Importance of Gradualism in Evolutionary Optimization for Finding Intermediate Transitional Solutions for Certain Problems

Ahmer Khan and Kalyanmoy Deb
{khanahm2,kdeb}@msu.edu
Michigan State University
East Lansing, MI, USA

COIN Report Number 2026009

Abstract

Population-based evolutionary computation (EC) algorithms have long drawn inspiration from natural evolution, yet one fundamental aspect is not often enforced in them: gradualism of populations over generations. Species do not leap to their optimal form in a single step; they transition through acceptable incremental states. In contrast, in most EC applications, the focus is often to judge how quickly and accurately the final optimal solution(s) are found and not how and with what gradualism these solutions were arrived at. This disconnect becomes critical in certain real-world problems where instead of just the final near-optimal solution, the focus is to discover how the currently practiced solution must be gradually transformed to the final solution. In a recent study, the so-called "innovation path" (IP) task and a novel bi-objective EC algorithm were proposed to find the ordered transformations. In this paper, we examine how the proposed IP-seeking algorithm utilizes the missing evolutionary principle in finding these transformations, and contrast it with the outcome of commonly-used EC algorithms. Through analytical discussions and illustrative scenarios, we show that only emphasizing converging to the optimal solution may not lead to the desired and acceptable transitional solutions. Our study highlights that if intermediate solutions are of importance, there is a need to rethink EC algorithms motivated by the gradual transitional aspect of natural evolution.

Keywords

Evolutionary optimization, transitional solutions, innovation path, point-based optimization.

1 Introduction

Population-based optimization methods have become foundational tools for solving complex design and decision problems. Their conceptual roots lie in natural evolution, where populations adapt through selection, variation, and inheritance. Yet, despite this inspiration, a crucial principle of biological evolution is largely absent from the mainstream evolutionary optimization applications: evolution proceeds through feasible, incremental transitions rather than abrupt jumps from generation to generation to an optimal form. Traditional applications focus their search to find a near-optimal solution, implicitly assuming that the path from the current to the final generation is irrelevant. This disconnect becomes evident in domains where intermediate generational solutions are as important as the final solution and abrupt reconfiguration in successive generations is not allowed. The land-use planning scenario in Figure 1 illustrates such a problem scenario. The "Current Solution" reflects a fragmented spatial allocation, while the "Optimized Solution" presents a more coherent arrangement. A conventional formulation treats this as a static mapping problem: identify the best final layout. However, real land-use transitions are constrained by zoning regulations, infrastructure continuity, economic inertia, and social acceptability. A sudden transformation from the current to the optimized configuration is not possible to achieve. What matters is a sequence of permissible and minimally disruptive steps that gradually reshape the landscape from current to the the final one.

This paper argues that the focus on final-state discovery fails to account for this transitional account. Most algorithms prioritize convergence as close to the

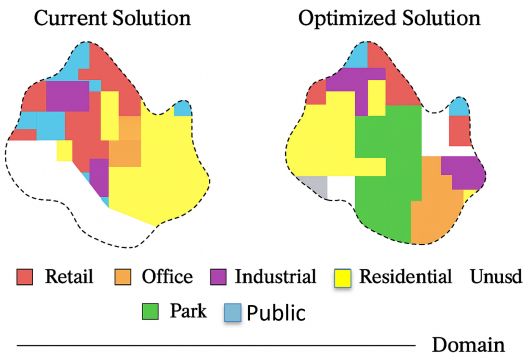
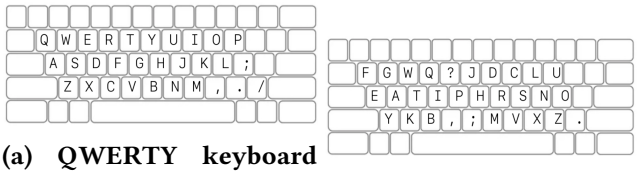


Figure 1: A current and optimal arbitrary land use map showing the difference between the two configurations.

optimal solution as possible, and they are evaluated solely on their convergence —- not on whether these solutions can be reached through a sequence of viable intermediate states. To address this gap, we build on the recently proposed Innovation Path Problem (IPP), which reframes optimization as a process of evolutionary transformation. IPP explicitly models the optimization task as a sequence of feasible transitions, each preserving structural integrity while incrementally improving performance. This formulation aligns more closely with the logic of natural evolution and with real-world problems that demand a gradual change.

2 Beyond the Final Solution: The Missing Evolutionary Path

Many design problems contain an implicit continuity constraint: the current configuration cannot be abandoned wholesale, even when a dramatically better alternative exists. Systems shaped by long-term use, habit, or infrastructure tend to accumulate structural inertia, making abrupt reconfiguration impractical regardless of how attractive the optimized end-state may be. Let us consider the design task of configuring an English-language keyboard. The most widely adopted layout is the QWERTY keyboard shown in Figure 2a, introduced commercially in 1878 for mechanical typewriters [28]. Its design is believed to have been influenced by the need to reduce jamming by separating frequently paired letters, rather than optimizing for ergonomic efficiency. Despite the obsolescence of mechanical constraints in modern electronic keyboards, QWERTY remains dominant, largely due to user familiarity and entrenched training practices. If cumulative finger movement during extended typing is considered a design criterion to be minimized, QWERTY performs poorly [17]. Figure 2b presents an optimized keyboard layout obtained using an evolutionary algorithm, designed to minimize total finger movement over a large corpus of English text. A direct comparison with the QWERTY layout (Figures 2a and 2b) reveals that 29 out of 30 keys differ between the two configurations. Such a drastic change renders the optimized layout impractical for immediate adoption, as it would require users to relearn nearly the entire keyboard -— an effort a few would undertake given the steep learning curve and disruption to productivity.



