0.754 ± 0.018
E3 (Top 5)	0.61 ± 0.093	0.784 ± 0.135	0.87 ± 0.034

Table 5: Performance on sparse real-world whale call datasets after Optuna tuning.

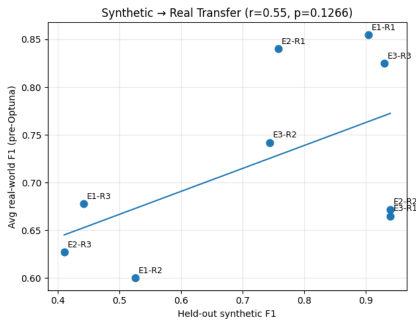
Algorithm	**Run 1**	**Run 2**	**Run 3**
E1 (Top 5)	0.933 ± 0.034	0.79 ± 0.012	0.79 ± 0.024
E2 (Top 5)	0.926 ± 0.12	0.84 ± 0.035	0.81 ± 0.044
E3 (Top 5)	0.907 ± 0.03	0.842 ± 0.11	0.93 ± 0.02

Table 6: Performance on dense real-world whale call datasets.

Algorithm	Run 1	Run 2	Run 3
E1 (Top 5)	0.81 ± 0.013	0.56 ± 0.073	0.60 ± 0.00
E2 (Top 5)	0.76 ± 0.04	0.69 ± 0.09	0.50 ± 0.02
E3 (Top 5)	0.72 ± 0.06	0.70 ± 0.023	0.78 ± 0.023

Table 7: Performance on dense real-world whale call datasets after Optuna tuning.

Algorithm	Run 1	Run 2	Run 3
E1 (Top 5)	0.806 ± 0.005	0.544 ± 0.08	0.654 ± 0.013
E2 (Top 5)	0.77 ± 0.025	0.79 ± 0.035	0.6 ± 0.1
E3 (Top 5)	0.84 ± 0.015	0.7 ± 0.04	0.78 ± 0.008



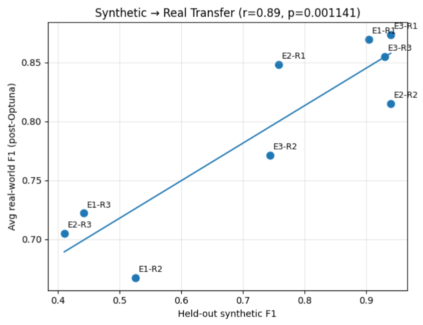


Figure 2: Correlation between held-out synthetic F1 and real-world F1 pre and post Optuna tuning. **Left:** Pre-tuning correlation (r = 0.55, p = 0.13). **Right:** Post-tuning correlation (r = 0.89, p = 0.001).

i) a good configuration can help improve on heuristics/properties of algorithms found in the initial stages; ii) a good search policy will retain the good properties of high performing algorithms from the earlier stages so the performance should not degrade over time; iii) the search process should be more stochastic/chaotic in the early stages and stabilize over time . We quantify these effects using the following metrics:

$$\Delta_{\text{recovery}}\ (\uparrow) = \max_{t\in\{8,9,10\}} \mathrm{F1}_t - \max_{t\in\{0,1,2\}} \mathrm{F1}_t$$

$$\begin{aligned}\Delta_{\text{avg}}\ (\uparrow) &= \overline{\mathrm{F1}}_{\text{late}} - \overline{\mathrm{F1}}_{\text{early}}\\ &= \frac{1}{3}\sum_{t\in\{8,9,10\}} \mathrm{F1}_t - \frac{1}{3}\sum_{t\in\{0,1,2\}} \mathrm{F1}_t\end{aligned}$$

$$\begin{aligned}\Delta_{\mathrm{Var}}\ (\downarrow) &= \mathrm{Var}_{\text{late}} - \mathrm{Var}_{\text{early}}\\ &= \mathrm{Var}(\{\mathrm{F1}_8, \mathrm{F1}_9, \mathrm{F1}_{10}\}) - \mathrm{Var}(\{\mathrm{F1}_0, \mathrm{F1}_1, \mathrm{F1}_2\})\end{aligned}$$

The above quantities for each search policy have been highlighted in 8. Qualitatively, the graphs presented in 3 show that the search policy in E3 configuration with warm starts and scheduled curriculum shows more stable convergence compared to E1 and E2. Similar trends for a smooth convergence were observed for E4 compared to E1, E2, or E5. However, in our analysis with limited runs, we didn't notice any hints of the search benefiting from using a composite objective as opposed to just F1 (E4 vs E3).

