Evolution Strategies' Genetic Operators Revisited Over Integer Search-Spaces

Jacob DE NOBEL*, Diederick VERMETTEN†, Ofer M. SHIR‡, Michael EMMERICH§, Thomas BÄCK*
*Leiden Institute of Advanced Computer Science, Leiden, The Netherlands
Emails: j.p.de.nobel@liacs.leidenuniv.nl, t.h.w.baeck@liacs.leidenuniv.nl
†CNRS, Sorbonne University, Paris, France
Email: diederick.vermetten@lip6.fr
‡Tel-Hai - University of Kiryat Shmona in the Galilee, Israel
Email: ofersh@telhai.ac.il
§Faculty of Information Technology, University of Jyväskylä, Finland
Email: michael.t.m.emmerich@jyu.fi

***Abstract*—Evolution Strategies (ESs) are highly effective for continuous black-box optimization, yet their standard operators do not transfer cleanly to integer lattices: rounding introduces artefacts, step sizes have a natural lower bound, and recombination can break feasibility and distort selection. We empirically isolate how *mutation distributions* and *recombination operators* jointly shape the behavior of classical self-adaptive (μ, λ)-ES variants on convex quadratic benchmarks over $\mathbb{Z}^n$, including ℓ_2- and ℓ_1-based landscapes up to $n = 100$, and we compare against an integer-handling CMA-ES baseline. Across all integer settings, Double Geometric (DG) mutation consistently dominates truncated-normal (rounded Gaussian) mutation and, when paired with uniform discrete recombination, often matches and sometimes surpasses the covariance-based baseline despite using only a single global step size. In contrast, intermediate recombination systematically degrades DG-based search, revealing a strong operator–domain mismatch. Finally, an ablation on *zero mutations* shows that permitting null steps can improve DG-based ES performance, suggesting a step-size adaptation lag specific to lattice-native mutations. Importantly, several methods exhibit a pronounced "last-mile" slowdown on $\mathbb{Z}^n$, where late-stage progress reduces to rare single-coordinate corrections; We hypothetise that DG mutation with discrete recombination mitigates this effect and improves final-phase reliability.**

I. Introduction

Evolution Strategies (ESs) constitute well-established randomized search heuristics for black-box optimization, as supported by several decades of theoretical analyses and empirical evidence [1], [2], [3], [4]. ESs were established as leading solvers in the continuous domain[1], on which they remarkably excel, but have also been introduced to integer and mixed-integer domains throughout the years [6]. Their behavior on unbounded integer search spaces, i.e., $f : \mathbf{x} \in \mathbb{Z}^n \to \mathbb{R}$, has been observed to significantly change. Mutation must respect the discrete structure of the search space, step sizes cannot shrink below one, and promising descent directions in the continuous landscape may not correspond to any feasible move on the integer lattice. As a consequence, the effectiveness of standard ESs design choices on integer domains is far less understood than in the continuous case [7].

These difficulties arise even for the simplest classes of convex functions. In continuous space, smooth positive-definite quadratic functions admit a unique global minimizer due to the convexity property. On the integer lattice, the situation is fundamentally different: If search moves are restricted to coordinate-wise flips, the global structure of a strictly convex quadratic model can be introduced with local minima [7], [8]. This effect induces a two-phase convergence behavior even on simple integer-valued sphere functions: when most coordinates are still far off the optimizer's values, the dynamics resemble those of a continuous ES, whereas once many coordinates are set correctly, progress is dominated by "fixing" the remaining few. The latter phase or *Last-Mile* is well captured by the so-called *Coupon Collector* perspective, in which the runtime is governed by the waiting time until the last incorrect coordinates are corrected.

In Integer ESs (IESs), Rudolph [9] introduced the Double Geometric (DG) distribution for mutations on unbounded integer search and, using its favorable theoretical properties, designed a DG-based mutation operator. This operator later became a standard component of mixed-integer parameter optimization with evolution strategies (e.g., Mixed-Integer ES variants [10]). The DG induces a heavy-tailed mutation law with a single scale parameter and avoids rounding artefacts; in this sense, it can be viewed as a discrete analogue of the continuous Laplace distribution (or, equivalently, a two-sided geometric law on $\mathbb{Z}$). More recently, [7] extended this line of work by constructing a correlated multivariate DG-based mutation operator that enables controlled covariance structures on the integer lattice and offers a possible analogue of rotational mutation mechanisms in classical ESs [11].

Recombination is another genetic operator whose behavior differs in the discrete setting. In modern continuous ES, (weighted) intermediate recombination is the standard method for combining parental information [4]. On the integer lattice, uniform discrete recombination offers a structurally compatible alternative. In discrete evolutionary algorithms, uniform

[1]Historically, Rechenberg's first experimental campaigns comprised integer problems, such as the drag minimization using a kinked plate setup, and the two-phase nozzle design [5].

crossover has been analysed rigorously and shown to accelerate optimisation by combining complementary information from multiple parents [12]. Despite this, recombination operators in IESs are rarely studied in isolation, and it remains unclear how different recombination schemes interact with discrete mutation distributions.

In this work, we investigate how both the mutation distribution and recombination operator jointly influence the behaviour of IESs on discrete optimisation problems with different geometric structures. We consider both ℓ_2-type and ℓ_1-type objective landscapes. Functions with an ℓ_2 geometry, such as sphere-, ellipse-, and discus-type quadratics, have smooth level sets and isotropic curvature patterns. In contrast, functions based on the ℓ_1 norm exhibit level sets with sharp corners and a coordinate-aligned structure. Comparing these geometries allows us to assess how norm-induced landscape structure influences ESs' behaviour in both continuous and integer domains. Explicitly, we pose the following **research question**:

> How do specific operator design choices—including mutation distributions, crossover schemes, and integer-repair mechanisms—modulate the optimization performance of IESs across quadratic landscapes with varying structural properties?

