Enhancing MAP-Elites with Multiple Parallel Evolution Strategies

Manon Flageat*
manon.flageat18@imperial.ac.uk
Imperial College London
London, U.K.

Bryan Lim*
bwl116@ic.ac.uk
Imperial College London
London, U.K.

Antoine Cully
a.cully@imperial.ac.uk
Imperial College London
London, U.K.

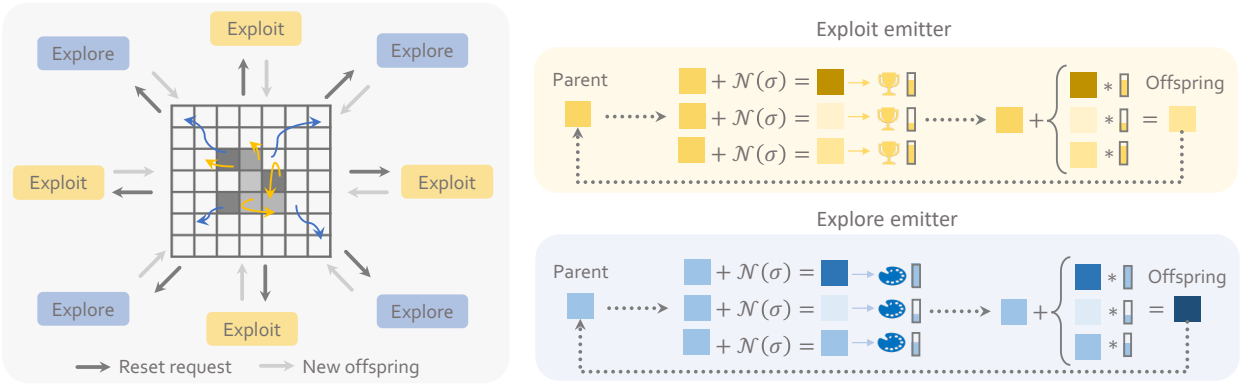


Figure 1: MEMES maintains multiple independent ES emitters to update a MAP-Elites archive. Part of the emitters optimise for the task fitness (exploit emitter) and the rest for the novelty score (explore emitter). Each emitter can be independently reset based on its ability to improve the archive; when reset, an emitter restarts from a uniformly sampled solution of the archive.

ABSTRACT

With the development of fast and massively parallel evaluations in many domains, Quality-Diversity (QD) algorithms, that already proved promising in a large range of applications, have seen their potential multiplied. However, we have yet to understand how to best use a large number of evaluations as using them for random variations alone is not always effective. High-dimensional search spaces are a typical situation where random variations struggle to effectively search. Another situation is uncertain settings where solutions can appear better than they truly are and naively evaluating more solutions might mislead QD algorithms. In this work, we propose MAP-Elites-Multi-ES (MEMES), a novel QD algorithm based on Evolution Strategies (ES) designed to exploit fast parallel evaluations more effectively. MEMES maintains multiple (up to $\sim$ 100) simultaneous ES processes, each with its own independent objective and reset mechanism designed for QD optimisation, all on just a single GPU. We show that MEMES outperforms both gradient-based and mutation-based QD algorithms on black-box optimisation and QD-Reinforcement-Learning tasks, demonstrating its benefit across domains. Additionally, our approach outperforms sampling-based QD methods in uncertain domains when given the same evaluation budget. Overall, MEMES generates reproducible solutions that are high-performing and diverse through large-scale ES optimisation on easily accessible hardware.

*Both authors contributed equally to this research.

This work is licensed under a Creative Commons Attribution International 4.0 License.
GECCO '24, July 14–18, 2024, Melbourne, VIC, Australia
© 2024 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-0494-9/24/07.
https://doi.org/10.1145/3638529.3654089

CCS CONCEPTS

• **Computing methodologies** → **Evolutionary robotics**.

KEYWORDS

Quality-Diversity, Evolution Strategies, Neuroevolution

ACM Reference Format:
Manon Flageat, Bryan Lim, and Antoine Cully. 2024. Enhancing MAP-Elites with Multiple Parallel Evolution Strategies . In *Genetic and Evolutionary Computation Conference (GECCO '24), July 14–18, 2024, Melbourne, VIC, Australia.* ACM, New York, NY, USA, 16 pages. https://doi.org/10.1145/3638529.3654089

1 INTRODUCTION

Advances in Quality-Diversity (QD) have highlighted the benefit of diversity in optimisation. Maintaining a collection of diverse and high-performing solutions has proven beneficial across a wide range of applications, from robotics [10] to design [23] and content generation [26]. QD algorithms help improve exploration during optimisation [5, 14], find better-performing solutions by maintaining stepping stones [24, 36], and facilitate rapid adaptation to unknown downstream tasks [10].

Previously, a large number of evaluations in QD algorithms implied costly compute resources as evaluations had to be parallelised across many CPU devices in most applications, causing a major bottleneck. However, advances in optimisation and machine learning have been facilitated by advances in compute and hardware acceleration. Evolutionary Algorithms, population-based approaches and QD algorithms are not to be outdone, and are shown to be

ideally positioned to take advantage of these advances. Recent libraries such as EvoJax [43], QDax [4] and EvoSax [30] allow a speed-up in run-times by several orders of magnitude thanks to the use of modern hardware accelerators such as GPUs or TPUs. The vectorisation of many computational operations on these devices leads to high parallelisation of evaluations. The development of fast highly-parallel simulators and domains [3, 22, 28, 35, 41, 43] also enables these advances to be applied in a wide range of both simple and complex applications. Lim et al. [33] demonstrate that QD algorithms, and in particular MAP-Elites, scale extremely well to massive parallelisation of evaluations. Increasing the batch size at each generation from $\sim 10^2$ to $\sim 10^5$, reduced the runtime of MAP-Elites by a few orders of magnitude ($\sim 100\times$) without altering the performance, all on a single, customer-grade GPU.

However, naive random variations might not be the most effective method to utilise a large number of parallel evaluations. For example, they remain inefficient and ineffective optimisers in domains that have high-dimensional search spaces. Methods proposed to overcome this limitation include replacing or augmenting the random variations of vanilla QD algorithms with other optimisers, such as policy gradients (PG) [37, 39] or natural gradients in the form of Evolution Strategies (ES) [7, 21]. ES-based QD approaches are demonstrated to be powerful and they can be applied more generally as they are black-box optimisers. However, they require many samples to estimate gradients and become only practical and competitive to PG-based algorithms with access to large numbers of samples and evaluations [42].

Another case where random variations also struggle is in uncertain domains where naive single-sample estimates of solutions can be "lucky" and not reproducible, leading to a loss in true performance [12, 17, 18, 29]. This problem setting is also referred to as Uncertain Quality-Diversity [18]. Similarly, solutions to this problem commonly revolve around taking more samples of the same solution to obtain better estimates [12, 29] or other optimisation schemes which provide and demonstrate more implicit stability [15, 17].

Both these cases are examples of settings where more effective methods than random variations exist but require a large number of evaluations, making them ideal settings for algorithmic innovation through the availability of large-scale parallel evaluations. Ideally, we want methods that (1) are black-box optimisers so that they can be generally used across domains, (2) rely on efficient optimisers, to be effective in both low and high-dimensional domains, (3) obtain reproducible and reliable solutions in uncertain domains, and (4) can benefit and scale well with parallelisation, to leverage and maximise utilisation of more powerful hardware being developed. It is important to note here that the use of the term scale does not just refer to the use of more compute or more devices. Instead, it also refers to better compute devices which allow for greater speed and parallelisation even on just a single easily-accessible and commonly available GPU device.

