A particle swarm optimization with a reduced-subspace trust-region safeguard for contact-rich controller tuning

Morteza Kimiaei[a], **Roya Khalili-Amirabadi**[b], **Mohsen Jalaeian-Farimani**[c]

[a] *Faculty of Mathematics, University of Vienna, 1090 Vienna, Austria*
Email: kimiaeim83@univie.ac.at
WWW: http://www.mat.univie.ac.at/~kimiaei

[b] *Department of Applied Mathematics, Ferdowsi University of Mashhad, Mashhad, Iran*
Email: roya.khalili.a@gmail.com

[c] *Department of Electronics, Information and Bioengineering (DEIB), Politecnico di Milano, Milan, Italy*
Email: mohsen.jalaeian@polimi.it
WWW: https://www.deib.polimi.it/eng/people/details/2012404

Abstract. Contact-rich robotic insertion is challenging because uncertainty in alignment, friction, and contact conditions can lead to unsuccessful assembly, excessive contact forces, and jamming. Designing a reliable controller therefore requires balancing insertion performance, robustness, and recovery from unexpected contact. In this paper, we propose `PSO+TR`, a derivative-free optimization (DFO) method that combines the exploratory search of particle swarm optimization (`PSO`) with a trust-region (`TR`) procedure for local refinement to improve the search process when progress stagnates. We apply the method to derivative-free policy optimization of an interpretable robotic insertion controller, jointly tuning its motion, feedback, and recovery parameters while preserving its predefined control structure.

The proposed approach is evaluated on 93 eligible benchmark problems from the bound-constrained programs (`BCP`) collection and a synthetic contact-rich planar insertion task under uncertain operating conditions. Three random seeds per benchmark problem yield 279 matched problem–seed instances per method. `PSO+TR` reaches the prescribed target on 228 of these instances (81.72%), the highest observed target-attainment rate among the five compared methods, which also include `PSO`, randomized black-box optimization (`VRBBO`), matrix adaptation evolution strategy (`MAES`), and differential evolution (`DE`). Across eight robotic optimizer seeds, the deployed controller-selection pipeline achieves 97.78% completion under development conditions and 58.00% under withheld stress conditions, compared with 91.22% and 31.12%, respectively, for the fixed staged controller. The stress violation rate is reduced from 20.69% to 2.19%.

These findings demonstrate the potential of combining global and local search for DFO and structured contact-rich controller tuning. The robotic results are limited to a synthetic planar simulator, and further experiments are required to establish performance on physical robots.

Keywords: Particle swarm optimization; derivative-free optimization; trust-region methods; adaptive subspace technique; derivative-free policy optimization; contact-rich robotic assembly; controller tuning.

2020 Mathematics Subject Classification. 90C56 (Primary); 90C15, 90C59, 93C85 (Secondary).

1 Introduction

Robotic assembly requires motion to remain effective as a robot interacts with its environment. During insertion, small differences in alignment, friction, or clearance can determine whether a part moves freely, contacts a rim, or becomes stuck. Accurate motion under nominal conditions is therefore only one part of the control problem. The controller must also accommodate uncertain interaction conditions while balancing progress and contact loading.

Even when a suitable feedback structure is available, selecting its numerical parameters can be difficult. Motion amplitudes, controller gains, and timing decisions jointly influence the resulting behavior. A setting that shortens completion time for a well-aligned part may increase contact loading when alignment is poor. Conversely, a more conservative setting may reduce loading but slow progress or prevent completion. These interactions make parameter selection a coupled problem: improving one aspect of the task can degrade another.

The information needed for tuning is also costly to obtain. Completion, elapsed time, and contact loads emerge over an entire closed-loop trajectory rather than from a single control update. Simulation or experimental trials can reveal these outcomes, but the number of trials available is limited by computational resources or experimental effort. Changes in contact and discrete controller decisions can further complicate the relationship between parameters and performance. Effective tuning must therefore extract useful information from a limited number of evaluations.

Uncertainty adds a further requirement: favorable performance on the conditions used during tuning does not by itself establish reliable behavior under different conditions. Performance criteria must account for both efficient motion and successful task completion, and observed improvements must be assessed across relevant operating conditions. This creates two connected questions: how efficiently can a search procedure identify promising parameter settings, and how consistently does the resulting controller perform beyond the trials that guided their selection?