To address this and isolate these mechanisms, we evaluate several variants of the classical self-adaptive ES using truncated-normal and double-geometric mutations, as well as different recombination operators. We compare their behaviour on integer versions of the functions described above, and also analyse the corresponding continuous problems using normal-mutation ES and CMA-ES [4]. Specifically, we provide an empirical comparison of mutation distributions and recombination operators for ES on integer quadratic problems, and identify consistent performance patterns across problem geometry, dimensionality, and domain type. Our results show that simple classical ES variants, when equipped with appropriate operators, can match or outperform covariance-based methods on the studied problems. In particular, the contributions of this paper are:

1) **Performance Superiority:** Empirical evidence that IESs perform best on quadratic programs when utilizing DG-based mutations paired with discrete recombination.
2) **Adaptive Decoupling:** A fundamental limitation in current self-adaptation mechanisms where the strategy parameter becomes "out-of-sync" with the discrete search space. This lag necessitates using zero mutations to allow the step size to adapt, thereby acting as a temporal buffer for the adaptive process.
3) **Functional Dissociation:** Whereas normal mutations (TN) benefit from the averaging repair of intermediate recombination, non-normal mutations (DG) strictly require the structural preservation of discrete recombination.

II. Evolution Strategies for Integer Spaces

ESs emerged from engineering practice rather than biological theory – they were originally devised as pragmatic randomized search heuristics for engineering optimization, with evolution serving more as an inspirational metaphor than a governing principle. In contrast, Genetic Algorithms (GAs) originated from the explicit computational modeling of genetic processes, drawing their foundational metaphor from the mapping between bitstring representations and nucleotide sequences [13]. In this section, we formulate the classical ESs' mechanism to form Integer ESs (IESs) while referring to GAs' operation as a reference.

Algorithm 1 Generic (μ, λ)-ES

Require: objective $f\colon \mathcal{X} \to \mathbb{R}$, population sizes μ, λ, mutation dist. family $Q_{\mathbf{s}}$, recomb. operator $\mathcal{R}$, projection $\Pi_{\mathcal{X}}$, learning rate $\tau \in \mathbb{R}_+$, step-size's lower bound lb_s

Initialise parent population $\mathcal{P}_0 = \{(\mathbf{x}_0^{(i)}, \mathbf{s}_0^{(i)})\}_{i=1}^{\mu}$
for $t = 0, 1, 2, \ldots$ **do**
  $\mathcal{O}_t \leftarrow \emptyset$
  for $j = 1$ to λ **do**
    $(\mathbf{x}_{\mathrm{rec}}, \mathbf{s}_{\mathrm{rec}}) \leftarrow \mathcal{R}(\mathcal{P}_t)$
    $\tilde{\mathbf{s}} \leftarrow \max\{lb_s,\ \mathbf{s}_{\mathrm{rec}} \cdot \exp(r_{\mathbf{s}})\}, \qquad r_{\mathbf{s}} \sim \mathcal{N}(0, \tau)$
    $\tilde{\mathbf{x}} \leftarrow \Pi_{\mathcal{X}}(\mathbf{x}_{\mathrm{rec}} + \mathbf{z}), \qquad \mathbf{z} \sim Q_{\tilde{\mathbf{s}}}$
    $\mathcal{O}_t \leftarrow \mathcal{O}_t \cup \{(\tilde{\mathbf{x}}, \tilde{\mathbf{s}})\}$
  $\mathcal{P}_{t+1} \leftarrow \mu$ best offspring according to $f(\tilde{\mathbf{x}})$

A. Framework and Notation

We consider a self-adaptive (μ, λ)-ES operating on a search space $\mathcal{X}$, which is either the continuous domain $\mathbb{R}^n$ or the integer lattice $\mathbb{Z}^n$ (see Algorithm 1). Following the classical ES literature [11] and the integer variant of [9], individuals are represented by candidate solutions $\mathbf{x} \in \mathcal{X}$ with associated step-size parameters $\mathbf{s} \in \mathbb{R}_+^n$. At each generation t, the algorithm maintains a parental population $\mathcal{P}_t = \{(\mathbf{x}_t^{(i)}, \mathbf{s}_t^{(i)})\}_{i=1}^{\mu}$, from which λ offspring $\mathcal{O}_t = \{(\tilde{\mathbf{x}}_t^{(j)}, \tilde{\mathbf{s}}_t^{(j)})\}_{j=1}^{\lambda}$ are generated via recombination and mutation. Offspring are evaluated on $f : \mathcal{X} \to \mathbb{R}$, and the μ best individuals form the next parental population. The algorithm is fully specified by the population sizes μ, λ, a recombination operator $\mathcal{R}\colon \mathcal{P}_t \mapsto (\mathbf{x}_{\mathrm{rec}}, \mathbf{s}_{\mathrm{rec}})$, a mutation distribution from a family $Q_{\mathbf{s}}$, and a projection operator $\Pi_{\mathcal{X}}$ enforcing feasibility (identity in the continuous case and rounding or clipping in the integer case). When operating on the integer lattice, it is critical to specify the underlying metric [7]. Let $\mathbf{z}$ denote the sampled mutation vector; we characterize its magnitude through the expected ℓ_1-norm, $S := \mathbb{E}[\|\mathbf{z}\|_1] = \sum_{i=1}^{n} \mathbb{E}[|z_i|]$, which constitutes the mean step-size on $\mathbb{Z}^n$. Differences between ES variants arise solely through the choice of $Q_{\mathbf{s}}$, $\mathcal{R}$, and $\Pi_{\mathcal{X}}$.