In this work, we address this by proposing MAP-Elites-Multi-ES (MEMES), an ES-based QD algorithm that exploits large-scale parallelisation to improve the search for quality and diversity. MEMES maintains multiple (up to $\sim$ 100) parallel and independent ES-processes each with its own objective, also called emitters, all on a single GPU device by leveraging tensorisation and parallel evaluations. Exploration and exploitation objectives are distributed between the parallel ES emitters, which together simultaneously perform optimisation by generating new solutions.

We compare the performance of MEMES against 10 baseline QD algorithms and demonstrate that it outperforms them across 4 different tasks. This includes beating state-of-the-art PG-based QD algorithms such as PGA-ME in high-dimensional and uncertain QD-RL neuroevolution tasks. In uncertain domains, solution archives from MEMES prove as reproducible as purpose-built sampling-based approaches that handle uncertain settings well [17, 18] while obtaining better performance, demonstrating the importance of effectively using samples and carefully considering them for addition. Finally, we provide an extensive ablation of the main mechanisms of MEMES. These results make MEMES a promising QD approach in settings where very large numbers ($\sim 10^5$) of evaluations are available at each generation. Moreover, MEMES presents an approach which enables powerful and large-scale ES optimisation on just a single GPU.

2 BACKGROUND AND RELATED WORK

2.1 Quality-Diversity

Quality-Diversity (QD) [11] is an optimisation approach that seeks to find a collection of diverse and high-performing solutions to a problem. In the QD setting, the diversity of solutions is defined according to a set of dimensions of interest referred to as features. Thus, each solution has an attributed fitness f, quantifying its quality; and a feature d, quantifying its novelty with respect to other solutions. Most recent QD approaches rely on one of the two main QD algorithms: MAP-Elites [36] or Novelty-Search with Local Competition [32]. Our work builds upon MAP-Elites.

2.1.1 ***MAP-Elites.*** MAP-Elites (ME) [36] is a common and popular QD algorithm. ME keeps an archive $\mathcal{A}$ of elite solutions encountered during optimisation. To do so, it discretises the feature space into equally spaced grid cells, and keeps the highest fitness solution encountered in that feature cell during optimisation. The algorithm follows the simple procedure of (i) selection from the archive $\mathcal{A}$, (ii) update of selected solutions via perturbations (i.e. mutations), (iii) evaluation to obtain the corresponding fitness f and feature d (iv) update of archive $\mathcal{A}$ by attempting to add solutions. A new solution is added to the archive $\mathcal{A}$ if it occupies a new cell (i.e. is novel) or if it is better performing than the solution in an occupied cell. This way the archive $\mathcal{A}$ incrementally improves as this optimisation procedure is repeated. In this paper, we use the notion of emitter for ME introduced in [21]. An emitter is the generic name given to a procedure used to generate new offspring to be added to the archive $\mathcal{A}$. It corresponds to steps (i) and (ii) above.

2.1.2 ***Uncertain QD.*** Uncertain QD (UQD) refers to the setting where the fitness and feature of solutions are no longer constants but instead are distributions denoted as $f \sim \mathcal{D}_f$ and $d \sim \mathcal{D}_d$ [15, 18]. In UQD domains, solutions can be "lucky" during an evaluation (i.e. outlier under the distribution) resulting in a fitness or features that make them appear more performing or more diverse. Due to their elitism, vanilla QD algorithms prioritise these "lucky" solutions over genuinely diverse or high-performing ones and thus return under-performing archives. Thus, core challenges in the UQD setting are (i) performance estimation (ii) optimising for solutions that

are reproducible [18]. (i) deals with correctly evaluating the true performance and novelty of a solution while (ii) tries to identify and prioritise solutions that exhibit consistent performance under uncertainty, also referred to as reproducible solutions. The most common UQD approach is ME-Sampling [12, 18, 29]. It re-samples each solution N times, and uses the average of the N reevaluations to better estimate its fitness and feature before addition to the archive. This helps to prevent illusory elitism. Our approach implicitly helps with both UQD challenges without compromising on strong QD optimisation by using samples more effectively.

2.1.3 ***Hardware-accelerated QD.*** Recent advances in hardware acceleration have led to new tools that run on devices such as GPUs and TPUs. This can drastically reduce the evaluation time for optimisation algorithms. Examples of such tools are highly-parallel simulators like Brax [22] and Isaac [35], which speed up robotic simulations. These tools led to new QD libraries such as QDax [33], that allow 10 to 100 times more evaluations per generation in parallel within the same amount of time. Using these tools, Lim et al. [33] demonstrated that the performances of ME are robust to large increases in the batch-size value. This drastically speeds up the run-time of QD algorithms by $\sim 100\times$ and opens the door to promising future applications and algorithmic innovations.

2.2 Evolution Strategies

Evolution Strategies (ES) are black-box optimisation algorithms inspired by evolution where a parameterised distribution over solutions is updated in the direction of higher fitness solutions [2, 47]. The different ES are classified according to their sampling and update procedure of the parameterised distribution. We focus on the ES introduced by Salimans et al. [42] for its ability to operate effectively on high-dimensional RL tasks. This algorithm is also commonly referred to as OpenAI-ES but we refer to it as ES for brevity from this point onwards. ES uses the natural gradient approximated from samples to update the parameters of the search distribution. The distribution is represented as an isotropic multivariate Gaussian distribution $\mathcal{N}(\theta_\mu, \sigma^2 I)$, where θ_μ and σ^2 are the mean and variance of the distribution. In this ES variant, σ remains constant and is a fixed hyperparameter. Therefore, the distribution is only parameterised by its mean value θ_μ. A batch or population of N search points are sampled from this parameterised distribution $\theta_i = \theta_\mu + \sigma\epsilon_i$ where $\epsilon_i \sim \mathcal{N}(0, I)$. Each sampled point θ_i is then evaluated on the objective function F and used to estimate the search gradient to update the parameters of the distribution:

$$\nabla E_{\theta\sim\mathcal{N}(\theta_\mu,\sigma^2 I)}[F(\theta)] \approx \frac{1}{N\sigma}\sum F(\theta_i)\epsilon_i \tag{1}$$

We give the pseudo-code for one ES optimisation step in Appendix A. This procedure is then repeated iteratively to improve the parameters of the search distribution. Following Salimans et al. [42], we also perform fitness shaping in the form of rank normalisation.

2.2.1 ***Novelty-Search with Evolution Strategies.*** The variant Novelty-Search ES (NS-ES) [8] also uses ES, described in the previous section, but augments the objective term to be able to handle sparse and deceptive rewards to encourage exploration in such settings. To do so, NS-ES uses a novelty score as the objective F for the ES. Taken from Novelty Search [31], the novelty score of a solution θ corresponds to the average distance in feature d to its K-nearest neighbours in a novelty archive $\mathcal{N}$, containing all previously evaluated solutions: $novelty(\theta) = \frac{1}{K}\sum_{k=1}^{K} \|d_\theta - d_k\|_2$.

2.3 Gradient-augmented QD

Despite the simplicity and success of conventional QD algorithms, they have generally been shown to be efficient in low-dimensional search spaces but suffer in high-dimensional ones due to the sample-inefficiency of their GA-based mutation [7]. To be able to function in applications such as neuroevolution, where the size of the parameter space is easily in the thousands, more efficient optimisation procedures were needed over the standard GA mutations used, such as gradient-based optimisation methods, that are generally known to be more efficient optimisation algorithms.