(a) QWERTY keyboard configuration with performance value: 178,779 (smaller is better)

(b) GA-optimized keyboard configuration with performance value: 76,933

Figure 2: Comparison of GA-optimized keyboard with the popularly used QWERTY keyboard. The latter keyboard causes a better performance, but is quite different from QWERTY, thereby discouraging users to jump to the better performing keyboard in a single step.

This challenge is not unique to the presented optimization. Historically proposed alternatives such as the Dvorak Simplified Keyboard (1930s) [12], Colemak (2006) [8], and Workman layouts have demonstrated ergonomic advantages over QWERTY, including reduced finger travel and improved typing comfort. However, none have achieved a widespread adoption. A probable cause might be the magnitude of change they impose: Dvorak alters nearly every key, while Colemak modifies 17 keys —- still a substantial cognitive and behavioral shift. These layouts, though optimal in performance metrics, suffer from the same adoption barrier illustrated by the optimized layout in Figure 2b: too many changes introduced at once, akin to the Japenese *Kaikaku* principle.

The keyboard example highlights a deeper structural issue: the practical difficulty does not stem from the optimized layout itself but from the absence of a feasible progression connecting the current configuration to that final design. An optimal solution that cannot be approached through a sequence of viable and gradual intermediate states (the Japanese *Kaizen* principle) remains effectively inaccessible, regardless of its performance advantages. This disconnect clarifies why focusing solely on the discovery of the final solution may provide an incomplete picture; without an accompanying path that respects continuity and adoptability, optimization risks producing outcomes that are theoretically superior yet operationally unattainable. We return to nature for an inspiration of graduality among consecutive generations in redesigning evolutuionary optimization algorithms.

3 Related Past Studies

Evolutionary algorithms have long drawn inspiration from natural evolution —- selection, variation, inheritance, and survival of the fittest. These principles are abstracted into computational frameworks that iteratively refine candidate solutions over generations. Yet, despite their biological origins, most evolutionary algorithms operationalize evolution as a static optimization task: the objective is to discover a final, globally optimal solution, not to model the gradual, transitional dynamics characteristic of natural evolutionary change. Foundational works such as [15, 16, 23] formalize this paradigm, emphasizing convergence toward optimality. Contemporary surveys and algorithmic developments continue this trajectory, prioritizing solution quality and convergence speed and Pareto front approximation for multi-objective optimization [9, 13]. Although biologically inspired, these methods do not capture the incremental, path-dependent nature of natural evolution, where each intermediate form must be viable within its own context. This absence of natural gradualism in both problem formulation and algorithmic design is central to the conceptual gap addressed in this study.

Within the EMO community, multi-objectivization has emerged as a powerful strategy for improving search flexibility by reformulating problems that are not inherently multi-objective [21]. A well-studied example is constrained single-objective optimization, where the challenge lies in minimizing a primary objective while satisfying strict feasibility requirements. In highly constrained landscapes, classical single-objective methods often struggle to balance objective improvement with constraint satisfaction [9, 26], typically relying on weighted penalty functions that require careful tuning. EMO-based approaches address this by treating constraint violation as a second objective, enabling simultaneous minimization of both the primary objective and the violation measure [7, 11, 22]. This separation of concerns yields a more interpretable and flexible search process [11]. Similar multi-objective formulations have been applied to keyboard layout optimization, where conflicting goals such as typing efficiency and layout familiarity must be jointly optimized [4, 27]. However, these studies, like most EMO studies, focus on discovering a set of trade-off solutions, not on constructing a sequence of feasible transitions from an incumbent design to a target configuration. They improve the diversity of final outcomes but do not introduce any mechanism for gradual and controlled transformation.

Outside computational optimization, the importance of incremental change has long been recognized in industrial and organizational practice. The Kaizen philosophy of continuous, step-wise improvement [3] has repeatedly proven more sustainable and less disruptive than the Kaikaku approach of radical overhaul [14]. While Kaikaku is appropriate when a complete system redesign is unavoidable [14, 24], it often incurs substantial operational disruption, retraining costs, and resistance to change. Kaizen, by contrast, has been widely adopted across manufacturing, product development, and process engineering as a pragmatic strategy for long-term innovation [1–3, 6, 25]. Despite its practical success, the literature lacks a general algorithmic framework capable of identifying and sequencing such incremental improvements.

Existing optimization methods can propose alternative end-state solutions, but they do not provide the structured, feasible transformation paths required for gradual, adoptable change.

However, it is important to note that not all optimization tasks care about the intermediate transitional solutions to the final outcome, but in certain problems, they are important. We discuss one such practical scenario in the following section.

4 Innovation Path: Beyond A Set of Alternative Solutions

Classical optimization techniques – whether linear, nonlinear, or heuristic – are fundamentally designed to identify a solution that optimizes a given objective function under a set of constraints. This solution, often referred to as the *global optimum*, is typically pursued without regard for the currently implemented (CI) solution or the practical feasibility of transitioning from it. As formalized in most optimization models, the goal is to find an optimal point $x^* \in X$ such that $f(x^*) \leq f(x)$ for all $x \in X$, where X is the feasible region and f is the objective function [5]. This formulation implicitly treats optimization as a one-step replacement problem: identify the best solution and assume it can be adopted directly, as shown with the keyboard example earlier.