1.1 Related work

To position the proposed method, we review population and subspace optimization alongside contact-rich controller tuning and identify the combination studied here. We first discuss black-box/derivative-free optimization (BBO/DFO) methods, with particular emphasis on population-based search and trust-region strategies. We then examine subspace optimization techniques for constructing and solving reduced-dimensional optimization models. Next, we review learning-based methods and compliance control for contact-rich robotic assembly, followed by derivative-free approaches to optimizing controller parameters. Together, these research directions provide the theoretical and methodological context for combining particle

swarm optimization with a reduced-subspace trust-region safeguard to tune an interpretable robotic insertion controller.

1.1.1 Derivative-free optimization

BBO addresses problems for which candidate solutions can be evaluated, but an explicit, computationally exploitable expression for the objective or reliable derivative information is unavailable. Such problems arise when evaluations involve numerical simulation, physical experiments, or closed-loop control. DFO develops search strategies that do not require the user to supply derivatives, encompassing direct-search, model-based, stochastic, and population-based techniques. General treatments are given by Audet and Hare [1], Conn et al. [2], and Larson et al. [3]. Comparative studies by Moré and Wild [4] and Rios and Sahinidis [5], together with the methods described in [6, 7], illustrate the diversity of these methods and the importance of evaluation-budget-aware benchmarking.

A further difficulty in BBO is that observed objective values may be inaccurate or uncertain. Numerical discretization and approximate models introduce deterministic errors, whereas random simulation inputs, measurement errors, and stochastic system dynamics can introduce variability. Consequently, an observed decrease in the objective need not imply a comparable improvement in its underlying expected value, especially when evaluations are expensive and few trials are available. For the controller-tuning experiment considered here, a fixed seed-specific batch of scenarios is used to compare candidates consistently within each optimization run; screening and validation on independent scenarios are then used to examine generalization beyond that batch.

Particle swarm optimization (`PSO`) is a population-based DFO metaheuristic introduced by Kennedy and Eberhart [8]. Each particle adjusts its velocity and position using its own best previously observed solution and the best solution identified by the swarm. This mechanism combines individual search experience with population-level information and does not require a local model of the objective. Its comparatively simple updates make it attractive for bounded BBO problems, but the swarm may stagnate or devote a substantial fraction of its evaluation budget to population updates that fail to improve the incumbent. Moreover, the standard `PSO` update does not explicitly construct a local approximation for systematic refinement of the current best solution. These considerations motivate the addition of a conditional local search mechanism, without replacing the underlying particle updates.

Vaz and Vicente [9] introduced a particle swarm pattern search (`PSwarm`), which incorporates particle swarm exploration into a pattern search framework for bound-constrained DFO. When the swarm search fails to improve the incumbent, the method polls coordinate directions around the current best point. An unsuccessful poll contracts the mesh size, allowing progressively finer local exploration. Their pattern search analysis establishes stationarity results under appropriate smoothness and mesh assumptions. This provides an early example of combining population exploration with conditional local refinement.

Trust-region (`TR`) methods instead seek improvement by approximately minimizing a local model within a neighborhood of the current iterate. In DFO, `TR` models can be built from sampled function values rather than analytic gradients; trial steps are typically assessed by comparing actual and predicted reductions [2, 3]. Quadratic `TR` subproblems

can be characterized through optimality conditions and solved using spectral or secular-equation techniques [10]. Reduced subspaces can make these models less costly to store and manipulate in higher dimensions [11]. A model-based trial is not automatically beneficial: the model may be inaccurate, its trial may violate the bounds, and constructing it consumes evaluations. Therefore, a `TR` safeguard within a population method should be activated selectively, and its model probes and trial evaluations must share the overall evaluation budget. Several derivative-free `TR` solvers have been developed, including `UOBYQA` [12], `NEWUOA` [13], `BOBYQA` [14], `BCDFO` [15], and, for nonsmooth objectives, `DFO-TRNS` [16].