Algorithm Diversity Analysis We characterize each algorithm by a pipeline tuple consisting of prepossessing method, embedding type, and association logic where each component is assigned to a coarse category based on the implemented operations:

Metric	E3	E2	E1
Δ_{recovery} ($\uparrow$)	0.04 ± 0.053	-0.041 ± 0.137	0.013 ± 0.025
Δ_{avg} ($\uparrow$)	0.21 ± 0.09	-0.033 ± 0.058	0.021 ± 0.036
Δ_{Var} ($\downarrow$)	-0.036 ± 0.025	-0.0026 ± 0.011	-0.0003 ± 0.0005
Diversity ($\uparrow$)	9.67 ± 0.47	8.67 ± 1.89	4.33 ± 0.47

Table 8: Early–late curriculum deltas and diversity metrics. Positive Δ_{recovery} and Δ_{avg} indicate improved late-stage performance, while negative Δ_{Var} indicates increased stability.

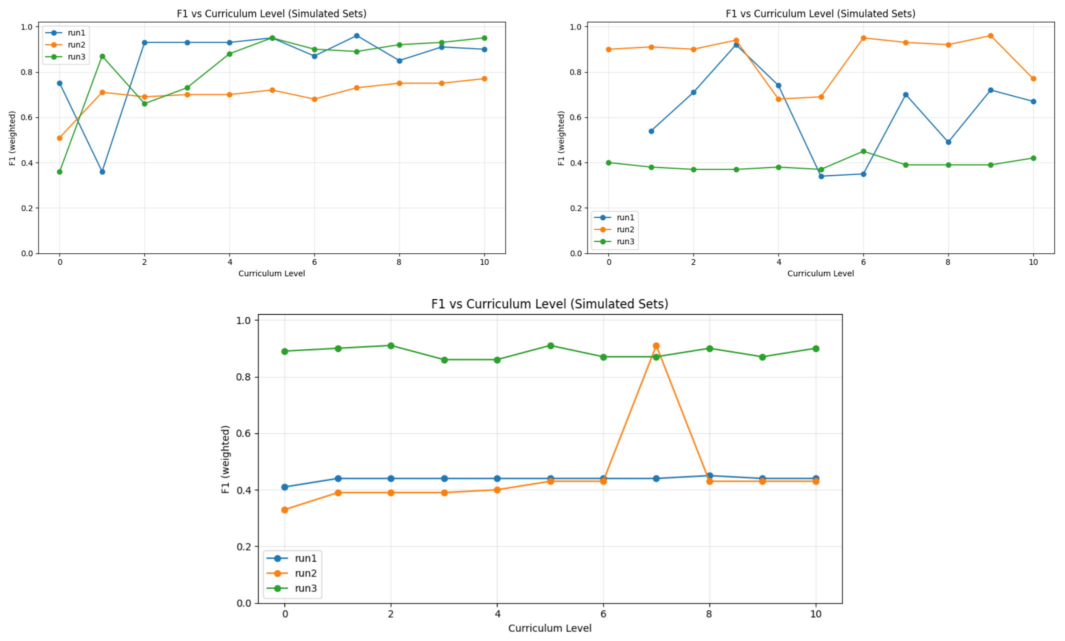


Figure 3: Evolution of F1 on held out synthetic set over iterations under different evolutionary configurations (Refer Table 1). Clockwise from top-left: **Top-Left**: Configuration E3 **Top-Right**: Configuration E2. **Bottom**: Configuration E1. E3 seems to have a more stable convergence overcoming poor, brittle heuristics in the initial stages.

Preprocessing: noise reduction (e.g., median / Wiener / Gaussian), normalization (min-max or z-score), log scaling, and resizing method.

Embedding: DCT / PCA / CNN (ResNet, ImageNet, etc) / flattened spectrogram / simple statistics.

Association Logic: Graph/DSU clustering with TDoA feasibility; thresholding method (percentile, Otsu, or GMM); channel-conflict enforcement.

For evaluating algorithm diversity we look at the number of unique tuples discovered over the sampled algorithms (11 levels/bins) which are reported in Table 8. Configurations E2 and E3 with warm starts show significantly higher diversity in the discovered algorithms compared to E1.