B. The Recombination Operator

Inspired by the organic mechanism of a *meiotic cell division*, where the genetic material is reordered by means of *crossover* between the chromosomes, the classical ESs' recombination operator considers sharing the information from up to ν parent individuals [14]; Whenever $\nu > 2$, it is usually referred to as *multi-recombination*. Unlike other Evolutionary Algorithms, and particularly GAs [15], the ESs' recombination operator

obtains only a single offspring. There are two fundamental ways to recombine parents:

- *Discrete recombination*: one of the alleles is randomly chosen among ν parents. Given a parental matrix of the old generation, $\mathbf{A}^{\mathcal{P}} = \left(\mathbf{a}_1^{\mathcal{P}}, \mathbf{a}_2^{\mathcal{P}}, ..., \mathbf{a}_\nu^{\mathcal{P}}\right)$, the new *recombinant* $\mathbf{a}^{\mathcal{O}}$ is constructed by:

$$\left(\mathbf{a}^{\mathcal{O}}\right)_i := \left(\mathbf{a}_{m_i}^{\mathcal{P}}\right)_i, \quad m_i := rand\{1, .., \nu\}$$

- *Intermediate recombination*: the values of ν parents are averaged, typically with uniform weights. Essentially, this is equivalent to calculating the *centroid* of the ν parents:

$$\left(\mathbf{a}^{\mathcal{O}}\right)_i := \frac{1}{\nu} \sum_{j=1}^{\nu} \left(\mathbf{a}_j^{\mathcal{P}}\right)_i \ . \tag{1}$$

In practice, the recombination operator in the standard ES typically chooses ν parents and applies *discrete recombination* to the decision variables ($\mathbf{x}$), and at the same time it applies *intermediate recombination* to the strategy parameters ($\mathbf{s}$). The suitability of these operators for IESs depends on how they handle the underlying domain. Intermediate recombination, though effective in continuous spaces, requires a rounding mechanism to map averages back to the integer lattice, which can introduce artifacts into the selection process. Discrete recombination avoids this necessity; by sampling coordinates directly from parents, it naturally maintains the discrete nature of the lattice and avoids the biases associated with rounding. Within traditional GA research, the *building block hypothesis* (BBH) (see, e.g., [15]) explains crossover's effectiveness through the combination of high-quality yet distinct building blocks – specific schemata or gene segments from different parents – as the primary mechanism for propagating fitness. This hypothesis, however, remains a subject of ongoing debate within the GA community. In ES populations, diversity decreases rapidly, rendering BBH largely inapplicable. Instead, Beyer's analysis [16] gave rise to the *genetic repair hypothesis*, which asserts that ES recombination over continuous domains operates through a distinct mechanism—exploiting *shared* parental traits to counteract stochastic perturbations. Specifically, recombination primarily mitigates deleterious mutation and noise effects on decision variables, which permits the use of larger step sizes without compromising convergence performance. Notably, this repair mechanism is amplified when intermediate recombination is employed rather than discrete recombination, but its applicability to integer search has never been questioned.

In this study we consider three recombination variants and denote them as follows:

($\mathcal{R}0$) *No recombination*: each offspring selects one parent uniformly at random from the μ best individuals.
($\mathcal{R}1$) *Intermediate recombination* with $\nu = \mu$.
($\mathcal{R}2$) *Uniform discrete recombination*.

C. The Mutation Operator

The ESs' mutation baseline is the normal distribution, which fits naturally for $\mathcal{X} = \mathbb{R}^n$. When turning to $\mathbb{Z}^n$, the 'reflex' action is to *round* the values, yielding the so-called Truncated Normal (TN) distribution. This distribution induces the probability for a random variable z to take unbounded integer values approximately as $\Pr\{\texttt{round}(z) = k\} \approx \frac{1}{\sqrt{\pi}} \exp\left(-\frac{k^2}{2\sigma^2}\right)$ [7]. Moreover, its expected ℓ_1-norm is approximated as a multiplicative function of its defining step-size σ_{TN} [7]:

$$S_{TN} \approx n\sqrt{\frac{2}{\pi}} \cdot \sigma_{TN} \quad \Longleftrightarrow \quad \sigma_{TN} \approx \sqrt{\frac{\pi}{2}} \cdot S_{TN}/n.$$

Next, when adopting the proposed operator by [9], the DG distribution induces the following probability for a random variable z to take unbounded integer values:

$$\Pr\{z = k\} = \frac{p}{2-p} \cdot (1-p)^{|k|}, \quad k \in \mathbb{Z}.$$

It is symmetric around zero, heavy-tailed, and controlled by a single parameter $p \in (0, 1)$. It enjoys the following relation with respect to its defining parameter p:

$$S_{DG}(p) = n \cdot \frac{2(1-p)}{p(2-p)} \Longleftrightarrow p = 1 - \frac{S_{DG}/n}{\sqrt{\left(1 + (S_{DG}/n)^2\right)} + 1}.$$

We do not account for correlations, and we additionally focus on scenarios where the n random variables are identically drawn under the same settings. We therefore reduce the strategy parameters' structure $\mathbf{s}$ to a scalar, and accordingly, we consider the mutation operator as a sequential independent sampling of the single-variable distributions above. Due to stochastic independence, the mean step-size behaves respectively with $S = n \cdot \mathbb{E}[|z|]$. Altogether, we consider in this work three mutation variants:

($\mathcal{M}0$) *Gaussian mutation* (continuous ES): $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \tilde{\sigma}^2\mathbf{I})$,
($\mathcal{M}1$) *TN* mutation (integer ES): a Gaussian step is drawn in $\mathbb{R}^n$ and mapped to $\mathcal{X}$ via $\Pi_{\mathcal{X}}$,
($\mathcal{M}2$) *DG* mutation (integer ES): $\mathbf{z}$ is sampled directly from a DG distribution on $\mathbb{Z}^n$.