Our work builds on MAP-Elites ES (ME-ES) [7], detailed in Section 2.3.1, which uses ES and maintains an archive of elites. Other QD algorithms that use ES also exist such as Novelty Search-Reward ES (NSR-ES) and Novelty Search-Reward Adaptive ES (NSRA-ES) [8] which also uses ES [42] but instead optimises for a weighted sum of the task reward and the novelty score as the objective. Covariance Matrix Adaptation ES (CMA-ES) [27], an alternative ES approach, has also been combined with ME resulting in CMA-ME [21]. The space and computational time complexity prevents CMA-ES and hence CMA-ME from being used on high-dimensional search spaces. However, recent extensions [44, 45] try to address this challenge by approximating the covariance matrix.

Another class of gradient-based QD methods tailored to Markov Decision Processes (MDP) and Deep Reinforcement Learning (DRL) problems [1] are actor-critic based methods such as Policy Gradient Assisted ME (PGA-ME) [37] and QD-PG [39]. They utilize policy-gradients (PG) from DRL which rely on training a critic network to apply variations to parent policies. They have shown to be effective in the QD-RL setting but require training an additional critic network. In the following, we refer to QD approaches using NES mutations as ES-based QD. Similarly, we refer to approaches mixing GA and PG as PG-based QD.

2.3.1 ***MAP-Elites with Evolution Strategies.*** MAP-Elites ES (ME-ES) [7] combines the optimisation procedure of ES [42] with the maintenance of an archive in ME [36]. It uses the ES optimisation procedure in two separate sequential modes; "exploit" and "explore". The "exploit" mode is used with the task fitness as the objective F for the ES while the "explore" mode uses a novelty score as the objective F. For the "explore" mode, ME-ES maintains a novelty archive $\mathcal{N}$ (see Section 2.2.1), separate and different from the archive of elites $\mathcal{A}$. ME-ES starts by selecting a solution from the archive of elites $\mathcal{A}$ and applies one of the modes (e.g. "exploit") for N_{optim} generations. Each generation corresponds to one gradient step (update of the distribution). At every generation, the updated mean of the distribution is considered for addition to the archive $\mathcal{A}$. After N_{optim} generations, ME-ES selects another solution from the archive and applies the other mode (e.g. "explore"). ME-ES alternates between the "exploit" and "explore" modes, stopping each mode after N_{optim} generations (in the original ME-ES, $N_{optim} = 10$).

After stopping a mode, a new solution is selected from the archive $\mathcal{A}$ following a biased selection procedure rather than a random uniform selection commonly used in ME. This procedure aims

to select solutions that are already high-performing and can be improved further. It selects either one of the highest performing for exploit mode or one of the most novel solutions for explore mode. The pseudo-code for ME-ES is provided in Appendix A.

3 METHODS

In this paper, we introduce MAP-Elites-Multi-ES (MEMES), which leverages tensorisation and advances in hardware accelerated QD to maintain multiple independent ES workers in parallel each with its own independent objective and reset mechanism designed for QD optimisation. In the following we use 32 parallel ES but we show in Appendix C that MEMES also scales to 128 parallel ES. These modifications address commonly found limitations of the ME-ES algorithm (see Sec. 2.3.1) such as slow and inefficient optimisation due to the large number of samples and low batch-size.

3.1 Parallel ES: from sequential to simultaneous

MEMES leverages the use of vectorisation and fast parallel evaluations [22] to be able to run hundreds of independent ES in parallel. Each independent ES represents a separate emitter that generates new solutions. In contrast, in ME-ES, a single thread of ES at each generation sequentially alternates between the exploitation of the task fitness objective and of the novelty objective (see Section 2.3.1). To effectively do this, we reduce the number of samples M (i.e. population size) used by each ES process to estimate the gradient. In ME-ES [7], a single ES update (i.e. gradient step) uses $M = 10,000$ samples. Here, we use $M = 512$ samples per ES emitter as we observe an insignificant increase in performance when using larger values of M beyond 512 in many of our experiments. Thus, for comparison, to generate 32 offspring (the value used in this work), MEMES uses 16384 samples per generation while ME-ES uses 10000 samples to generate 1 offspring.

This parallelisation allows multiple independent emitters to be run together in order to perform the explore and exploit modes simultaneously. In MEMES, a proportion $p_{exploit}$ of the emitters is dedicated to the exploitation of the task fitness objective, and a proportion $1 - p_{exploit}$ is dedicated to the optimisation of the novelty score. While our experiments only use these objectives for comparison, other objectives such as improvement [21], curiosity [34, 38] and others can also be used at the same time [9]. This enables many objectives to be optimised simultaneously and in parallel. This interplay could lead to the generation of new and interesting solutions and greater efficiency and more effective search.

3.2 Dynamic emitter reset

An ES-emitter reset refers to selecting a new solution to restart an ES optimisation process from. This is done by selecting a solution from the ME archive $\mathcal{A}$. In ME-ES, the ES-emitter is reset every fixed number of generation N_{optim} to switch between explore and exploit mode. This is imposed by the constraint of having only a single emitter that needs to switch between the two modes. However, this might impair the optimisation process as solutions with high evolvability, dwelling in promising areas of the search space, might not be fully exploited. Additionally, as the optimiser state is reset with the emitter itself, even if an evolvable solution is later re-sampled, it would still take a few generations to be able to use it meaningfully. Our experimental results (shown in the following sections) indicate that some emitters can generate promising individuals for hundreds of generations, beyond the optimisation lifetime imposed in ME-ES ($N_{optim} = 10$).

Algorithm 1: MEMES proposed in this work.

Inputs: sample size N, number of generations N_{gen}, number of emitters $N_{emitter}$, proportion of exploit emitter $p_{exploit}$
Initialisation: archive of elites $\mathcal{A}$ and novelty-archive $\mathcal{N}$
// Set objective for each emitter
$n = N_{emitter} * p_{exploit}$
foreach $e = 1, ...n$ **do** $F_e \leftarrow$ fitness_objective()
foreach $e = n, ...N_{emitter}$ **do** $F_e \leftarrow$ novelty_objective($\mathcal{N}$)
// Main loop
for $g = 1, ...N_{gen}$ **do**
  do in parallel for $e = 1, ...N_{emitter}$
    if *require_reset* **then**
      $\theta \leftarrow$ uniform_selection($\mathcal{A}$)
      $S = 0$
      require_reset = $False$
    $\tilde{\theta} \leftarrow$ ES_step(F_e, N, θ)
    $\mathcal{A}, \mathcal{N} \leftarrow$ update_archives($\tilde{\theta}$)
    if *added_to_archive*($\mathcal{A}, \tilde{\theta}$) **then**
      $S = 0$
    else
      $S = S + 1$
      if $S > S_{max}$ **then** require_reset = $True$
return $\mathcal{A}$

MEMES introduces a reset strategy where each independent emitter can be reset separately to fully maximise having simultaneous explore and exploit emitters in parallel. We propose dynamically and automatically resetting the exploit and explore emitters based on their usefulness to the QD optimisation. Each ES emitter generates one offspring $\tilde{\theta}$ per generation (gradient step), which is a candidate to be added to the archive $\mathcal{A}$. Each time a candidate is not added to $\mathcal{A}$, we increase a stale counter S; each time the candidate is added to $\mathcal{A}$, we reset S to 0. This introduces a new hyperparameter: the stale budget S_{max}. If S exceed S_{max}, meaning if the emitter does not generate promising offspring for more than S_{max} consecutive generations, the emitter is reset. This mechanism allows MEMES to fully leverage lineages of promising solutions, and automatically determine good optimisation paths. We demonstrate the benefits of this mechanism in an ablation study in the experimental section.