6 Conclusion

We formulated multi-channel call association as an agentic program synthesis problem, using simulation-based evaluation to discover constraint-aware association pipelines without relying on manually annotated real-world datasets during synthesis. Across held-out synthetic regimes, the discovered algorithms generalize to unseen simulator settings and show meaningful transfer to real whale call datasets, with lightweight Optuna tuning further improving real-world performance under domain shift. These results indicate the feasibility of scalable, automated multi-sensor association via simulation-driven agentic search, and motivate future work on improving simulator fidelity and adaptive regime scheduling.

Limitations. Our preliminary evaluations are based on a single LLM backbone and search policy (MCTS). It remains to be seen whether the observed improvements hold under other configurations. In addition, our evaluations rely on a manually specified curriculum of simulation regimes. While this enables scientists to steer the synthesis toward regimes of interest, it also introduces an hyperparameter that may potentially require algorithmic understanding. Future work could reduce this burden and improve robustness by introducing adaptive hyperparameter-selection strategies.

References

Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019. URL `https://arxiv.org/abs/1907.10902`.

Whitlow WL Au and Mardi C Hastings. *Principles of marine bioacoustics*, volume 510. Springer, 2008.

Isabel C Avila, Kristin Kaschner, and Carsten F Dormann. Current global risks to marine mammals: taking stock of the threats. *Biological Conservation*, 221:44–58, 2018.

Paul M Baggenstoss. Separation of sperm whale click-trains for multipath rejection. *The Journal of the Acoustical Society of America*, 129(6):3598–3609, 2011.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th International Conference on Machine Learning (ICML)*, pp. 41–48, 2009.

Mark Chen. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Christopher W Clark, Moira W Brown, and Peter Corkeron. Visual and acoustic surveys for north atlantic right whales, eubalaena glacialis, in cape cod bay, massachusetts, 2001–2005: Management implications. *Marine mammal science*, 26(4):837–854, 2010.

Genevieve E Davis, Mark F Baumgartner, Julianne M Bonnell, Joel Bell, Catherine Berchok, Jacqueline Bort Thornton, Solange Brault, Gary Buchanan, Russell A Charif, Danielle Cholewiak, et al. Long-term passive acoustic recordings track the changing distribution of north atlantic right whales (eubalaena glacialis) from 2004 to 2014. *Scientific reports*, 7(1):13460, 2017.

Erica Fleishman, Danielle Cholewiak, Douglas Gillespie, Tyler Helble, Holger Klinck, Eva-Marie Nosal, and Marie A Roch. Ecological inferences about marine mammals from passive acoustic data. *Biological Reviews*, 98(5):1633–1647, 2023.

Kaitlin E Frasier. A machine learning pipeline for classification of cetacean echolocation clicks in large underwater acoustic datasets. *PLoS Computational Biology*, 17(12):e1009613, 2021.

Marissa L Garcia, Irina Tolkova, Shyam Madhusudhana, Ashakur Rahaman, C Scott Baker, Charles A Mayo, Christine A Hudak, and Holger Klinck. Acoustic abundance estimation for critically endangered north atlantic right whales in cape cod bay, massachusetts, usa. *Endangered Species Research*, 56:101–115, 2025.

Rishi Hazra, Alkis Sygkounas, Andreas Persson, Amy Loutfi, and Pedro Zuidberg Dos Martires. Revolve: Reward evolution with large language models using human feedback. *arXiv preprint arXiv:2406.01309*, 2024.

Tyler A Helble, Glenn R Ierley, Gerald L D'Spain, and Stephen W Martin. Automated acoustic localization and call association for vocalizing humpback whales on the navy's pacific missile range facility. *The Journal of the Acoustical Society of America*, 137(1):11–21, 2015.

Benjamin Hendricks, Janie L Wray, Eric M Keen, Hussein M Alidina, T Aaron Gulliver, and Chris R Picard. Automated localization of whales in coastal fjords. *The Journal of the Acoustical Society of America*, 146(6):4672–4686, 2019.

Kexin Huang, Serena Zhang, Hanchen Wang, Yuanhao Qu, Yingzhou Lu, Yusuf Roohani, Ryan Li, Lin Qiu, Gavin Li, Junze Zhang, et al. Biomni: A general-purpose biomedical ai agent. *biorxiv*, 2025.