Self-adaptation on the integer lattice requires a fundamental shift in perspective: the expected ℓ_1-norm S constitutes the proper mean step-size. Unlike the traditionally-used standard deviation, which relies on Gaussian assumptions, S provides a robust measure of scale that generalizes across non-normal discrete distributions [7]. Simultaneously, we leverage the known relation $p(S_{DG})$ as well as the approximate multiplicative relationship $\sigma_{TN}(S)$, allowing us to maintain standard self-adaptive update rules across our various mutation distributions (see lines 6-7, Alg. 1). Note that the value of $\tilde{s}$ is constrained to a lower bound of lb_s, which is 1 on the integer lattice and 0 otherwise [9].

III. Problem Description

We consider standard convex quadratic benchmark functions on both continuous and integer domains. Experiments are conducted in dimensions $n \in \{2, 3, 5, 10, 20, 40, 100\}$. For each benchmark, we evaluate four variants: continuous ℓ_2, continuous ℓ_1, integer ℓ_2, and integer ℓ_1. All objective functions are unbounded. For evaluation, search points are

initialized within the hypercube $[-50, 50]^n$, and the global optimum is placed uniformly at random within this region. Specifically, we consider weighted, axis-aligned quadratics of the form

$$f(\mathbf{x}) = \|\mathbf{W}(\mathbf{x} - \mathbf{x}^\star)\|_p, \tag{2}$$

where $\mathbf{x}^\star \in \mathcal{X}$ is the global optimum, $p \in \{1, 2\}$ denotes the norm, and $\mathbf{W} = \mathrm{diag}(w_1, \ldots, w_n)$ defines the conditioning. For $p = 2$, this corresponds to a standard quadratic objective up to a monotone transformation. We consider four standard benchmark classes defined by the choice of weights w_i:

- **Sphere:** $w_i = 1$ for all i.
- **Ellipse:** $w_i = c^{\frac{i-1}{n-1}}$, $i = 1, \ldots, n$.
- **Discus:** $w_1 = c$, $w_i = 1$ for $i > 1$.
- **Cigar:** $w_1 = 1$, $w_i = c$ for $i > 1$.

For the ill-conditioned benchmarks (Ellipse, Discus, and Cigar), the condition number is fixed to $c = 10^4$. All functions are axis-aligned; rotated variants are deliberately excluded to isolate the effects of mutation distributions and recombination operators.

IV. Experimental Setup

For each benchmark function, dimension, and problem variant (continuous/integer, ℓ_1/ℓ_2), we perform 25 independent runs using independent random seeds. Each run is allocated a fixed evaluation budget of $10^4 \cdot n$ function evaluations. The code and data used in this study, together with additional figures, are available in our Zenodo repository [17].

a) Algorithms: We compare several (μ, λ)-ES, differing only in their mutation distributions and recombination operators; all other algorithmic parameters are kept fixed across variants: $\lambda = \lfloor 7 \log n \rfloor$, $\mu = \lambda/2$, and $\tau = \sqrt{1/n}$. For continuous problems, Gaussian mutation ('Norm') is used exclusively. On the integer lattice, both the TN and DG mutation distributions (see Section II-C) have been used, denoted by 'Norm' and 'dGeom' respectively. For each (μ, λ)-ES variant, we simultaneously evaluate the three recombination strategies (see Section II-B), denoted by: (μ, λ), $(\mu{:}\mu, \lambda)$ and $(\mu{:}2, \lambda)$ for $\mathcal{R}0$, $\mathcal{R}1$ and $\mathcal{R}2$, respectively. This yields six ES variants for integer optimization, three in the continuous domain. As a *baseline*, we include the integer-handling variant of CMA-ES (CMA-IH) [6], which applies a rounding-based projection to integer domains. The default population size recommended in the CMA-ES literature is used for each dimension, and no additional parameter tuning is performed. CMA-IH serves as a reference for state-of-the-art covariance-based search on integer domains.

b) Zero mutations: For IESs, both TN and DG mutation operators can generate offspring identical to the recombined parent when $\mathbf{z} = \mathbf{0}$, which can occur when $\mathbf{s} \downarrow lb_s$. To assess the impact of such zero mutations, we include an ablation in which zero steps are explicitly disallowed. In this no-zero variant, mutation steps are resampled until at least one coordinate differs from zero, ensuring that every offspring induces a change on the integer lattice. All other algorithmic parameters, including step-size adaptation and recombination, are kept identical.