The Innovation Path (IP) framework [19] challenges this assumption by redefining optimization as a process of guided transformation rather than isolated end-state discovery. To introduce gradualism into the search process, and recognizing that multi-objective optimization naturally can naturally yield a series of trade-off solutions, rather than a single endpoint solution, the IP problem was formulated as a bi-objective optimization task, referred to as the IP Problem (IPP):

$$\begin{aligned} \text{Minimize} \quad & \{s(\mathbf{x}), f_D(\mathbf{x})\}, \\ \text{subject to} \quad & \widehat{G}_l(\mathbf{x}, \mathbf{X}_k^*) \leq 0, \quad \forall k = 1, \ldots, |\mathbf{X}_:^*|, \quad l = 1, 2, \ldots, L, \\ & \mathbf{x} \in \mathcal{X}. \end{aligned} \tag{1}$$

which seeks for a finite set of anchor solutions $\mathbf{X}^*$. Here $s(\mathbf{x})$ is the desired goal function and $\mathcal{X}$ is the set of feasible solutions of the target problem satisfying the problem constraints: $g_j(\mathbf{x}) \leq 0$, for $j = 1, \ldots, J$. While $f_D(\mathbf{x})$ is the difference between the variable vector $(\mathbf{x})$ and the prescribed current feasible solution $(\mathbf{x}^C)$, to be minimized. The cardinality of the anchor set $\mathbf{X}^*$ corresponds to the number of intermediate states along the IP.

Notice from the formulation that, unlike in a standard numerical optimization problem, an IPP constraint (referred as a step-constraint) $\widehat{G}_l$ is a function of two variable vectors: $\mathbf{x}$ and the k-th anchor solution already discovered so far. Each solution $\mathbf{x}$ must differ from all obtained anchors by a bounded, non-trivial amount of change. This transforms the problem from static optimization into a structured sequence of transitions, where each intermediate configuration is locally optimal under the prescribed step constraints. These constraints ensure that every transition introduces a meaningful progress while preventing the algorithm from collapsing into infinitesimal or redundant updates.

The step-constraints introduce substantial challenges to an algorithm in solving the IPP. First, as mentioned, $\widehat{G}_l$ is a function of two variable vectors. Second, the step-constraint is a dynamically changing constraint, as for a fixed $\mathbf{x}$, it's value depends on the previously-found anchor points $\mathbf{X}_k^*$. Unless the anchor points have stabilized, step-constraints stay dynamic and can introduce noise in its evaluation. This dependence on the step-constraint value on stabilization of past anchors makes the IPP challenging to solve.

Additionally, by enforcing bounded, feasible transitions, the IPP formulation mirrors the incremental dynamics observed in natural evolution. Each new configuration emerges from its predecessor under the joint pressures of performance improvement and continuity preservation. Hence, the outcome is not a single optimal solution, rather a realizable path of intermediate solutions, each adoptable, each feasible, and collectively bridging the gap between the current system and its desired future form.

5 Limitations of Current Optimization Algorithms

The requirement of producing a gradual, feasible transformation naturally raises two questions: Can classical point-based algorithms generate such a path, and if not, can population-based evolutionary algorithms do so with a multi-objective formulation? To address both questions, the following subsections present empirical demonstrations that illuminate the limitations of each class of methods.

5.1 Point-based Optimization Approaches

Point-based methods might appear promising because they update a single solution iteratively, suggesting the possibility of tracing a continuous trajectory. However, their intermediate states are determined entirely by algorithmic update rules—such as descent directions or quasi-Newton steps and not by any trade-off between improvement and controlled deviation.

Let us consider the minimization of the Himmelblau's function in which the search space is the first coordinate ($x_1 \geq 0$, $x_2 \geq 0$). If we start with an initial point $x^{(0)} = (5, 5)$, we can apply an optimization algorithm to find the minimum ($\mathbf{x}^* = (3, 2)$). Matlab's `fminsearch()` algorithm (implementing Nelder and Meade optimization algorithm) shows the iteration-wise intermediate points in Figure 8a in red circles. The intermediate points do not follow any specific trade-off among objective improvement and graduality in solutions. The intermediate points are outcomes of the update principles of the optimization algorithm laid out at the time of designing the algorithm. As long as the intermediate points finally lead to a near-optimal solution, the purpose of the optimization process is served. However, in an IPP, we would like the current solution to transform into the optimal solution in a series of ordered solutions that strikes an optimal trade-off between $s(\mathbf{x})$ and $f_D(\mathbf{x})$. For the sake of a better understanding, we mark the non-dominated intermediate points with filled circles (minimizing the above objectives) and dominated ones in open circles.

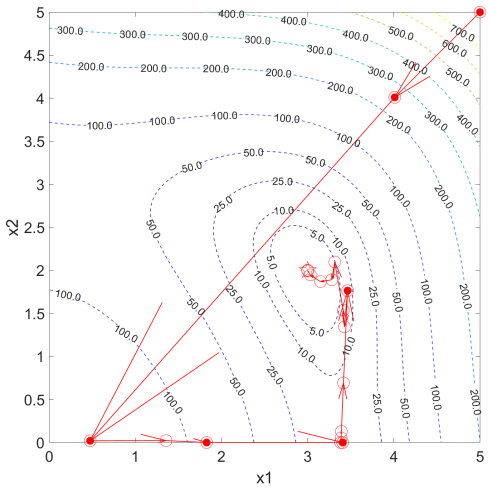


Figure 3: Obtained intermediate solutions for the Himmelblau problem by a point-based optimization routine `fminsearch()` of MATLAB.