Zhengyao Jiang, Dominik Schmidt, Dhruv Srikanth, Dixing Xu, Ian Kaplan, Deniss Jacenko, and Yuxiang Wu. Aide: Ai-driven exploration in the space of code. *arXiv preprint arXiv:2502.13138*, 2025.

Oliver S Kirsebom, Fabio Frazao, Yvan Simard, Nathalie Roy, Stan Matwin, and Samuel Giard. Performance of a deep neural network at detecting north atlantic right whale upcalls. *The Journal of the Acoustical Society of America*, 147(4):2636–2646, 2020.

Sam Lapp, Tessa Rhinehart, Louis Freeland-Haynes, Jatin Khilnani, Alexandra Syunkova, and Justin Kitzes. Opensoundscape: an open-source bioacoustics analysis package for python. *Methods in Ecology and Evolution*, 14(9):2321–2328, 2023.

Tiago A Marques, Len Thomas, Stephen W Martin, David K Mellinger, Jessica A Ward, David J Moretti, Danielle Harris, and Peter L Tyack. Estimating animal population density using passive acoustics. *Biological reviews*, 88(2):287–309, 2013.

David K Mellinger. A comparison of methods for detecting right whale calls. *Canadian Acoustics*, 32(2):55–65, 2004.

Sarah E Nelms, Joanna Alfaro-Shigueto, John PY Arnould, Isabel C Avila, Susan Bengtson Nash, Elizabeth Campbell, Matt ID Carter, Timothy Collins, Rohan JC Currey, Camila Domit, et al. Marine mammal conservation: over the horizon. *Endangered Species Research*, 44:291–325, 2021.

Eva-Marie Nosal. Methods for tracking multiple marine mammals with wide-baseline passive acoustic arrays. *The Journal of the Acoustical Society of America*, 134(3):2383–2392, 2013.

Alexander Novikov, Ngân Vũ, Marvin Eisenberger, Emilien Dupont, Po-Sen Huang, Adam Zsolt Wagner, Sergey Shirobokov, Borislav Kozlovskii, Francisco JR Ruiz, Abbas Mehrabian, et al. Alphaevolve: A coding agent for scientific and algorithmic discovery. *arXiv preprint arXiv:2506.13131*, 2025.

Tessa A Rhinehart, Lauren M Chronister, Trieste Devlin, and Justin Kitzes. Acoustic localization of terrestrial wildlife: Current practices and future opportunities. *Ecology and Evolution*, 10(13): 6794–6818, 2020.

Brendan P Rideout, Stan E Dosso, and David E Hannay. Underwater passive acoustic localization of pacific walruses in the northeastern chukchi sea. *The Journal of the Acoustical Society of America*, 134(3):2534–2545, 2013.

Baptiste Roziere, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Romain Sauvestre, Tal Remez, et al. Code llama: Open foundation models for code. *arXiv preprint arXiv:2308.12950*, 2023.

Elena Schall, Idil Ilgaz Kaya, Elisabeth Debusschere, Paul Devos, and Clea Parcerisas. Deep learning in marine bioacoustics: a benchmark for baleen whale detection. *Remote Sensing in Ecology and Conservation*, 10(5):642–654, 2024.

Ana MM Sequeira, Jorge P Rodríguez, Sarah A Marley, Hannah J Calich, Mirjam van der Mheen, Michelle VanCompernolle, Lucy M Arrowsmith, Lauren R Peel, Nuno Queiroz, Marisa Vedor, et al. Global tracking of marine megafauna space use reveals how to achieve conservation targets. *Science*, 388(6751):1086–1097, 2025.

Yu Shiu, KJ Palmer, Marie A Roch, Erica Fleishman, Xiaobai Liu, Eva-Marie Nosal, Tyler Helble, Danielle Cholewiak, Douglas Gillespie, and Holger Klinck. Deep neural networks for automated detection of marine mammal species. *Scientific reports*, 10(1):607, 2020.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*, 2024.

Renata S Sousa-Lima, Thomas F Norris, Julie N Oswald, and Deborah P Fernandes. A review and inventory of fixed autonomous recorders for passive acoustic monitoring of marine mammals. *Aquatic Mammals*, 39(1):23–53, 2013.

Dan Stowell. Computational bioacoustics with deep learning: a review and roadmap. *PeerJ*, 10: e13152, 2022.

Ludovic Tenorio-Hallé, Aaron M Thode, Marc O Lammers, Alexander S Conrad, and Katherine H Kim. Multi-target 2d tracking method for singing humpback whales using vector sensors. *The Journal of the Acoustical Society of America*, 151(1):126–137, 2022.