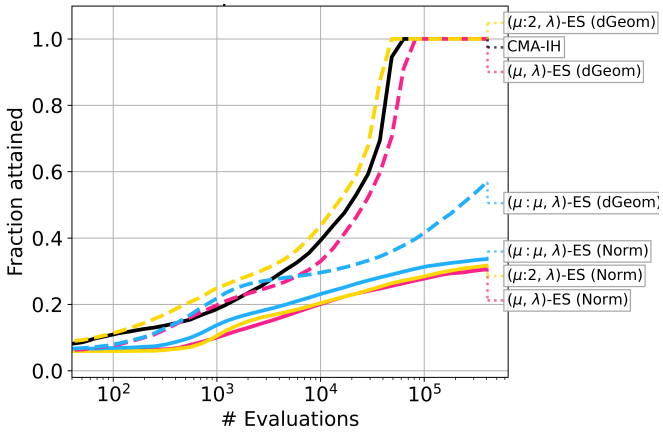


Fig. 1: ECDF for the ℓ_2-ellipse on $\mathbb{Z}^{40}$. Dashed lines correspond to algorithms using the DG distribution, while solid lines indicate algorithms based on the TN distribution. Colors denote the recombination operator; **CMA-IH** is shown in black.

c) Performance Measure: To quantify the performance of the different algorithms, we take an anytime performance perspective [18]. Specifically, we normalize the convergence curves by adding 1 to all function values before aggregating over runs to compute a logarithmically spaced Empirical Cumulative Distribution Function (ECDF) ranging from 10^8 to 10^0. We aggregate the ECDF for each function, dimension, and algorithm into a single scalar by computing *the area under the ECDF*, which is equivalent to the Area Over the Convergence Curve (AOCC) [18]. For the AOCC metric, larger values indicate better performance (i.e., reaching lower function values earlier in the optimization process). All performance calculations are done using the IOHinspector package [19].

V. Results

We begin our analysis by examining a single experiment. Figure 1 shows the ECDF of the ℓ_2-ellipse on $\mathbb{Z}^{40}$. A clear distinction by $Q_\mathbf{s}$ is evident in this plot, with variants based on the Normal distribution performing considerably worse than their DG counterparts. In fact, the DG-based IES with discrete recombination even slightly outperforms the state-of-the-art CMA-IH baseline. This is interesting, since this problem is highly conditioned ($c = 10^4$), and the classic IES uses only a single step-size strategy parameter. Moreover, we observe that although intermediate recombination $(\mu : \mu)$ is the top-performing operator for the TN-based IES, it clearly degrades performance for the DG-based IES. To assess how these observations generalize across dimensions, we plot the AOCC extracted from the ECDF (see Section IV-0c) as a function of dimensionality in Figure 2. The results show a clear dimensionality effect for TN-based algorithms: while their performance is comparable to DG-based variants for $n \leq 3$, it rapidly deteriorates, and they are unable to achieve meaningful optimization for $n \geq 10$. For DG-based IES variants, performance differences are small in low dimensions; however, the intermediate recombination operator similarly breaks down for $n \geq 10$. In contrast, the no-recombination and discrete-recombination DG variants remain largely stable

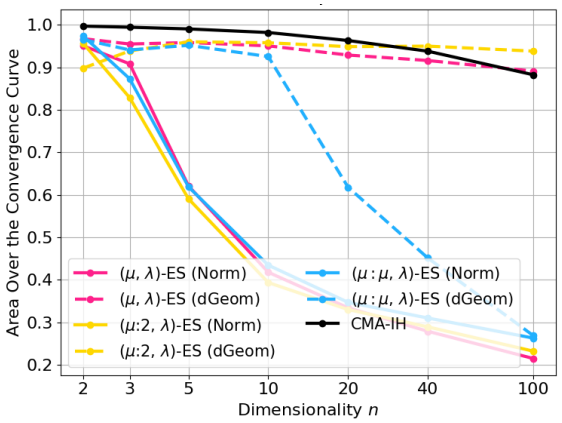


Fig. 2: Area over the convergence curve for the ℓ_2-ellipse integer problem. The x-axis shows the dimensionality, and values are computed from the ECDF.

with respect to dimensionality. The CMA-IH baseline shows a gradual degradation as n increases, allowing the DG-based IES with discrete recombination to outperform CMA-IH for $n \geq 40$.

a) Comparing across benchmarks: We now turn to comparing the effects of domain (continuous vs. integer), metric (ℓ_2- vs. ℓ_1-norm), and conditioning across benchmark classes. Figure 3 summarizes these effects for the Sphere and Ellipse functions, which represent the two extremes in conditioning. The Discus and Cigar benchmarks exhibit intermediate behaviour and follow the same overall ranking patterns. Results for these problems are provided on Zenodo [17]. For the Sphere function, performance differences are primarily driven by the choice of metric (norm) and search domain. In the continuous case, all algorithms perform similarly on the ℓ_2 sphere, whereas the ℓ_1 norm substantially increases problem difficulty, particularly in higher dimensions. On the integer sphere, DG-based mutation consistently outperforms TN-based mutation and achieves the strongest performance when combined with discrete recombination. Notably, DG-based IES variants with discrete or no recombination match or outperform CMA-IH in higher dimensions ($n \geq 40$). The Ellipse benchmark highlights the impact of conditioning. In the continuous domain, CMA-ES clearly dominates classical ES variants for both norms, reflecting the benefits of learning a coordinate-wise scaling. On the integer ellipse, however, DG-based ES variants perform on par with or better than CMA-IH despite using only a single global step size. As in the Sphere case, discrete recombination is most effective for DG mutation, while intermediate recombination consistently degrades performance. Finally, across all integer benchmarks, DG-based IES variants favor discrete over no recombination, with intermediate recombination considerably degrading performance, whereas TN-based variants show the reverse ordering, albeit with only minor performance differences.