Hybrid designs already demonstrate how DFO globalization mechanisms can augment evolutionary search. The matrix adaptation derivative-free optimization (`MADFO`) method [17] combines matrix adaptation with a randomized nonmonotone line search for noisy objectives. Its successor the matrix adaptation trust-region strategy (`MATRS`) [18] integrates mutation, selection, recombination, trust-region steps, and a dedicated mixed-integer search phase for noisy, bound-constrained mixed-integer problems. These methods establish that a population-oriented search can be combined with additional local-search or trust-region components, but their matrix-adaptation updates and intended problem classes differ from those of the present work.

Together, these approaches highlight a recurring trade-off in evaluation-limited DFO. Broad exploration can identify promising regions, while local models use sampled information to guide more focused improvement. The potential benefit of refinement must, however, be weighed against the evaluations, storage, and numerical work required to construct those models. This makes the choice of model dimension and search directions important. Restricting a model to a lower-dimensional space can reduce its computational cost, but also limits the directions it represents. Subspace methods therefore raise a further question: how can a compact representation retain the information most useful for continued improvement?

1.1.2 Subspace optimization methods

Subspace optimization is a widely used strategy for addressing high-dimensional problems by restricting search steps or local models to lower-dimensional spaces. Its main potential benefit is to reduce the cost of constructing and solving the relevant subproblems; a reduction in the number of objective evaluations does not automatically follow.

In large-scale gradient-based optimization, limited-memory and active-set methods exploit information on lower-dimensional spaces of search and free-variable directions. For example, the limited-memory method for bound-constrained optimization (`LMBOPT`) algorithm of Kimiaei et al. [19] combines limited-memory subspace models and search directions with an active-set strategy. Cartis et al. [20] also developed a randomized subspace framework for smooth nonconvex optimization, including subspace Gauss–Newton methods for nonlinear least squares.

For nonlinear least-squares problems in DFO, Kimiaei and Neumaier [21] proposed the limited-memory method for nonlinear least squares (`LMLS`), a limited-memory trust-region method that estimates residual Jacobian information using random coordinate subspaces and incorporates nonmonotone acceptance and adaptive-radius strategies. Cartis and Roberts developed the derivative-free block Gauss–Newton (`DFBGN`) framework, which constructs local interpolation models within random subspaces to improve scalability on

large nonlinear least-squares problems [11].

For general DFO problems, Kimiaei, Neumaier, and Faramarzi [22] introduced the subspace method for unconstrained derivative-free optimization (`SSDFO`), which constructs quadratic models in subspaces spanned by previous search directions and uses the resulting models to obtain line-search directions. More recently, Cartis and Roberts [23] analyzed randomized subspace quadratic models in DFO, including complexity guarantees for approximate second-order stationarity.

Related work by Cartis, Massart, and Otemissov [24, 25] investigates multiple random embeddings for bound-constrained global optimization of functions with low effective dimensionality and a broader random-embedding framework for globally optimizing Lipschitz-continuous objectives.

These studies illustrate different ways of constructing reduced search spaces: exploiting problem structure, sampling random directions, or retaining information from previous iterations. Each choice emphasizes different information and relies on particular assumptions. The usefulness of a reduced model therefore depends not only on its dimension, but also on whether the represented directions allow meaningful improvement in the objective. For objectives defined by robotic task performance, parameter changes act through feedback, motion, and physical interaction. In contact-rich assembly, understanding these effects requires attention to both the controller's behavior and the uncertainty it encounters.

1.1.3 Learning and compliance in assembly

Contact-rich insertion presents a control challenge because successful assembly requires both accurate geometric alignment and appropriate interaction with the environment. Uncertainty in the relative pose of mating parts, friction, and clearance can turn an otherwise feasible insertion motion into rim contact, jamming, or excessive loading. Consequently, position tracking alone may be insufficient: the controller must account for contact forces and determine when to change its motion in response to stalled progress. Classical hybrid position–force control separates position- and force-controlled task directions [26], whereas impedance control addresses the dynamic relationship between interaction forces and motion [27]. Both approaches provide important conceptual foundations for contact-aware manipulation, although the controller studied here does not implement the classical hybrid position–force architecture or online impedance adaptation.