When resetting an ES emitter, a random solution is selected with a uniform probability from the archive as a new starting point of the ES process. This is done to reduce bias on any particular task and increase the probability of utilising stepping stones from unexpected feature space regions [36]. More details regarding this is available in Appendix A.

3.3 Improving the explore emitter

Different QD algorithms use different strategies for exploration. The most commonly used strategy is random mutations (GAs) [36, 37]. However, as motivated, solutions from these operators might not be the most effective way to leverage access to a large number of evaluations. Other approaches that rely on explicit novelty optimisation also exist. QD-PG [39], which only applies in RL-MDP settings, follows a state-novelty gradient. CMA-MEGA [20] follows feature gradients in whitebox optimisation settings, but it can also be applied to problems where gradients can be approximated [45]. In

MEMES, following NS-ES, NSR-ES [8] and ME-ES [7], we compute approximated-gradients using ES to move in the direction of high novelty-score. As explained in Section 2.2.1, computing the novelty score requires a novelty-archive $\mathcal{N}$, consisting of all previously evaluated solutions, regardless of how they perform. Note that this is separate and different to the archive of elites $\mathcal{A}$, which contains only the best-performing solutions (see Section 2.2.1). Maintaining this novelty archive $\mathcal{N}$ is extremely memory-intensive, especially in highly-parallel settings, and requires infinitely increasing computation as the archive expands linearly with generations. Hence, in MEMES, we propose and use a first-in-first-out (fifo) novelty archive to overcome this challenge. The fifo novelty archive stores all evaluated offspring until it reaches full capacity. Then, each time a new solution enters the fifo novelty archive, it replaces the oldest solution. We also explore alternative approaches to compute novelty scores but demonstrate experimentally that this simple approach works the most effectively while alleviating memory usage. We consider the following approaches:

- **Novelty-archive-fifo-ES explore (Ours):** uses the fifo novelty archive described before which alleviates memory congestion.
- **Novelty-archive-all-ES explore:** conventional novelty archive $\mathcal{N}$ that keeps all evaluated solutions. This has computational time and memory limitations but is used as a baseline.
- **Elites-archive-ES explore:** replaces the novelty archive with the QD archive of elites $\mathcal{A}$, when computing novelty score. This requires significantly less memory as only the elite of each niche are kept in $\mathcal{A}$. However, this also leads to a loss of historical and cumulative information about solutions seen before. The QD archive only considers the filling of an empty cell as novel while a novelty archive keeps all solutions seen during search, and accounts for the density of solutions in an area of the feature space when computing novelty. This means that a QD-archive cannot identify when a part of the feature space has been explored many times unsuccessfully.
- **GA:** uses random mutation [46]. Non-archival method which does not require any memory resources at all.

4 EXPERIMENTS

4.1 Experimental Setup

4.1.1 Environments. We evaluate MEMES and baselines on four tasks from QD literature. Two of these tasks (Arm and Hexapod) are black-box deterministic QD benchmarks while the other two tasks (Ant and AntTrap) are neuroevolution RL tasks in the UQD setting. Both neuroevolution tasks consider noise in the initial state and in the dynamics of the rollout induced by the simulator.

- **Redundant Arm (Arm)** is a commonly used as a benchmark QD task [10, 11, 20, 25]. This task aims to learn the smoothest angular-position vectors to reach all (x, y) positions with a 1000-DoF planar robotic arm.
- **Omnidirectional Hexapod (Hexapod)** is a QD benchmark that aims to learn the vectors of a parameterised open-loop controller to walk in every direction with a 18-DoF Hexapod, while minimising its orientation error [6, 13].
- **Unidirectional Ant (Ant)** proposed as part of the QDGym suite [19, 37]. In this neuroevolution task, the algorithms learns closed loop neural network policies for a four-legged Ant robot with randomised initial state. The objective is to find a set of diverse gaits for moving forward as fast as possible.
- **Ant Trap** is a neuroevolution task similar to Ant but consists of deceptive rewards and is categorized as hard-exploration [5, 39]. In Ant Trap, exploiting the task fitness directly would result in getting stuck in the trap present in the environment. In contrast, exploitation of the task fitness in the Ant task is always a good idea. Additionally, in Ant Trap, the policies that are close to the origin will inherently have a low task fitness and not solve the task (i.e. pass the trap) despite contributing to the diversity. To the best of our knowledge, this task has not been considered and evaluated in the UQD setting prior to this work, making it an interesting task.

Detailed descriptions of the fitness and feature (including equations) for each task is provided in Appendix A.

4.1.2 Baselines. In the following, we compare MEMES to a wide range of baselines. We also consider a variant of our approach named MEMES-all, which considers all samples and not just offspring of ES emitters for addition to the archive.

- **ES** [42] from Sec. 2.2 optimises for task fitness.
- **NS-ES** [8] from Sec. 2.2.1 optimises for novelty score.
- **NSR-ES and NSRA-ES** [8] from Sec. 2.3. They use a weighted average of the task fitness and the novelty score as objective.
- **ME** [36] which uses random mutations (GAs). Here, we use the line mutation operator [46]. We consider two variants of ME. ME-128 with a batch size of 128 which is a commonly used value and ME-16384 with a batch size of 16384 which corresponds to the same number of samples used by MEMES at each generation.
- **ME-Sampling** [12] Given that re-evaluation is one of the most common methods in uncertain domains [18], we also consider the ME-sampling method which uses the same 16384 evaluations at each generation but distributed to re-evaluate solutions. Here, each of the solutions is re-evaluated 32 times [16] to obtain a better estimate of f and d before addition.
- **ME-ES** [7] from Sec. 2.3.1 alternating explore and exploit ES.
- **PGA-ME** [37] described in Sec. 2.3. applies policy gradient (PG) updates to half of the selected solutions, in addition to GAs. PGA-ME represents the upper baseline and state-of-the-art for high-dimensional neuroevolution QD-RL tasks such as Ant and AntTrap. However, it can only be used for RL tasks with MDP assumptions.
- **CMA-ME** [21] described in Sec. 2.3. CMA-ME is only used as a baseline for the lower-dimensional optimisation task (i.e. non-neuroevolution). We do not compare to its extension CMA-MEGA with approximated gradients as it demonstrates similar or lower performing results than PGA-ME on QD-RL tasks [45].

It is important to note that ES, NS-ES, NSR-ES and NSRA-ES do not have an archive of elites and do not rely on re-selection from the archive during the entire optimisation process. For comparison, we maintain a passive archive of elites and consider the addition of the mean of the ES at every generation. We open-source our code at https://github.com/adaptive-intelligent-robotics/MEMES.