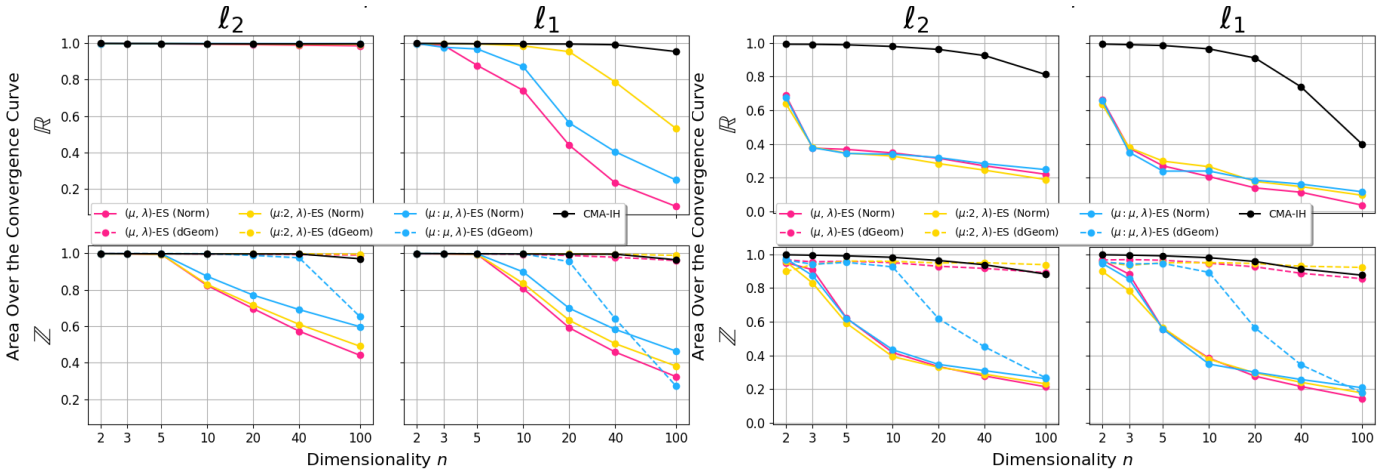
b) Zero Mutations: We next investigate the effect of allowing zero mutations on the performance of integer ES. Figure 4 shows the paired AOCC differences between the default ES (zero steps allowed) and the no-zero variant across all integer benchmarks; positive values indicate a benefit from allowing zero mutations. For the TN-based mutation, the effect of zero mutations is negligible across all recombination operators and benchmarks. For DG-based mutation, however, a clearer pattern emerges: allowing zero mutations is consistently beneficial when no recombination is used. This observation is counterintuitive, as zero mutations correspond to evaluating offspring identical to their parent and thus consume function evaluations without inducing a change. Since these evaluations are still counted toward the evaluation budget, the effective performance gain is, in fact, underestimated by the reported differences. In addition, on the Ellipse benchmark, allowing zero mutations also appears to yield a positive effect for DG-based mutation when combined with discrete recombination.

c) Last-Mile: To illustrate late-stage convergence effects, Figure 5 shows convergence curves on the 100-dimensional Sphere benchmark for both ℓ_2 and ℓ_1 norms in continuous ($\mathbb{R}^{100}$) and integer ($\mathbb{Z}^{100}$) domains. We compare CMA-ES with integer handling (CMA-IH) to the best-performing integer ES variant, namely the DG-based ES with discrete recombination. On the continuous ℓ_2 Sphere, CMA-ES converges smoothly to the optimum, whereas the ℓ_1 norm introduces stagnation near the optimum. In the integer domain, CMA-IH exhibits similar stagnation under both norms, with nearly overlapping convergence curves: after an initial phase of approximately log-linear progress, improvements slow markedly near the optimum. This behaviour illustrates the *last-mile problem*, which is particularly pronounced on the integer lattice, where progress in the final phase requires identifying rare single-coordinate improvements. In contrast, the DG-based ES with discrete recombination does not exhibit premature stagnation. Although it is slower in the initial search phase, it continues to make progress in the final phase and converges more reliably to the optimum for both norms, with the ℓ_1 case remaining more challenging.

VI. Discussion

Our results reveal consistent interaction patterns between mutation and recombination on integer domains. DG-based mutation consistently outperforms TN mutation on integer domains as expected from previous results [9], [7]. Surprisingly, DG-based IES variants also match or outperform CMA-IH on several benchmarks, including ill-conditioned problems such as the ellipse. Notably, this is achieved using only a single global step size, indicating that lattice-native mutation distributions can partially compensate for the absence of coordinate-wise adaptation when optimizing over $\mathbb{Z}^n$. Alternatively, this could also indicate that the mechanism in CMA-IH is suboptimal, as observed in Figure 5, which identified the *last-mile* problem. Developing a derandomized ES that uses the DG distribution is, therefore, an interesting avenue.

ℓ_1 versus ℓ_2 Geometry: On the integer lattice, the most elementary notion of "move size" is coordinate-wise: changing one coordinate by ± 1 is the smallest non-trivial mutation, and progress in the late phase typically consists of fixing a



(a) Sphere **(b)** Ellipse

Fig. 3: Area over Convergence Curve (AOCC) for the Sphere (a) and Ellipse (b) benchmarks across dimensions $n \in \{2, 3, 5, 10, 20, 40, 100\}$. Results are shown for continuous ($\mathbb{R}^n$, top row) and integer ($\mathbb{Z}^n$, bottom row) domains, each evaluated with the ℓ_2 norm (left column) and the ℓ_1 norm (right column). Higher values indicate better anytime performance.