Edan Toledo, Karen Hambardzumyan, Martin Josifoski, Rishi Hazra, Nicolas Baldwin, Alexis Audran-Reiss, Michael Kuchnik, Despoina Magka, Minqi Jiang, Alisia Maria Lupidi, et al. Ai research agents for machine learning: Search, exploration, and generalization in mle-bench. *arXiv preprint arXiv:2507.02554*, 2025.

Ildar R Urazghildiiev and Christopher W Clark. Comparative analysis of localization algorithms with application to passive acoustic monitoring. *The Journal of the Acoustical Society of America*, 134 (6):4418–4426, 2013.

Hanchen Wang, Tianfan Fu, Yuanqi Du, Wenhao Gao, Kexin Huang, Ziming Liu, Payal Chandak, Shengchao Liu, Peter Van Katwyk, Andreea Deac, et al. Scientific discovery in the age of artificial intelligence. *Nature*, 620(7972):47–60, 2023.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*, 2022.

Matthew Wijers, Andrew Loveridge, David W Macdonald, and Andrew Markham. Caracal: A versatile passive acoustic monitoring tool for wildlife research and conservation. *Bioacoustics*, 30 (1):41–57, 2021.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36:11809–11822, 2023.

Huan Zhang, Yu Song, Ziyu Hou, Santiago Miret, and Bang Liu. Honeycomb: A flexible llm-based agent system for materials science. *arXiv preprint arXiv:2409.00135*, 2024.

A Scheduled Simulation Regimes

Table 9 summarizes the predefined curriculum used during synthesis. Difficulty increases progressively with call density, array sparsity, and noise levels. Candidate algorithms are evaluated sequentially across levels, and progression to the next level is triggered when performance exceeds a fixed threshold or a maximum iteration budget is reached.

Table 9: Scheduled simulation regimes used during synthesis. Difficulty increases with call density, array sparsity, and noise.

Level	# Calls	# Receivers	Det. Noise	Spec. Noise
0	60	3	50000	0.00
1	60	3	45000	0.05
2	80	4	42000	0.08
3	80	4	38000	0.10
4	100	5	35000	0.12
5	100	5	30000	0.15
6	140	5	28000	0.15
7	140	5	24000	0.15
8	170	7	20000	0.15
9	170	7	17000	0.15
10	200	7	15000	0.15

B Base Search Policy

We use an MCTS-style tree search to choose which candidate program to modify at each step. The policy first creates 5 initial root drafts. After drafting, with probability 0.4 it selects a debuggable buggy leaf (maximum debug depth 3) and applies a debug action; otherwise it selects a non-buggy node for improvement using UCT:

$$\mathrm{UCT}(i) = \hat{Q}_i + c\sqrt{\frac{\ln N_p}{N_i}}, \quad c = 4.0,$$

where $\hat{Q}_i$ is normalized mean utility of the program at node i, N_i is node visits, and N_p is parent visits.

C Problem Description Provided to the Agent

Whale Call Association Framework

Task Overview

Implement the required entry point for the evaluator:

```
def associate_calls(detections) -> list[str]:
    ...
```

This function should use **arrival times** and **spectrograms of whale calls** to cluster or group detections across multiple receivers that originated from the **same whale at the same time**.

CRITICAL DESIGN REQUIREMENTS:

1. **Adaptive Parameters:** Your algorithm must automatically adapt parameters based on the data characteristics. DO NOT use hard-coded thresholds or constants. Instead, compute parameters dynamically from: - Distribution of arrival time differences - Statistics of spectrogram similarities - Receiver geometry constraints (TDoA bounds) - Detection density and temporal clustering patterns

2. **Noise Handling and Preprocessing:** Design appropriate preprocessing to handle noise: - Develop and implement noise reduction/filtering techniques - Consider adaptive preprocessing based on detected noise levels - Ensure preprocessing improves robustness without losing signal information - Preprocessing functions should be implemented as helper functions in `associate_core.py` - Handle spectrograms of varying sizes robustly so concatenation/stacking works without shape errors

3. **Spectrogram Embeddings:** Extract meaningful feature representations from spectrograms for similarity computation: - Load spectrograms from '.npy' files - Apply noise preprocessing/denoising before feature extraction - Choose any embedding strategy that generalizes well; tune

adaptively based on observed performance rather than fixed recipes - Ensure embeddings can be computed on consistently shaped inputs (after your chosen resizing/padding scheme) - Compare embeddings with cosine/Euclidean distance or another suitable similarity metric

Input Format

`detections` — a `List[dict]`, one dictionary per detection, with the following keys:

- **Channel** (int): Receiver index (1-indexed) - **ArrivalTime** (float): Arrival time (seconds) - **SxxPath** (str): Path to the '.npy' spectrogram file

From the inputs, only use `ArrivalTime`, `SxxPath`, and `Channel`. Other fields should not be used.