4.1.3 Metrics. We consider the QD-Score, Max-Fitness, and Coverage metrics [40] when conducting our analysis. In our evaluation, all three metrics provide valuable insights: the effect of explore and

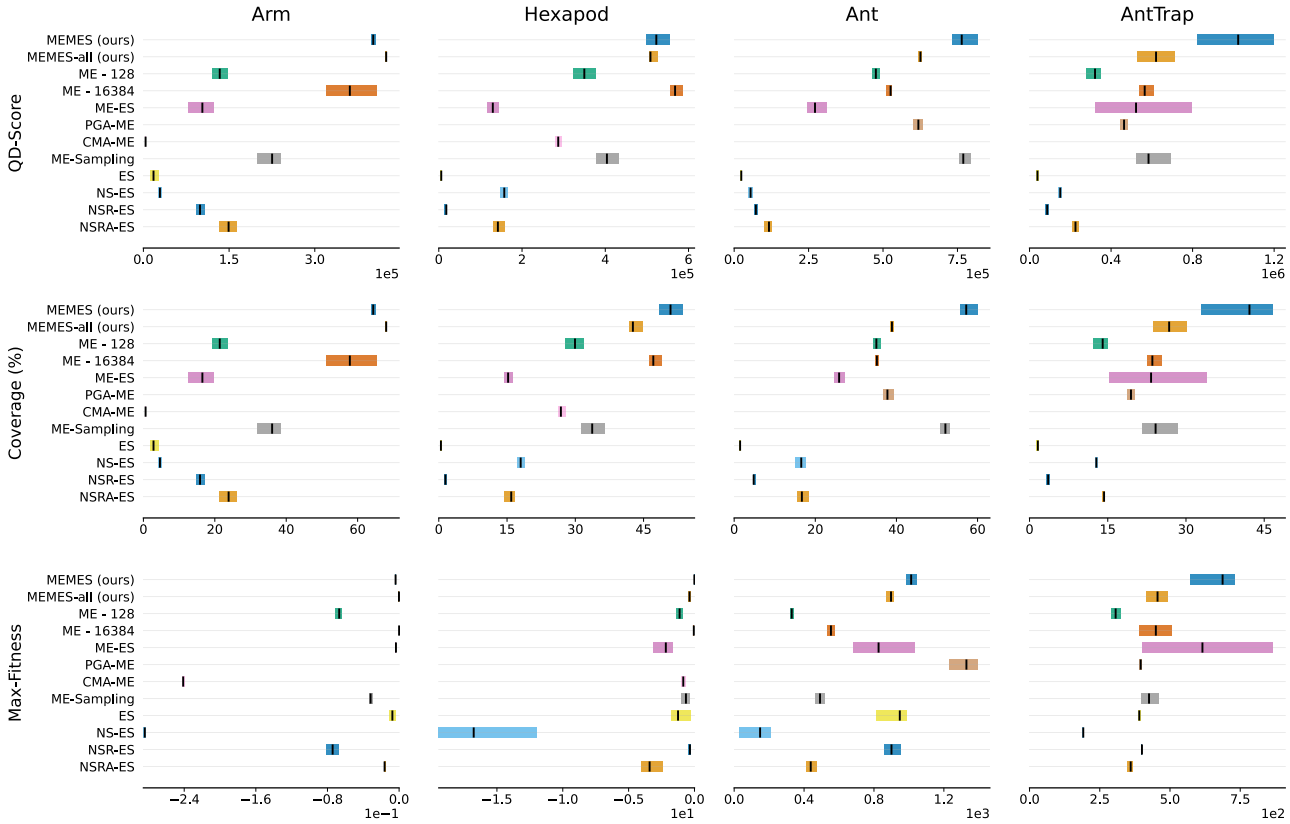


Figure 2: Final QD-Score (top), Coverage (middle) and Max-Fitness (bottom). We report the median and CI over 10 seeds.

exploit emitters can be respectively assessed from the Coverage and Maximum-Fitness scores, while the QD-score evaluates the overall algorithm performance. In the uncertain domains (neuroevolution tasks), we consider the Corrected-metrics [18, 29], computed by re-evaluating each solution in the archive 512 times and adding them to an empty archive. We run Arm, Hexapod and Ant for 2000 generations and AntTrap for 10000 generations. We compute p-values based on the Wilcoxon rank-sum test with Bonferroni correction.

4.2 Overall Performance

Figure 2 shows the performance of MEMES compared to the baselines. MEMES performs better than all baselines on Arm ($p < 0.08$ for QD-Score and Coverage). On the Hexapod, MEMES performs on par with ME-16384 while outperforming all other baselines ($p < 0.03$). Additionally, the MEMES-all variant outperforms MEMES on Arm ($p < 2.10^{-4}$ for QD-Score and Coverage), where the attempted addition of more samples (from the ES) to the archive results in better performance, but performs on par with MEMES on Hexapod. In the Ant neuroevolution task, we observe that MEMES performs better than PGA-ME, which is known to consistently do well on this task [37, 45], in both QD-Score and Coverage ($p < 0.003$) but reaches a lower Max-Fitness. The Max-Fitness of MEMES seems to be constrained by the performance of the ES algorithm as we observe Max-Fitness performance converging to that of conventional ES. This makes sense given that the adaptive emitter reset in MEMES would enable a conventional ES optimisation thread to be present. However, the benefit of parallel explore and exploit ES emitters each following independent gradients in MEMES is still overall evident from the higher Coverage compared to GA-based approaches such as ME and PGA-ME ($p < 0.003$ for Coverage) and also single ES-based approaches such as ES, NS-ES and NSR-ES. In the AntTrap neuroevolution task, MEMES successfully performs the best across all baselines, proving the only approach to solve the task and move around the deceptive trap. Recall that an uncertain version of these neuroevolution tasks are considered and the corrected metrics are used in the comparison.

These results suggest that with access to cheap and large numbers of evaluations, MEMES can be the go-to option across a range of domains. Given the use of vectorisation and parallelisation of evaluations, powerful large-scale search optimisation (MEMES) is accessible and practical to be run in a reasonable time-frame with only a marginal increase in run-time. An analysis of the run-time of MEMES and baselines are provided in Appendix C. Resulting archives of elites on the various tasks can be found in the Appendix C and videos of resulting behaviours are available at: https://sites.google.com/view/memes-qd/home.

4.2.1 Sampling-wise Comparison and Reproducibility. MEMES requires a large number of evaluations to approximate gradients. We evaluate if this could be effective in handling the UQD problem (see Sec. 2.1.2). On the Ant and AntTrap tasks, we observe in Figure 2 that samples used by MEMES are indeed more effective as

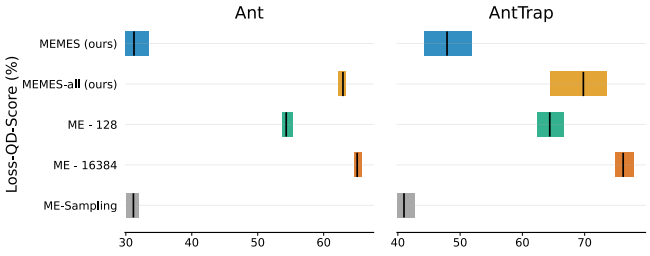


Figure 3: QD-Score loss that quantifies the ability of algorithms to correctly estimate the performance of solutions in UQD setting. We report the median and CI over 10 seeds.

	Arm	Hexapod	Ant	AntTrap
Explore emitter	143.0	362.4	249.5	270.1
Exploit emitter	265.7	337.2	366.5	387.7

Table 1: Average lifespan (in num. of generations) of exploit and explore emitters when using adaptive reset in MEMES.

they result in higher QD-scores than those of ME-16384 and also of ME-Sampling and ME-16384 for AntTrap.