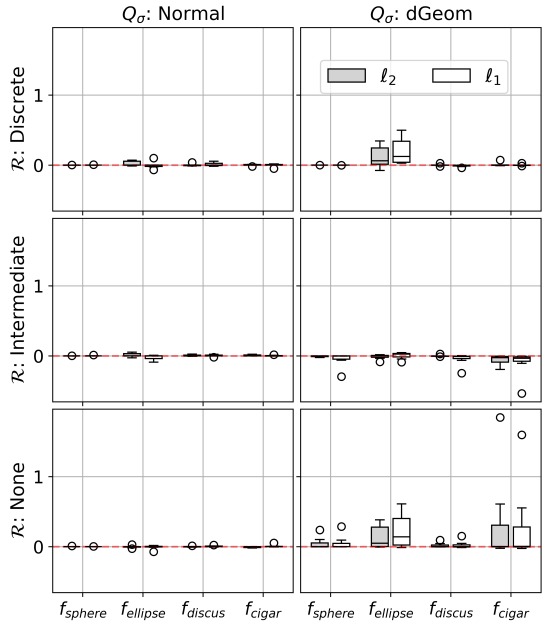


Fig. 4: Paired AOCC differences between the default ES (zero steps allowed) and the no-zero variant on integer benchmarks. Differences are computed per instance as $(\text{default} - \text{no-zero})/\text{no-zero}$; positive values favour allowing zero mutations.

small set of remaining incorrect coordinates. This motivates the usage of S as a robust measure of scale on the integer lattice, as previously argued in II-A. In contrast, ℓ_2-symmetric (spherical) parameterizations are geometrically natural in $\mathbb{R}^n$, but become less informative after projection to $\mathbb{Z}^n$: much of the ℓ_2-mass concentrates in directions that do not correspond to "cheap" lattice moves, and after rounding/clipping it can translate into many ineffective steps (e.g., zero-mutations or overly diagonal moves) whose discrete effect is poorly predicted by $\|\mathbf{z}\|_2$. By controlling mutation through an ℓ_1-symmetric notion of scale, we directly control the expected

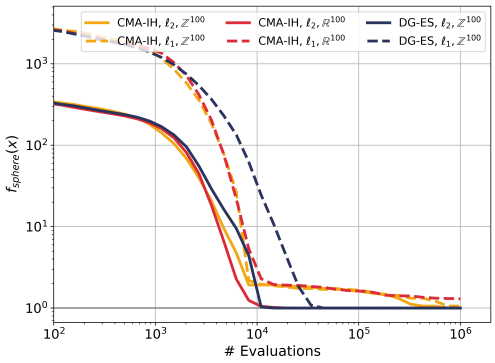


Fig. 5: Geometric mean of 25 independent runs on the 100-dimensional sphere benchmark, for the ℓ_1 and ℓ_2 on both domains, i.e. $\mathbb{Z}^{100}$ and $\mathbb{R}^{100}$. Only the CMA-IH and the $(\mu : 2, \lambda)$-ES with the DG distribution (DG-ES) are shown.

number and magnitude of coordinate changes, which aligns with coupon-collector-like last-mile dynamics and with ℓ_1-type landscapes whose level sets have axis-aligned corners. This also yields an additive, dimension-transparent step-size interpretation (via $\mathbb{E}\|\mathbf{z}\|_1$) that transfers naturally across n and avoids artifacts introduced by projecting an ℓ_2-calibrated continuous distribution onto the lattice. Moreover, differences between ℓ_1- and ℓ_2-based objective functions are pronounced in the continuous domain, where ℓ_1 problems are consistently harder to optimize. On integer domains, this gap is reduced, suggesting that discreteness partially mitigates the geometric difficulties induced by non-smooth level sets. This observation aligns with the coordinate-wise nature of progress on the integer lattice, where late-stage optimization typically consists of correcting a small number of remaining coordinates.

Recombination: The choice of recombination operator plays a substantially more critical role in integer domains than in continuous ones. For DG-based IES variants, a

consistent performance hierarchy emerges: discrete recombination is most effective, followed by no recombination, while intermediate recombination consistently degrades performance. In contrast, for TN-based variants, this ordering is largely reversed—with intermediate recombination performing best—though the performance gap between operators is notably narrower. This behavior is rooted in the structural compatibility between the operator and the search space. While intermediate recombination exploits the continuity of real-valued spaces through weighted averaging, its application to integer lattices is fundamentally indirect. Because the arithmetic mean of two integers rarely yields an integer, the necessary rounding step introduces a quantization error that can distort selection pressure and perturb the self-adaptation of the step size. At the same time, adherence to Beyer's genetic repair hypothesis may be beneficial for Gaussian-based mutation distributions, such as the TN. Conversely, discrete recombination is inherently aligned with the lattice structure: by selecting coordinates from the parents, it preserves the integrity of the integer grid without requiring heuristic repair. This constitutes a *functional dissociation*, where two conditions (TN vs. DG) show opposite responses to the recombination type; recombination operators that preserve lattice structure are crucial when mutation itself is defined directly on the lattice. Notably, DG-based mutations are drawn from a non-normal discrete distribution and therefore cannot adhere to Beyer's hypothesis.

Zero Mutations: Allowing zero mutations introduces an additional effect specific to DG-based IES variants. Zero mutations are consistently beneficial when no recombination is used, despite producing offspring identical to their parent. This suggests that, for the considered non-elitist IES variants, the step-size adaptation may *lag* behind, requiring additional updates to converge toward appropriate values while fixing $\mathbf{x}$.