This task naturally leads to the belief that trust-region methods might offer a remedy, since they explicitly restrict the step size between successive iterates and therefore appear more compatible with the notion of gradual transformation. However, this resemblance is only superficial. Although the trust-region radius may look like a step-constraint boundary, it is always defined around the current iterate and in the variable space, which prevents the algorithm from accumulating continuity across iterations or building upon a structured sequence of previously accepted solutions. This limitation is also compounded by the fact that trust regions operate solely in the variable space, whereas the IPP requires gradualism in a problem-specific deviation space, such as the number of keys moved, components altered, or structural changes introduced. A small step in variable space may therefore correspond to a large or negligible change in the deviation metric, offering no meaningful control over the transformation process. Moreover, trust-region iterates are internal artifacts of the algorithm, chosen exclusively to improve the objective function rather than to balance improvement against deviation from the current solution. Even when the trust-region radius is tuned to enforce small steps, the resulting intermediate points are not optimized for adoptability or feasibility and thus do not constitute valid IPs. In short, trust-region methods can mimic the appearance of gradual movement but cannot satisfy the semantic, bi-objective requirements of the IPP.

5.2 Population-based EC Optimization

By definition, the IPP seeks a sequence of solutions that lie on the Pareto front of a bi-objective optimization problem: one objective captures performance improvement, while the other penalizes deviation from the current configuration. At first glance, this formulation resembles a conventional multi-objective optimization problem, and one might assume that any standard EMO algorithm could be applied directly. Indeed, we can cast the problem into the familiar EMO setting and solve it as such.

5.2.1 Post-optimization Filtering. To illustrate this, we revisit the keyboard example introduced in Section 2, where $s(\mathbf{x})$ measures the cumulative finger movement while typing and $f_D(\mathbf{x})$ quantifies the number of keys repositioned relative to QWERTY [17].

Solving this relaxed formulation with a conventional EMO method (NSGA-II here) yields a Pareto front representing trade-offs between efficiency and deviation, as shown in Figure 4. Each point corresponds to a layout that is individually optimal for a specific deviation

level, for example, the solutions marked as 2, 4, and 8 key changes. While some layouts may, by coincidence, exhibit partial continuity, this is neither expected nor enforced. As illustrated in Figure 5, the solutions on the Pareto front arise independently of one another: the algorithm has no mechanism to ensure that Pareto-optimal points are found in any sequence and differ by a controlled or meaningful amount. Any apparent progression is therefore incidental rather than structural. Consequently, the Pareto front produced by classical EMO methods is a collection of alternative optima, not a guaranteed sequence of feasible transitions. Hence, the belief to get an IP by post-optimization filtering of the PO set obtained by a standard NSGA-II run would not guarantee a continuous optimal transitional path.

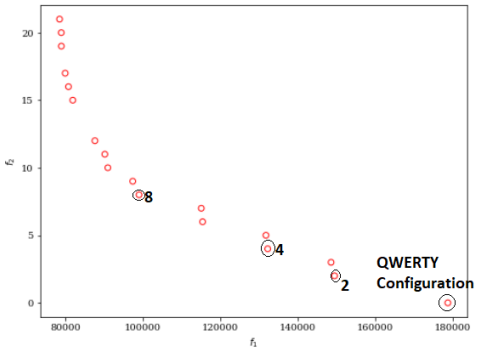
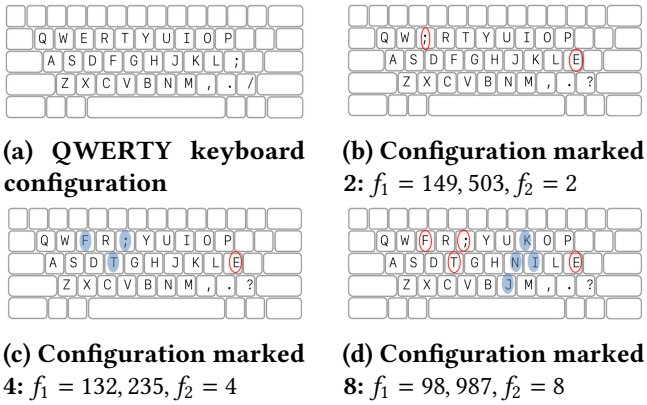


Figure 4: Obtained Pareto front using NSGA-II for the multi-objective version of the keyboard optimization problem with some randomly marked solutions for specific changes.

5.2.2 Filtering based NSGA-II. Since we did not explicitly use the IP constraints in the above example, it may seem the sole reason for the absence of an ordered path. To demonstrate this further, we extend the original NSGA-II [10] with a slight modification, as shown in Figure 6a. First, the existing and known current solution $\mathbf{x}^{\mathrm{C}}$ is included in the initial population. Second, all non-dominated (ND) solutions corresponding to the two proposed objectives – minimization of $s(\mathbf{x})$ and $f_D(\mathbf{x})$, which satisfy all step-constraints given in Equation 1 are chosen as the first ND front. The ND rank of all other solutions is increased by one. The rest of the NSGA-II procedures are kept as they are. Figure 6b shows the innovation path discovered in a single run. While the path begins at the current solution and progresses toward better performance, it fails to reach the



(a) QWERTY keyboard configuration

(b) Configuration marked 2: $f_1 = 149,503, f_2 = 2$

(c) Configuration marked 4: $f_1 = 132,235, f_2 = 4$

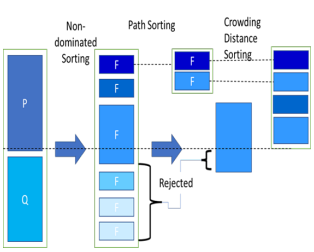
(d) Configuration marked 8: $f_1 = 98,987, f_2 = 8$

Figure 5: Highlighted solution from the NSGA-II Pareto front (Figure 4) with specific changes from the QWERTY configuration marked and inter-configuration changes filled with blue.