Learning-based methods seek to reduce the manual design and tuning effort associated with contact-rich assembly. Inoue et al. [28] used reinforcement learning (RL) with recurrent deep neural networks for high-precision peg-in-hole assembly and investigated robustness to errors in position and orientation, including experiments on a physical robot. Beltran-Hernandez et al. [29] studied variable-compliance control for peg-in-hole insertion under uncertain hole position, combining off-policy RL with domain randomization and simulation-to-real transfer techniques. These studies illustrate how learning can adjust motion and contact response under geometric uncertainty rather than relying exclusively on a fixed nominal trajectory.

A complementary approach preserves a conventional controller and learns corrections to its output. Johannink et al. [30] investigated residual RL, superimposing a learned action

on a hand-engineered feedback controller and demonstrating the approach on a physical block-assembly task. Zhang et al. [31] developed an offline–online framework in which RL determines motion and compliance-control parameters in randomized simulation, followed by online residual adaptation of admittance parameters using force measurements. Such approaches exploit existing control structure while allowing the response to change under uncertain interaction conditions; they differ from the offline fixed-parameter optimization considered in this paper.

Simulation-based RL has also been developed for multi-stage robotic assembly. `IndustReal` [32] combines simulation-aware policy updates, signed-distance-field rewards, and sampling-based curricula for contact-rich manipulation and assembly; it also introduces a policy-level action integrator to reduce execution errors during deployment. Its reported experiments include transfer of learned policies to a physical robotic assembly system. Force-guided exploration for robust contact-rich manipulation under uncertainty (`FORGE`) [33] addresses pose uncertainty through force-aware policy learning, dynamics randomization, and conditioning on an allowable force threshold. It additionally predicts task success to support termination and force-threshold adjustment and demonstrates physical contact-rich assembly tasks. Together, these studies highlight the importance of evaluating contact-sensitive policies beyond their training simulations.

These studies illustrate that learning can act at different levels of contact-rich control, including task actions, corrections to an existing feedback law, and motion or compliance parameters. The choice of what to learn influences the search space and the information needed to assess improvement. When a useful controller structure is already available, selecting its numerical settings still requires balancing progress, completion, and contact loading. Observed task outcomes provide criteria for comparing candidate settings, connecting contact-aware assembly with policy optimization based on closed-loop performance. This connection motivates examining how controller parameters can be selected when analytical performance derivatives are unavailable.

1.1.4 Derivative-free policy optimization

Derivative-free policy optimization (DFPO) treats the selection of feedback-law, motion-primitive, or staged-controller parameters as a BBO problem, using measured closed-loop performance over complete episodes. In this setting, the objective may depend on contact events, switching rules, termination conditions, or stochastic scenario realizations, and its derivatives with respect to policy parameters may be unavailable or unreliable. This form of direct policy search offers an alternative to learning an unrestricted state-to-action mapping: it optimizes a predefined parameterization using observed rollout costs or rewards. In contrast, theoretical policy-optimization results for structured linear–quadratic control may exploit differentiable policies and more specific modeling assumptions [34]. Such assumptions are not imposed on the insertion-controller objective considered here.

Evolutionary strategies are established tools for episodic policy search. The covariance matrix adaptation evolution strategy (`CMAES`) [35] adapts a multivariate sampling distribution over continuous parameters, whereas the matrix adaptation evolution strategy (`MAES`) [36] updates a sampling transformation directly. In a recent robotic application, Jafari-Tabrizi et al. [37] used `CMAES` and policy-improvement techniques to optimize parameterized robot

scanning trajectories from image-quality feedback. Sago et al. [38] investigated multiobjective path-integral policy improvement for robotic motion. These studies illustrate how rollout-level performance measures can guide policy parameter updates without requiring analytical differentiation through the robot and its environment.

Population heuristics provide another route to controller tuning. Blanck-Kahan et al. [39] employed `DE` to tune controller gains for a parallel robot, while Rodríguez-Molina et al. [40] combined polynomial-regression identification with online `DE`-based tuning for simulated robotic manipulators. Hasan et al. [41] optimized a fuzzy mobile-robot controller using a social-spider search strategy and compared its simulated navigation performance with `PSO`-based alternatives. Huang et al. [42] used `PSO` and beetle-antennae search to tune a neuro-fuzzy active-force controller in robotic trajectory-tracking simulations. These examples motivate population-based parameter optimization, but they do not establish that a particular metaheuristic is uniformly best across different robotic tasks or evaluation budgets.