On top of the corrected-metrics, we also further analyse the archive-reproducibility, by evaluating the algorithms on the archive loss metric. The archive loss is defined as the percentage difference before and after computing the corrected archive [15]. In Figure 3, MEMES is shown to have a much lower QD-Score loss than the ME baselines. MEMES losses are comparable to those of ME-Sampling, which is considered a strong baseline for this metric [15], while ME-Sampling has a significantly lower QD-Score on AntTrap. This demonstrates that MEMES archives can reliably maintain its quality and diversity and that MEMES better estimates and reflects the true performance in its metrics. This ability of MEMES to favour solutions that reproducibly solve the task likely leads to it outperforming other approaches on AntTrap, while ME-16384 suffer from elitism and archive loss even with a significant number of iterations, and ME-sampling is crippled from inefficient optimisation. MEMES-all also suffers from a high archive loss as it also considers many "lucky" solutions (from ES samples) for addition causing illusory archive elitism as well. This means that only the offspring of ES emitters are more reproducible and further highlights the importance of carefully and effectively using samples and considering them for addition in uncertain domains.

4.2.2 Effective Local Optimisation. We also evaluate some characteristics of the optimisation paths and behaviour of MEMES. Interestingly, we find in MEMES the ability to perform optimisation locally within or surrounding a cell niche, which is uncommon in QD algorithms. This is an important characteristic in QD as optimising locally in low-performing niches/cells is crucial to ensure high fitness across the entire feature space. This ability was measured using the distance between each parent and its offspring in the feature space. This metric measures whether generated individuals stay within or around the vicinity of the same cell. As the exploration aspect of QD algorithms such as novelty-ES are expected to maximise the distance in feature space, we only evaluate this metric for the task-fitness-maximising aspect of algorithms. For

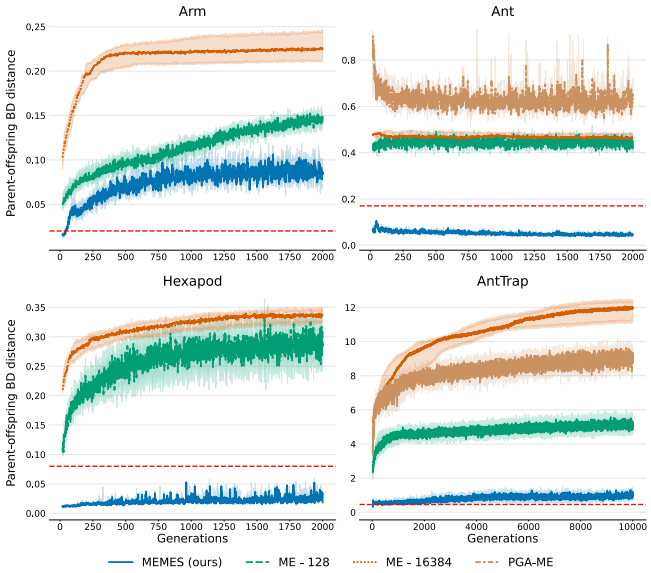


Figure 4: Parent-offspring feature-distance for the exploit-ES emitter of MEMES, the PG emitter of PGA-ME and the GA emitter of ME. The solid line is the median and the shaded areas are the quartiles over 10 replications. We display in red the average width of the cells in the each task.

MEMES, we compute it for the exploit-ES and compare it to the PG variation of PGA-ME, and GA variation of ME.

Figure 4 shows this distance measure plotted across generations of the algorithm. For comparison, we display on the same graph the cell-size for each task (red line). In Ant and Hexapod, the exploit-ES emitters produce offspring solutions after a gradient step that have a distance of less than one cell (below red line). This demonstrates some implicit ability of the emitter to perform local optimisation around a certain niche. This is beneficial in tasks where the task fitness is orthogonal to the feature. In the AntTrap tasks, the goal of the exploit-ES emitter is to move forward as fast as possible and hence the nature of this task does not require such local optimisation. Interestingly, we also observe that the exploit-ES emitter largely results in offspring with much lower distance than PGA-ME and ME. PGA-ME uses policy gradient variations that rely on training a global critic. We hypothesise that this global critic systematically pushes updates in the same direction of the search space, which while can improve the policy might also result in similar features far away from parent solutions. Likewise, the GA variation of ME has no specific mechanism to optimise solutions while remaining locally in a feature region and hence results in offspring further away in the feature space.

4.3 Analysis and ablation of MEMES

In this section, we conduct experiments to study the different components of MEMES and show that each one of them contribute to the effectiveness of the approach.

4.3.1 Simultaneous over sequential. To demonstrate the importance of having simultaneous explore and exploit emitters, we

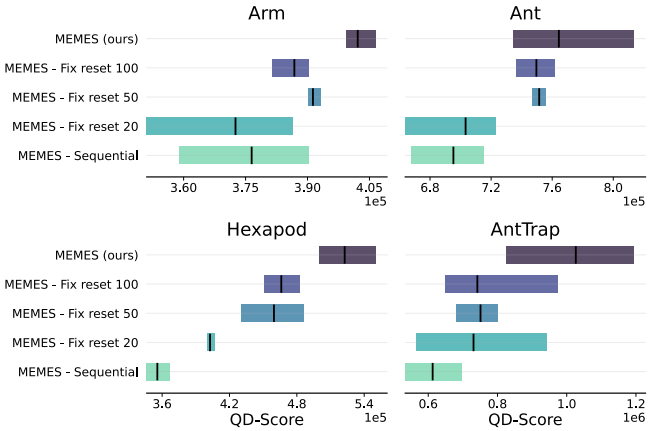


Figure 5: Comparison of the final QD-score for different fixed reset values, compared to the adaptive reset mechanism. We report the median and CI over 5 seeds.

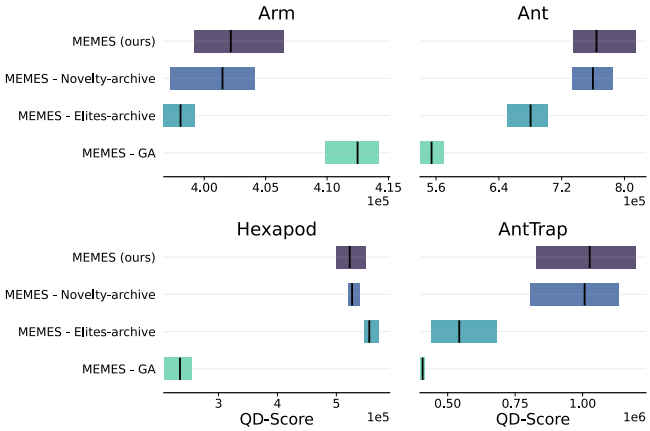


Figure 6: Final QD-score for different novelty-computation mechanisms. We report the median and CI over 5 seeds.

compare against MEMES-Sequential, a variant of MEMES with parallel ES emitters that follows the original ME-ES implementation of alternating between either fully explore or fully exploit modes. The emitters are reset and swapped mode after 10 generations. It is important to note that being sequential necessitates having a fixed number of generations in order to swap between modes. Figure 5 shows that MEMES-Sequential slightly under-performs MEMES with simultaneous explore and exploit (MEMES - Fix reset 10). While the improvement of having simultaneous emitters is marginal, they allow independent ES emitters with adaptive resets.