The number of receivers varies across curriculum levels (3–7 receivers). Load receiver positions from `data/receivers.csv`.

Output Format

Return a list of group labels (strings), one per detection.

The evaluator compares predicted groupings to ground-truth labels using a **Hungarian-aligned F1 score**.

- Group label values do not need to match ground truth. - Every detection must receive a valid label.

Constraints

- At most one detection per channel in each group - Group size can be 1 to N receivers - Favor groups that are: - TDoA-feasible given receiver geometry - Similar in spectrogram embeddings - Prioritize TDoA feasibility first; use spectral cues as secondary evidence

Design Intent

Groups must obey **TDoA invariances**:

Boundedness:

$$-\frac{\|r_i - r_j\|}{c} \leq \Delta t_{ij} \leq \frac{\|r_i - r_j\|}{c}$$

Consistency:

$$\Delta t_{ij} + \Delta t_{jk} + \Delta t_{ki} \approx 0$$

Prefer groups that are mutually consistent and close in spectrogram feature space.

Receiver Geometry

Receiver positions are provided in `data/receivers.csv` with columns: - **receiver_id** - **x** - **y**

Load receiver positions as:

```
import pandas as pd
import numpy as np
from pathlib import Path

receivers_path = Path(__file__).parent / "data" / "receivers.csv"
receivers_df = pd.read_csv(receivers_path, index_col="receiver_id")
receivers = receivers_df[["x", "y"]].values
c = 1500.0
```

Evaluation Metrics

1. **Weighted F1 Score** (primary metric): Hungarian-aligned F1 score after optimal label matching and composition with secondary metrics. 2. Diagnostics: - Singleton fraction - Mean group size - Duplicate channel violations - TDoA violation rate

Objective: **maximize weighted F1 score**.

Evaluation Pipeline

- `associate_calls(detections)` is executed with a 1000-second timeout - Output written to CSV with columns `[ArrivalTime, Group]` - Evaluation performed after Hungarian alignment

Available Packages

`numpy, pandas, torch, torchvision, scipy, sklearn`, and standard library packages. GPU available if present.

Performance Expectations

Primary goal: maximize F1 score through adaptive, constraint-aware association. **Secondary goal:** computational efficiency.

D Real World Dataset Description

This dataset consists of North Atlantic right whale (*Eubalaena glacialis*; henceforth NARW) vocalizations as described in Garcia et al (2025) Garcia et al. (2025). NARW are critically endangered, necessitating effective monitoring to inform conservation efforts. In particular, the "upcall" vocalization is commonly produced by NARW, and thereby frequently used as an indicator of presence Mellinger (2004); Davis et al. (2017). The multichannel audio was recorded by an array of five Marine Autonomous Recording Units (MARUs) positioned within Cape Cod Bay, Massachusetts, a known foraging and resting site for NARW Clark et al. (2010). As upcalls span a range from about 50 to 350 Hz, a sampling rate of 2 kHz was used for recording. We leveraged the analysis pipeline described in Garcia et al. (2025) to construct a realistic, physically-feasible ground-truth dataset for association. Upcalls were first detected by the automated classifier, associated across five channels based on TDoA, and all matches were manually screened to ensure correct association. Next, the associated sets were used to construct realistic datasets with varying call densities by randomly sampling an emission time while maintaining relative arrival times across channels. In total, the dataset consists of 2025 audio samples representing 405 unique calls across four days (2019-02-20, 2019-03-05, 2019-04-07, 2019-04-25). After random sampling of emission times, variants of the dataset spanned from a duration of 400 seconds (60 calls / minute) to 12800 seconds (about 2 calls/minute). The start times, end times, and magnitude of the spectrograms of all audio samples (computed with a window size of 256 samples, or 0.064 seconds) were provided as input to the agent.