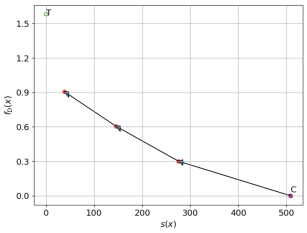
exact target solution. This might be because the approach is still limited by its intermediate anchor discovery mechanism. As it only looks for anchors in the first non-dominated set, and is dependent on the solutions available in the front, resulting in an incomplete path or a path with big jumps in between.

To gain deeper insight into internal dynamics, we first identify a set of ideal anchors by computing the Pareto front of the bi-objective formulation and selecting solutions that satisfy the step constraint $\widehat{G}_1(\mathbf{x}, \mathbf{x}^{(k)}) \equiv \left[1 - \left(f_D(\mathbf{x}) - f_D(\mathbf{x}^{(k)})\right)\right]/\Delta_1 \leq 0$ with $\Delta_1 = 0.3$, beginning from the current solution. Figure 7a presents the Euclidean distance between the anchors discovered during each generation and their corresponding ideal counterparts in the bi-objective space. Notably, the discovery process lacks a consistent order: for instance, anchor 3 is identified before anchor 1, and then both converge simultaneously. To assess the consistency of this behavior, we conduct a second run with an extended generation limit ($t_{max} = 200$), as shown in Figure 7b. In this case, the discovered sequence is different: anchor 2 emerges first, while anchor 1 converges last. Across both runs, we observe a recurring pattern: the algorithm does not exhibit any directional pressure to discover subsequent anchors following the convergence of earlier ones. In both instances, the fourth anchor fails to converge at all.

To further probe this phenomenon, we increase the population size ($N = 200$) and perform an additional



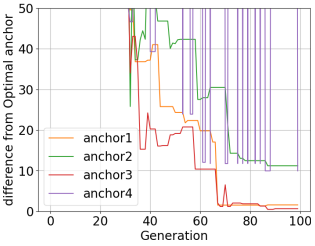
(a) Filtering based NSGA-II for handling IPP constraints



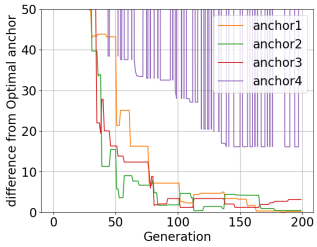
(b) IP found by filtering based NSGA-II, for the 10-variable Rosenbrook problem

Figure 6: The filtering based NSGA-II structure and its generated IP for the 10-variable Rosenbrook problem.

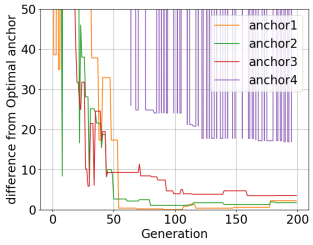
run. While the order of anchor discovery again varies, the fourth anchor remains undiscovered.



(a) Anchor discovery for $N = 100, t_{max} = 100$



(b) Anchor discovery of for $N = 100, t_{max} = 200$



(c) Anchor discovery for $N = 200, t_{max} = 200$

Figure 7: Euclidean distance of anchors discovered in the filtering based NSGA-II run from ideal IP anchors for the Rosenbrook problem with 10 variables across different maximum generations (t_{max}), population size (N), and initial population.

Taken together, these examples illustrate that neither point-based nor population-based optimization algorithms, in their current form, possess an automated mechanism required to generate a coherent transformation path. Their intermediate states arise from update rules designed solely to reach an optimum, not to regulate the magnitude or structure of change between successive iterations. Enforcing such gradualism would therefore require modifications to the underlying principles of these algorithms -— alterations that go beyond parameter tuning or operator selection. While no general framework exists for point-based algorithms, at least one population-based extension that incorporates step-wise feasibility has been proposed previously [20]. The next section provides a brief overview of this approach, which offers a possible route toward constructing viable innovation paths.

6 Existing IP-Seeking Algorithm

Here, we present a brief description of the IP-BCGA procedure [20]. To enable IP-EMO to identify an ordered set of solutions, the ranking, survival, and selection operators of the original NSGA-II had to be redefined. The crowding distance operator was also replaced with a new measure called 'Association' (α) – smaller α indicating closer association, and a new definition of *directed domination* was introduced.

To rank solutions across fronts at each generation, the first front comprises of solutions that satisfy step constraints and are non-dominated under the directed domination definition [18]. These solutions are referred to as *anchors*. The remaining solutions are associated with specific anchors [19]. Once the population is ranked and associated, anchors are given priority during the acceptance phase for the next generation. All other solutions are subsequently accepted, front by front, based on their association with anchors and the degree of step-constraint violation defined in [18].