4.3.2 Adaptive emitter resets. Next, we evaluate the importance of the adaptive reset of the emitters by increasing the number of optimisation steps of ES before resetting the emitter. We observe in Figure 5 that increasing the number of optimisation steps of the ES emitters seems to help (i.e. from 20 to 100). However, the optimal value may vary depending on the tasks. For example, we can observe that there are minimal gains (Hexapod, Ant, AntTrap) or detrimental to overall performance (Arm) when moving from 50 to 100 optimisation steps. Similarly, the optimal value also could depend on the mode of ES used (exploit or explore). On the Ant task, more steps are critical for the exploit-ES for optimal maximum fitness performance while potentially unnecessary for the explore-ES model (coverage metric). Conversely, in the AntTrap, more steps are important for the explore-ES (coverage) while the gain in steps might even be detrimental to exploit-ES at times. Table 1 shows that the adaptive reset mechanism in MEMES can modulate and shows different average lifespans of the ES emitters depending on the task and the mode of the ES. Figure 5 demonstrates that our adaptive mechanism achieves the best on all the metrics across all the tasks without much hyperparameter tuning.

4.3.3 Explore Emitter. Lastly, we study the different options for effective exploration (see Sec. 3.3). For fair comparison, we maintain the same ES-exploit emitter with the adaptive resets to isolate the performance of the different explore-emitter choices. Figure 6 shows the performance of the different explore emitters considered. The fifo novelty archive performs similarly to the full novelty archive whilst being more memory efficient. Interestingly, the archive of elites $\mathcal{A}$ also represents another compute and memory efficient alternative that performs strongly on some tasks (i.e. Hexapod). The decrease in performance in some tasks (especially deceptive) can be attributed to the loss of historical information by only maintaining elites as highlighted in Section 3.3. However, in tasks such as the Hexapod, where this information is less important and relevant, it performs the best. The GA variant performs well on Arm but it clearly under-performs on more complex tasks, highlighting the benefit of the novelty ES emitters.

5 DISCUSSION AND CONCLUSION

In summary, we present MEMES, a QD algorithm that leverages tensorisation and large-scale parallelisation, to maintain multiple independent ES emitters (up to $\sim$ 100) in parallel, each with their own objective on a single GPU. MEMES also introduces an adaptive emitter reset mechanism to manage the parallel optimisation threads and maximise the improvement efficiency of the QD archive. We demonstrate that MEMES outperforms a large variety of existing QD algorithms on a range of high-dimensional neuroevolution tasks and also lower-dimensional optimisation tasks. Importantly, in addition to finding higher-performing and more diverse solutions, we also show that archive solutions generated by MEMES are more reproducible when operating in uncertain domains, making MEMES a more reliable QD algorithm. This highlights the importance of effectively using samples and carefully considering them for addition in uncertain domains.

While MEMES performs consistently well, it is important to remember that this approach requires a large number of evaluations for estimating gradients. This limitation means that our proposed approach is currently effective only in settings where evaluations are cheap and can be parallelised on accelerator devices. However, with powerful compute becoming cheaper, smaller, more widely available and heavy development and movement of environments on accelerated hardware [3, 28, 41], we believe MEMES can be an effective approach across a wide range of applications. Given the flexibility of our algorithm, we hope more maximise the use of many parallel ES processes by extending MEMES to more objectives beyond fitness and novelty objectives in future work.

REFERENCES

[1] Kai Arulkumaran, Marc Peter Deisenroth, Miles Brundage, and Anil Anthony Bharath. 2017. Deep reinforcement learning: A brief survey. *IEEE Signal Processing Magazine* 34, 6 (2017), 26–38.

[2] Hans-Georg Beyer and Hans-Paul Schwefel. 2002. Evolution strategies–a comprehensive introduction. *Natural computing* 1 (2002), 3–52.

[3] Clément Bonnet, Daniel Luo, Donal Byrne, Shikha Surana, Vincent Coyette, Paul Duckworth, Laurence I. Midgley, Tristan Kalloniatis, Sasha Abramowitz, Cemlyn N. Waters, Andries P. Smit, Nathan Grinsztajn, Ulrich A. Mbou Sob, Omayma Mahjoub, Elshadai Tegegn, Mohamed A. Mimouni, Raphael Boige, Ruan de Kock, Daniel Furelos-Blanco, Victor Le, Arnu Pretorius, and Alexandre Laterre. 2023. Jumanji: a Diverse Suite of Scalable Reinforcement Learning Environments in JAX. arXiv:2306.09884 [cs.LG] https://arxiv.org/abs/2306.09884

[4] Felix Chalumeau, Bryan Lim, Raphael Boige, Maxime Allard, Luca Grillotti, Manon Flageat, Valentin Macé, Arthur Flajolet, Thomas Pierrot, and Antoine Cully. 2023. QDax: A Library for Quality-Diversity and Population-based Algorithms with Hardware Acceleration. *arXiv preprint arXiv:2308.03665* (2023).

[5] Felix Chalumeau, Thomas Pierrot, Valentin Macé, Arthur Flajolet, Karim Beguir, Antoine Cully, and Nicolas Perrin-Gilbert. 2022. Assessing Quality-Diversity Neuro-Evolution Algorithms Performance in Hard Exploration Problems. (2022). https://doi.org/10.48550/ARXIV.2211.13742

[6] Konstantinos Chatzilygeroudis, Vassilis Vassiliades, and Jean-Baptiste Mouret. 2018. Reset-free trial-and-error learning for robot damage recovery. *Robotics and Autonomous Systems* 100 (2018), 236–250.

[7] Cédric Colas, Vashisht Madhavan, Joost Huizinga, and Jeff Clune. 2020. Scaling map-elites to deep neuroevolution. In *Proceedings of the 2020 Genetic and Evolutionary Computation Conference*. 67–75.

[8] Edoardo Conti, Vashisht Madhavan, Felipe Petroski Such, Joel Lehman, Kenneth Stanley, and Jeff Clune. 2018. Improving exploration in evolution strategies for deep reinforcement learning via a population of novelty-seeking agents. *Advances in neural information processing systems* 31 (2018).

[9] Antoine Cully. 2021. Multi-emitter map-elites: improving quality, diversity and data efficiency with heterogeneous sets of emitters. In *Proceedings of the Genetic and Evolutionary Computation Conference*. 84–92.

[10] Antoine Cully, Jeff Clune, Danesh Tarapore, and Jean-Baptiste Mouret. 2015. Robots that can adapt like animals. *Nature* 521, 7553 (2015), 503–507.

[11] Antoine Cully and Yiannis Demiris. 2017. Quality and diversity optimization: A unifying modular framework. *IEEE Transactions on Evolutionary Computation* 22, 2 (2017), 245–259.

[12] Antoine Cully and Yiannis Demiris. 2018. Hierarchical behavioral repertoires with unsupervised descriptors. In *Proceedings of the Genetic and Evolutionary Computation Conference*. 69–76.

[13] Antoine Cully and Jean-Baptiste Mouret. 2013. Behavioral repertoire learning in robotics. In *Proceedings of the 15th annual conference on Genetic and evolutionary computation*. 175–182.

[14] Adrien Ecoffet, Joost Huizinga, Joel Lehman, Kenneth O Stanley, and Jeff Clune. 2021. First return, then explore. *Nature* 590, 7847 (2021), 580–586.

[15] Manon Flageat, Felix Chalumeau, and Antoine Cully. 2022. Empirical analysis of PGA-MAP-Elites for Neuroevolution in Uncertain Domains. *ACM Transactions on Evolutionary Learning* (2022).

[16] Manon Flageat, Felix Chalumeau, and Antoine Cully. 2023. Empirical analysis of PGA-MAP-Elites for Neuroevolution in Uncertain Domains. *ACM Transactions on Evolutionary Learning* 3, 1 (2023), 1–32.

[17] Manon Flageat and Antoine Cully. 2020. Fast and stable MAP-Elites in noisy domains using deep grids. In *Artificial Life Conference Proceedings 32*. MIT Press One Rogers Street, Cambridge, MA 02142-1209, USA journals-info …, 273–282.