Data-efficient DFPO has also been investigated within model-based and RL frameworks. Black-box data-efficient robot policy search (`Black-DROPS`) combines DFPO with an uncertain learned dynamics model [43]. Lei et al. [44] integrated DFO-based policy updates into an actor–critic framework for automatic controller tuning, reporting numerical studies of cruise control and trajectory tracking. More recently, He et al. [45] combined robot and simulator observations in a probabilistic model for DFPO to reduce required physical interactions. Metaheuristic optimization has also been combined with value approximation for nonlinear optimal control [46]. These approaches differ in whether simulation data support model learning, policy improvement, or direct optimization of a fixed controller parameter vector.

Surrogate-assisted methods place particular emphasis on expensive evaluations. Chen et al. [47] studied multiobjective Bayesian optimization of robot impedance parameters using transient and steady-state force-tracking objectives, with experimental validation. Nobar et al. [48] used guided Bayesian optimization and a data-driven digital twin for controller tuning. For high-dimensional model-predictive-control parameter selection, Kudva et al. [49] proposed Bayesian optimization with sparse-subspace surrogate models. Zagórowska et al. [50] examined more efficient safe learning for controller tuning and validated their method on a precision motion system. These studies highlight the benefits of learned performance models and explicit validation, although constructing a global performance surrogate or providing an in-experiment safety guarantee is not part of the present algorithm.

Across these approaches, controller tuning involves more than choosing a search algorithm. The policy parameterization determines which behaviors can change, while the objective and evaluation conditions determine what counts as improvement. Evaluation efficiency must therefore be interpreted alongside the quality and scope of the evidence obtained about a controller. Meaningful assessment requires explicit performance criteria, transparent accounting of evaluation effort, and a distinction between improvement on the trials used for parameter selection and performance under additional operating conditions. These considerations provide a basis for formulating tuning problems and assessing their solutions.

1.2 Contributions and relation to existing methods

The preceding review highlights three connected concerns: allocating limited evaluations between exploration and local refinement, selecting the information retained in reduced models, and assessing controller performance beyond the trials used for parameter selection. We investigate these concerns through a conditional integration of `PSO` and reduced-subspace `TR` refinement, together with its application to a predefined insertion-controller family. The numerical assessment examines both benchmark optimization performance and the complete controller-tuning and deployment pipeline.

Figure 1 distinguishes learning paradigms, policy-optimization formulations, and numerical solvers. RL is a learning paradigm, DFPO selects policy parameters from episode evaluations without requiring objective derivatives, and `PSO+TR` is the DFO solver used here. RL can also use derivative-free policy search; the dashed connection indicates this overlap rather than mutually exclusive categories. The central route shows the prescribed controller family, parameter tuning, and independent adoption checks used in this study.

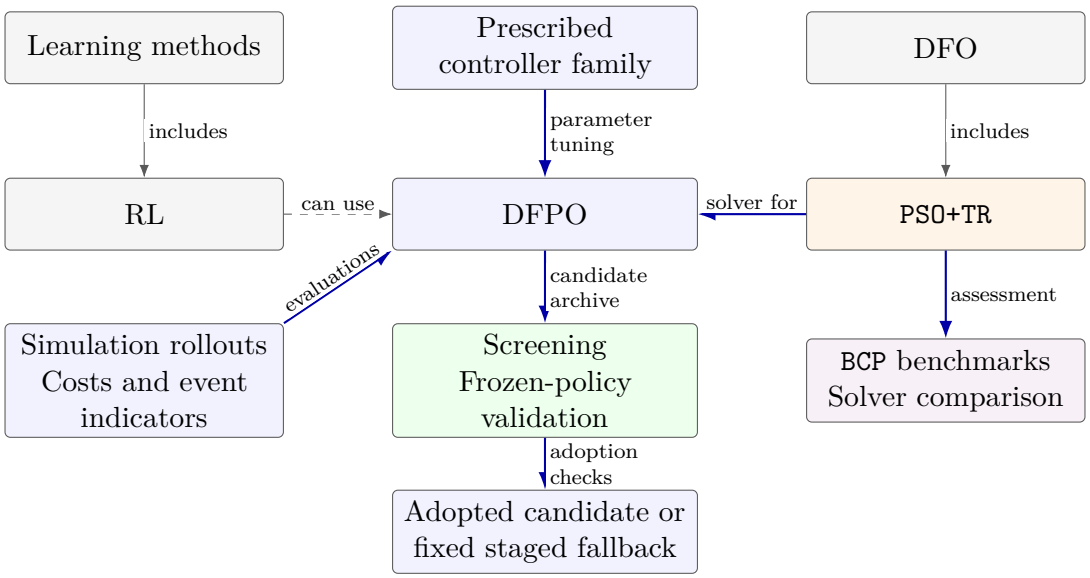

Figure 1: High-level relationships between learning, RL, DFPO, and the `PSO+TR` solver in the present paper.