VII. Future Work

The empirical evidence for the effectiveness of non-normal mutations and discrete recombination operators for IESs provides clear motivation to further develop IESs. In particular, the observation of zero mutations calls for the derandomization of those IESs by carefully accounting for the ℓ_1 geometry and integrating appropriate genetic operators. In the continuous domain, derandomized evolution strategies rely on (weighted) *intermediate recombination* [4]. Given selected mutation steps $y_{1:\lambda}, \ldots, y_{\mu:\lambda}$ with weights w_i, the search center is shifted by $\bar{y} = \sum_{i=1}^{\mu} w_i y_{i:\lambda}$. This update preserves the expectation of successful steps and forms the basis for path-based adaptation mechanisms. On the integer lattice, however, the weighted mean $\bar{y}$ is generally not feasible. A direct projection back to $\mathbb{Z}^n$ introduces systematic bias, which may accumulate over generations. Moreover, our results indicate that, in combination with a DG mutation, intermediate recombination can be especially ineffective. A derandomized recombination operator for integer-valued search spaces would therefore need to satisfy two competing requirements: it should remain feasible on the lattice, while preserving the expectation and directional information of the continuous update. This suggests the need for projection or rounding operators $R : \mathbb{R}^n \to \mathbb{Z}^n$ that do not distort the mean update in expectation, i.e., satisfy $\mathbb{E}[R(\bar{y})] = \bar{y}$, and that are compatible with cumulative adaptation of the search distribution. Related ideas appear in the CMA-IH, which compensates for rounding effects through integer centering [6]. Exploring such operators and their interaction with discrete mutation distributions constitutes an important direction for future work.

References

[1] I. Rechenberg, *Evolutionsstrategie*. Stuttgart: Friedrich Fromman Verlag (Günther Holzboog KG), 1973.

[2] H.-P. Schwefel, *Numerical optimization of computer models*. John Wiley & Sons, Inc., 1981.

[3] A. Ostermeier, A. Gawelczyk, and N. Hansen, "A derandomized approach to self-adaptation of evolution strategies," *Evolutionary Computation*, vol. 2, no. 4, pp. 369–380, 1994.

[4] N. Hansen and A. Ostermeier, "Completely derandomized self-adaptation in evolution strategies," *Evolutionary computation*, vol. 9, no. 2, pp. 159–195, 2001.

[5] J. Klockgether and H. P. Schwefel, "Two-phase nozzle and hollow core jet experiments," in *Proc. 11th Symp. Engineering Aspects of Magnetohydrodynamics*, 1970, pp. 141–148.

[6] T. Marty, N. Hansen, A. Auger, Y. Semet, and S. Héron, "Lb+ ic-cma-es: Two simple modifications of cma-es to handle mixed-integer problems," in *International Conference on Parallel Problem Solving from Nature*. Springer, 2024, pp. 284–299.

[7] O. M. Shir and M. Emmerich, "Foundations of correlated mutations for integer programming," in *Proceedings of the 18th ACM/SIGEVO Conference on Foundations of Genetic Algorithms*, ser. FOGA '25. New York, NY, USA: Association for Computing Machinery, 2025, p. 285–296.

[8] A. Thomaser, J. De Nobel, D. Vermetten, F. Ye, T. Bäck, and A. Kononova, "When to be discrete: Analyzing algorithm performance on discretized continuous problems," in *Proceedings of the Genetic and Evolutionary Computation Conference*, ser. GECCO '23. New York, NY, USA: Association for Computing Machinery, 2023, p. 856–863.

[9] G. Rudolph, "An evolutionary algorithm for integer programming," in *Parallel Problem Solving from Nature — PPSN III*, Y. Davidor, H.-P. Schwefel, and R. Männer, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 1994, pp. 139–148.

[10] R. Li, M. T. M. Emmerich, J. Eggermont, T. Bäck, M. Schütz, J. Dijkstra, and J. H. C. Reiber, "Mixed integer evolution strategies for parameter optimization," *Evol. Comput.*, vol. 21, no. 1, p. 29–64, Mar. 2013.

[11] T. Bäck, *Evolutionary Algorithms in Theory and Practice*. New York, NY, USA: Oxford University Press, 1996.

[12] D. Sudholt, "How crossover speeds up building-block assembly in genetic algorithms," *Evolutionary Computation*, vol. 25, no. 2, pp. 237–260, 2017.

[13] J. H. Holland, *Adaptation in natural and artificial systems: An introductory analysis with applications to biology, control, and artificial intelligence*. Univ. of Michigan Press, 1975.

[14] H.-P. Schwefel, *Evolution and Optimum Seeking*. New York, NY, USA: John Wiley & Sons, Inc., 1995.

[15] D. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*. Reading, MA: Addison Wesley, 1989.

[16] H.-G. Beyer, "An Alternative Explanation for the Manner in which Genetic Algorithms Operate," *BioSystems*, vol. 41, no. 1, pp. 1–15, 1997.

[17] J. de Nobel, D. Vermetten, O. Shir, M. Emmerich, and T. Bäck, "Reproducibility files and additional figures," Jan 2026. [Online]. Available: https://doi.org/10.5281/zenodo.18428845

[18] M. López-Ibáñez, D. Vermetten, J. Dreo, and C. Doerr, "Using the empirical attainment function for analyzing single-objective black-box optimization algorithms," *IEEE Transactions on Evolutionary Computation*, 2024.

[19] D. Vermetten, J. Rook, O. L. Preuß, J. de Nobel, C. Doerr, M. López-Ibañez, H. Trautmann, and T. Bäck, "Mo-iohinspector: Anytime benchmarking of multi-objective algorithms using iohprofiler," in *International Conference on Evolutionary Multi-Criterion Optimization*. Springer, 2025, pp. 242–256.