[18] Manon Flageat and Antoine Cully. 2023. Uncertain Quality-Diversity: Evaluation methodology and new methods for Quality-Diversity in Uncertain Domains. *IEEE Transactions on Evolutionary Computation* (2023).

[19] Manon Flageat, Bryan Lim, Luca Grillotti, Maxime Allard, Simón C. Smith, and Antoine Cully. 2022. Benchmarking Quality-Diversity Algorithms on Neuroevolution for Reinforcement Learning. https://doi.org/10.48550/ARXIV.2211.02193

[20] Matthew Fontaine and Stefanos Nikolaidis. 2021. Differentiable Quality Diversity. *Advances in Neural Information Processing Systems* 34 (2021).

[21] Matthew C Fontaine, Julian Togelius, Stefanos Nikolaidis, and Amy K Hoover. 2020. Covariance matrix adaptation for the rapid illumination of behavior space. In *Proceedings of the 2020 genetic and evolutionary computation conference*. 94–102.

[22] C. Daniel Freeman, Erik Frey, Anton Raichuk, Sertan Girgin, Igor Mordatch, and Olivier Bachem. 2021. *Brax - A Differentiable Physics Engine for Large Scale Rigid Body Simulation*. http://github.com/google/brax

[23] Adam Gaier, Alexander Asteroth, and Jean-Baptiste Mouret. 2018. Data-efficient design exploration through surrogate-assisted illumination. *Evolutionary computation* 26, 3 (2018), 381–410.

[24] Adam Gaier, Alexander Asteroth, and Jean-Baptiste Mouret. 2019. Are quality diversity algorithms better at generating stepping stones than objective-based search?. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*. 115–116.

[25] Adam Gaier, Alexander Asteroth, and Jean-Baptiste Mouret. 2020. Discovering representations for black-box optimization. In *Proceedings of the 2020 Genetic and Evolutionary Computation Conference*. 103–111.

[26] Daniele Gravina, Ahmed Khalifa, Antonios Liapis, Julian Togelius, and Georgios N Yannakakis. 2019. Procedural content generation through quality diversity. In *2019 IEEE Conference on Games (CoG)*. IEEE, 1–8.

[27] Nikolaus Hansen. 2006. The CMA evolution strategy: a comparing review. *Towards a new evolutionary computation* (2006), 75–102.

[28] Minqi Jiang, Michael Dennis, Edward Grefenstette, and Tim Rocktäschel. 2023. minimax: Efficient Baselines for Autocurricula in JAX. *arXiv preprint arXiv:2311.12716* (2023).

[29] Niels Justesen, Sebastian Risi, and Jean-Baptiste Mouret. 2019. Map-elites for noisy domains by adaptive sampling. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*. 121–122.

[30] Robert Tjarko Lange. 2022. evosax: JAX-based Evolution Strategies. *arXiv preprint arXiv:2212.04180* (2022).

[31] Joel Lehman and Kenneth O Stanley. 2011. Evolving a diversity of virtual creatures through novelty search and local competition. In *Proceedings of the 13th annual conference on Genetic and evolutionary computation*. 211–218.

[32] Joel Lehman and Kenneth O Stanley. 2011. Novelty search and the problem with objectives. In *Genetic programming theory and practice IX*. Springer, 37–56.

[33] Bryan Lim, Maxime Allard, Luca Grillotti, and Antoine Cully. 2022. Accelerated Quality-Diversity for Robotics through Massive Parallelism. *arXiv preprint arXiv:2202.01258* (2022).

[34] Bryan Lim, Alexander Reichenbach, and Antoine Cully. 2022. Learning to walk autonomously via reset-free quality-diversity. In *Proceedings of the Genetic and Evolutionary Computation Conference*. 86–94.

[35] Viktor Makoviychuk, Lukasz Wawrzyniak, Yunrong Guo, Michelle Lu, Kier Storey, Miles Macklin, David Hoeller, Nikita Rudin, Arthur Allshire, Ankur Handa, et al. 2021. Isaac gym: High performance gpu-based physics simulation for robot learning. *arXiv preprint arXiv:2108.10470* (2021).

[36] Jean-Baptiste Mouret and Jeff Clune. 2015. Illuminating search spaces by mapping elites. *arXiv preprint arXiv:1504.04909* (2015).

[37] Olle Nilsson and Antoine Cully. 2021. Policy gradient assisted MAP-Elites. In *Proceedings of the Genetic and Evolutionary Computation Conference*. 866–875.

[38] Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. 2017. Curiosity-driven exploration by self-supervised prediction. In *International conference on machine learning*. PMLR, 2778–2787.

[39] Thomas Pierrot, Valentin Macé, Felix Chalumeau, Arthur Flajolet, Geoffrey Cideron, Karim Beguir, Antoine Cully, Olivier Sigaud, and Nicolas Perrin-Gilbert. 2022. Diversity policy gradient for sample efficient quality-diversity optimization. In *Proceedings of the Genetic and Evolutionary Computation Conference*. 1075–1083.

[40] Justin K Pugh, Lisa B Soros, and Kenneth O Stanley. 2016. Quality diversity: A new frontier for evolutionary computation. *Frontiers in Robotics and AI* 3 (2016), 40.

[41] Alexander Rutherford, Benjamin Ellis, Matteo Gallici, Jonathan Cook, Andrei Lupu, Gardar Ingvarsson, Timon Willi, Akbir Khan, Christian Schroeder de Witt, Alexandra Souly, Saptarashmi Bandyopadhyay, Mikayel Samvelyan, Minqi Jiang, Robert Tjarko Lange, Shimon Whiteson, Bruno Lacerda, Nick Hawes, Tim Rocktaschel, Chris Lu, and Jakob Nicolaus Foerster. 2023. JaxMARL: Multi-Agent RL Environments in JAX. *arXiv preprint arXiv:2311.10090* (2023).

[42] Tim Salimans, Jonathan Ho, Xi Chen, Szymon Sidor, and Ilya Sutskever. 2017. Evolution strategies as a scalable alternative to reinforcement learning. *arXiv preprint arXiv:1703.03864* (2017).

[43] Yujin Tang, Yingtao Tian, and David Ha. 2022. EvoJAX: Hardware-Accelerated Neuroevolution. *arXiv preprint arXiv:2202.05008* (2022).

[44] Bryon Tjanaka, Matthew C Fontaine, Aniruddha Kalkar, and Stefanos Nikolaidis. 2022. Training Diverse High-Dimensional Controllers by Scaling Covariance Matrix Adaptation MAP-Annealing. *arXiv preprint arXiv:2210.02622* (2022).

[45] Bryon Tjanaka, Matthew C Fontaine, Julian Togelius, and Stefanos Nikolaidis. 2022. Approximating Gradients for Differentiable Quality Diversity in Reinforcement Learning. *arXiv preprint arXiv:2202.03666* (2022).

[46] Vassilis Vassiliades and Jean-Baptiste Mouret. 2018. Discovering the Elite Hypervolume by Leveraging Interspecies Correlation. In *GECCO 2018 - Genetic and Evolutionary Computation Conference*. Kyoto, Japan. https://doi.org/10.1145/3205455.3205602

[47] Daan Wierstra, Tom Schaul, Tobias Glasmachers, Yi Sun, Jan Peters, and Jürgen Schmidhuber. 2014. Natural evolution strategies. *The Journal of Machine Learning Research* 15, 1 (2014), 949–980.