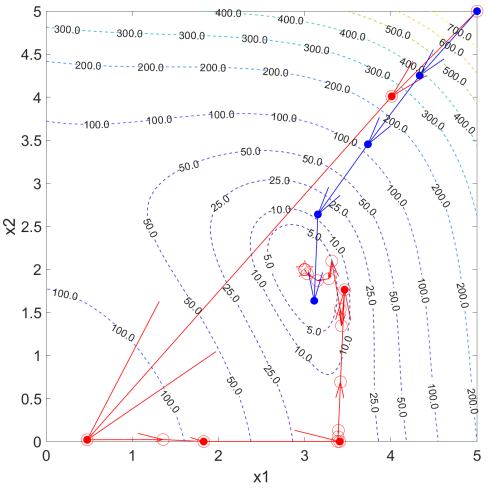
6.1 Representative Results

We compare IP-BCGA to the stated approaches in Section 5 and show how IP-BCGA finds the transitional optimal paths on Himmelblau, keyboard, and Rosenbrock problems.

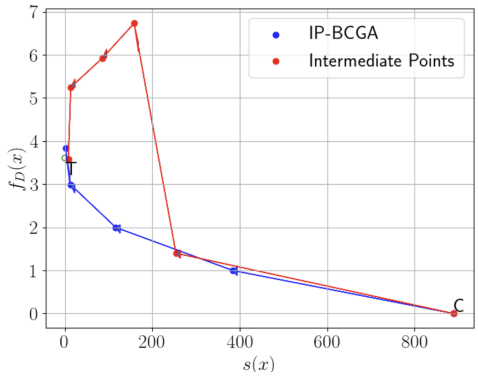
6.1.1 Point-based algorithm on Himmelblau Problem. When IP-BCGA is applied to the Himmelblau problem, starting with the same initial point as the current point and a step constraint $\widehat{G}_1(\mathbf{x}, \mathbf{x}^{(k)}) \equiv \left[1 - \left(f_D(\mathbf{x}) - f_D(\mathbf{x}^{(k)})\right]/\Delta_1 \leq 0$ with $\Delta = 1$, four anchor

points are found, shown in filled blue circles in Figure 8. First, notice that the IP-anchor solutions are more organized. Second, notice when these points are plotted in the two-objective space (Figure 8b), the IP-anchor solution set dominates mostly the `fminsearch()`-obtained intermediate points (Figure 8a). This clearly states that a typical optimization algorithm focuses on finding the final optimal solution rather than on a gradual transformation of the current solution into that final solution through a sequence of viable intermediate steps. Consequently, simply extracting the intermediate solutions generated during an optimization run, even if some appear non-dominated, does not yield the desired trade-offs or a meaningful transformation path. Such an algorithm was never tasked with producing anchor points, and its update rules do not encode the goals required by the IP problem. To obtain intermediate solutions that genuinely balance improvement and controlled deviation, the optimization setting itself must be reformulated, and the core principles of the algorithm must be modified accordingly.

6.1.2 Post-optimization filtering on Keyboard Layout Problem. For the keyboard layout problem, we use a population size of 100 and run for 100 generations with $\Delta_1 = 4$. Figure 9 shows the IP solutions starting from the QWERTY layout at the bottom right corner of the plot. The blue points show the IP solutions found by IP-BCGA, while the red points show the solutions found by filtering-based NSGA-II satisfying the $\widehat{G}_1$ IPP constraint. Figure 10 shows all four keyboard configurations found by IP-BCGA. It can be seen how the number of gradual changes at each step is made to arrive at the final configuration, thereby reducing the typing efficiency from 180,000 to about 80,000. Four key changes are made from one IP solution to the next to make the transition in achieving maximum typing efficiency improvement in all consecutive pairs of IP solutions in one application of IP-BCGA. We can also see how the path is turning asymptotic at the end, meaning we have to make large changes for a small improvement beyond the discovered final layout, which might not be suitable, hence the IP stops. In comparison to the post optimization filtering points the IP found by IP-BCGA improves a lot further and even dominates the post optimization filtering solutions available. As stated earlier in Section



(a) Variable space



(b) Bi-objective space

Figure 8: Intermediate points of the point-based approach (red) and anchors of the IP-BCGA algorithm (blue) in (a) variable space and (b) objective space, for the Himmelblau problem. Intermediate points filtered after `fminsearch()` optimization are inferior to IP-anchors in the context of trade-off between improvement in $s(\mathbf{x})$ and difference ($f_D(\mathbf{x})$) from the current point.

5.2, the post optimization IP is dependent on the solutions available in the PO set and cannot go any further.

6.1.3 Filtering based NSGA-II on Rosenbrock Problem. Next, we demonstrate the finding of IP-BCGA for the 10-variable Rosenbrock problem for the same setting as used for the filtering-based NSGA-II in the section 5.2. Figure 11 shows the path found by IP-BCGA in blue in addition to the red path found by the filtering-based

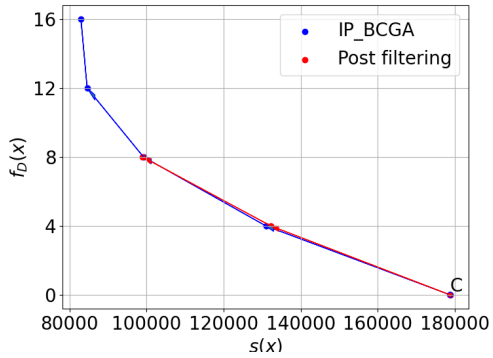
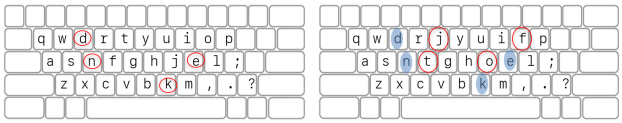
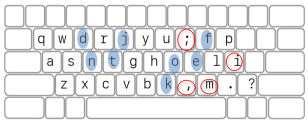


Figure 9: IP in IPP space for keyboard problem compared between IP-BCGA and post-optimization filtering NSGA-II.

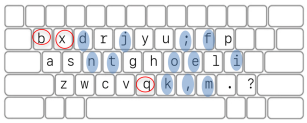


(a) Config. 1: 4 key differences from C

(b) Config. 2: 8 diff. from C, 4 from (a)



(c) Config. 3: 12 diff. from C, 4 from (b)



(d) Final: 16 diff. from C, 4 from (c)

Figure 10: Keyboards on IP obtained by IP-BCGA starting from QWERTY.

NSGA-II algorithm. It can been seen the blue points overall dominate the red points and form a complete path reaching closer to the target 'T'.

7 Discussion

Current evolutionary computation frameworks are built around the assumption that optimization is a destination-oriented task. They are designed to locate an optimal end state, not to articulate how a system can move from its present configuration to that improved state through a sequence of viable steps. This structural omission arises from the problem formulation itself, which encodes no representation of continuity, controlled deviation, or step-wise feasibility. As a result, no algorithm operating within this setting, regardless of its sophistication, can naturally produce gradual and adoptable transitions.

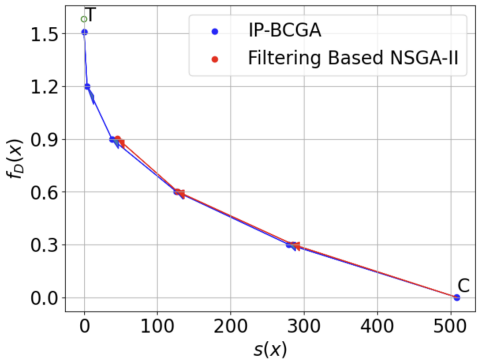


Figure 11: IP found by the filtering-based NSGA-II algorithm and IP-BCGA on 10-variable Rosenbrock problem.

Introducing gradualism into optimization therefore requires rethinking both the problem definition and the algorithmic principles that follow from it. The IPP and the IP-BCGA algorithm illustrate one way to embed deviation-aware reasoning into the search process, but they represent only one possible direction. Other approaches may be equally promising, e.g., new dominance relations could be designed to incorporate deviation metrics directly into selection, enabling existing EMO algorithms to prioritize solutions that are not only high-performing but also compatible with a structured transformation path. Such a dominance principle would allow populations to self-organize around feasible transitions rather than merely around Pareto-optimal endpoints. Similarly, deviation-aware variation operators, path-preserving archives, or multi-stage selection strategies could be explored as modular mechanisms for injecting gradualism into standard evolutionary frameworks. These ideas explore a more general research agenda on how gradualism might be integrated into existing evolutionary algorithms without fully redesigning them.

The need for such a shift in focus is increasingly evident across a wide range of real-world scenarios where abrupt replacement of a system is infeasible. Land-use planning, for example, requires transitioning from current zoning or ecological configurations toward more sustainable layouts while respecting legal, social, and environmental constraints at every stage. Mega-construction and infrastructure projects, such as the development of smart cities, must coordinate thousands of interdependent components over long time horizons, where redesigns must occur incrementally

to maintain feasibility and continuity. Operational domains such as scheduling, maintenance planning, and workforce allocation similarly demand updates that preserve stability and resource commitments rather than discarding the existing schedule entirely. Even policy implementation often requires gradual roll-outs to maintain institutional stability and public acceptance. In all these contexts, the ability to compute structured, feasible transformation paths is not merely a theoretical curiosity but a practical necessity.

Together, these observations suggest that gradualism should be treated as a primary concept in optimization, not an incidental by-product of algorithmic iterations. Developing a systematic framework for IPs, whether through new problem formulations, new dominance definitions, or new evolutionary operators, represents a promising direction for future research and remains as a critical step toward addressing the increasingly dynamic and transition-constrained problems encountered in practice.

8 Conclusions

This work has demonstrated that current optimization paradigms lack the structural elements required to generate Innovation Paths – sequences of feasible and minimally disruptive transitions from a current solution to an improved one. Through illustrative examples and diagnostic analyses, we have shown that the absence of gradual evolution is not an incidental limitation of specific algorithms, but a structural consequence of the underlying optimization setting. When continuity, controlled deviation, and step-wise feasibility are not encoded in the problem definition, they cannot emerge in the algorithmic behavior. Addressing this gap requires re-framing the optimization task itself so that the discovery of viable transformation paths becomes an explicit goal rather than an accidental by-product of iterative search.

To address this gap, we have examined the Innovation Path Problem (IPP) and evaluated the Innovation Path Bi-Criteria Genetic Algorithm (IP-BCGA), an existing algorithmic solution designed to embed gradualism into the search process. Comparative results with point-based and two filtering-based optimization methods have demonstrated that IP-BCGA can handle the limitations of these approaches by balancing objective improvement with bounded deviation, thereby producing step-wise, adoptable transitions. Point-based intermediate solutions are generated solely as stepping stones toward the final optimum, so they often bear no resemblance to a coherent path and rarely align with the structure of an Innovation Path. Post-optimization filtering is further constrained by whatever solutions happen to lie on the approximated Pareto front, which can leave large gaps, discontinuities, or even the complete absence of the desired IP. Even filtering-based NSGA-II, though slightly more integrated, still operates as a post-hoc selection mechanism; they may extract something that looks like a path, but there is no guarantee that it will be complete, continuous, or located near the true anchors that define the intended transformation trajectory.

Looking ahead, the need for path-aware optimization is likely to grow across diverse domains. Many real-world systems -— land-use configurations, large-scale construction projects, scheduling processes, and policy implementations -— cannot be replaced wholesale. They must evolve through sequences of feasible, adoptable steps that respect operational, social, or institutional constraints at every stage. Developing optimization methods capable of generating such structured transitions is therefore not only a theoretical challenge but a practical necessity.

References

[1] H. Abdulmouti. The role of kaizen (continuous improvement) in improving companies' performance: A case study. In *2015 International Conference on Industrial Engineering and Operations Management (IEOM)*, pages 1–6. IEEE, 2015.

[2] M. H. Al-Rifai. Redesigning and optimizing an electronic device assembly cell through lean manufacturing tools and kaizen philosophy: an application case study. *International Journal of Productivity and Performance Management*, 73(4):1273–1301, 2024.

[3] A. Berger. Continuous improvement and kaizen: standardization and organizational designs. *Integrated manufacturing systems*, 8(2):110–117, 1997.

[4] X. Bi, T. Ouyang, and S. Zhai. Both complete and correct? multi-objective optimization of touchscreen keyboard. In *Proceedings of the SIGCHI conference on human factors in computing systems*, pages 2297–2306, 2014.

[5] C. Carissimo and M. Korecki. Limits of optimization. *Minds and Machines*, 34(1):117–137, 2024.

[6] D. Carnerud, C. Jaca, and I. Bäckström. Kaizen and continuous improvement–trends and patterns over 30 years. *The TQM Journal*, 30(4):371–390, 2018.

[7] C. A. C. Coello. Treating objectives as constraints for single objective optimization. *Engineering Optimization*, 32(3):275–308, 2000.

[8] S. Coleman. The colemak keyboard layout. https://colemak.com/, 2006. Accessed October 2025.

[9] K. Deb. *Multi-Objective Optimization Using Evolutionary Algorithms*. Wiley, Chichester, UK, 2001.

[10] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan. A fast and elitist multi-objective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2002.

[11] K. Deb and R. Datta. A bi-objective constrained optimization algorithm using a hybrid evolutionary and penalty function approach. *Engineering Optimization*, 45(5):503–527, 2012.

[12] A. Dvorak, N. Dealey, and W. Ford. *Typewriting Behavior: Psychology Applied to Teaching and Learning Typewriting*. American Book Company, 1936.

[13] A. E. Eiben and J. E. Smith. *Introduction to Evolutionary Computing*, volume 53. Springer, 2003.

[14] D. Gåsvaer and J. von Axelson. Kaikaku - Radical improvement in production. In *International Conference on Operations and Maintenance, Singapore*, pages 37–47, 2012.

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

[16] J. H. Holland. *Adaptation in Natural and Artificial Systems*. Ann Arbor, MI: MIT Press, 1975.

[17] A. Khan and K. Deb. Optimizing keyboard configuration using single and multi-objective evolutionary algorithms. In *Proceedings of the Companion Conference on Genetic and Evolutionary Computation*, pages 219–222, 2023.

[18] A. Khan and K. Deb. Towards an efficient innovation path seeking algorithm using directed domination. Technical Report COIN Report Number 2024007, Michigan State University, East Lansing, USA, 2024.

[19] A. Khan and K. Deb. Gradual innovative transitional solutions improving current to a desired target: Innovation path. *IEEE Transactions on Evolutionary Computation*, pages 1–1, 2025.

[20] A. Khan and K. Deb. Hierarchical convergence to multiple alternate solutions: Population versus point-based algorithms. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, pages 563–566, 2025.

[21] J. D. Knowles, D. W. Corne, and K. Deb. *Multiobjective problem solving from nature*. Springer Natural Computing Series, Springer-Verlag, 2008.

[22] E. Mezura-Montes and C. A. C. Coello. Constraint-handling in nature-inspired numerical optimization: past, present and future. *Swarm and Evolutionary Computation*, 1(4):173–194, 2011.

[23] M. Mitchell. *Introduction to Genetic Algorithms*. Ann Arbor, MI: MIT Press, 1996.

[24] W. A. Moi and S. H. Sing. Application of toyota way incorporating kaizen, kaikaku and 5s in agricultural sector. *International Journal for Research in Applied Science and Engineering Technology*, 9(10):1565–1579, 2021.

[25] Y. Qu, X. Ming, S. Qiu, Z. Liu, X. Zhang, and Z. Hou. Process optimization through closed-loop kaizen with discrete event simulation: A case study in china. *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, 235(3):568–579, 2021.

[26] G. V. Reklaitis, A. Ravindran, and K. M. Ragsdell. *Engineering Optimization Methods and Applications*. New York : Wiley, 1983.

[27] J. Shen, J. Hu, J. J. Dudley, and P. O. Kristensson. Personalization of a mid-air gesture keyboard using multi-objective bayesian optimization. In *2022 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 702–710. IEEE, 2022.

[28] H. Yamada. *A Historical Study of Typewriters and Typing Methods: From the Position of Planning Japanese Parallels*, volume 2. J. of Inform. Proces., 1980.