The principal contributions are summarized as follows:

- **A conditional and successive `TR` safeguard for `PSO`.** The standard swarm-generation and personal/global-best updates are retained. A `TR` invocation is eligible only after a completed generation fails to improve the incumbent and sufficient evaluation allowances remain. Within an invocation, accepted steps may be followed by further reduced-space trials under explicit continuation rules. Initialization, population evaluations, model probes, and trial attempts share one charged evaluation budget.

- **A persistent basis with incremental reduced-curvature updates.** An initially randomized orthonormal basis is retained across iterations and updated from successful population displacements, with random-column refreshes during prolonged non-improvement. Updates following accepted `TR` steps can change column signs but preserve the subspace span in exact arithmetic. A frozen swarm-geometry factor

defines a positive-definite reduced curvature surrogate. Replacing one basis column requires updating only the corresponding factor column and reduced-matrix row and column, without constructing an ambient-dimensional curvature matrix. The surrogate represents search geometry and is not presented as an objective-Hessian estimate.

- **A bounded DFPO formulation for contact-rich insertion.** We tune 14 parameters of a staged approach–search–alignment–insertion controller with history-triggered probing and retreat over $\Theta = [-3, 3]^{14}$. The objective uses the larger expected penalized episode cost under nominal and development scenario distributions. The feedback law, phase logic, and recovery rules are prescribed; optimization selects their numerical parameters. The controller receives measured pose, velocity, and net contact wrench, without access to the true socket pose or contact parameters.

- **Independent controller selection and evaluation.** Candidate screening uses scenarios excluded from optimization. The selected candidate is then frozen before independent validation, and adoption requires predefined screening and validation criteria. Otherwise, the fixed staged reference is retained. Final testing includes a withheld stress distribution that does not enter optimization or adoption decisions. This protocol separates the observations used to choose parameters from those used to assess the selected controller.

- **Complementary numerical assessments.** We compare `PSO+TR` with four DFO solvers on 93 eligible `BCP` problems with dimensions 1–12 from the Branch-And-Reduce Optimization Navigator (`BARON`) collection of Sahinidis [51], using common targets and matched evaluation budgets. Separately, eight independent optimization runs assess the controller-tuning pipeline on a reconstructed synthetic planar insertion model, with paired final-test comparisons against three fixed controllers. These experiments distinguish benchmark solver performance from application-specific controller outcomes.

Relation to optimization methods. Population methods such as `PSO`, `CMAES`, and `MAES` provide established mechanisms for search without objective derivatives [8, 35, 36]. A close precedent is `PSwarm` [9], which combines swarm search with coordinate polling when the search fails to improve the incumbent. Hybrid methods such as `MADFO` and `MATRS` already combine evolutionary search with additional local-search or `TR` mechanisms [17, 18]. Likewise, `SSDFO` and `DFBGN` demonstrate the use of reduced spaces for local optimization models [11, 22]. The present design specifies how a completed-generation trigger, successive `TR` attempts based on reduced quadratic models, persistent basis information, and a frozen geometry factor operate together within a `PSO` search. Its methodological contribution concerns this integration and its evaluation; none of the constituent mechanisms is claimed to be new in isolation.

Relation to contact-rich policy learning and tuning. The use of motion and force feedback has classical foundations in contact-aware control [26, 27], while direct optimization of parameterized robotic behavior is established in controller and trajectory tuning [37, 40]. Learning-based assembly methods span policy learning, residual correction, and compliance adaptation [32, 29, 33, 30]. Here, the controller architecture is specified in advance and its numerical parameters are selected offline. The tuning procedure does not fit a dynamics model or a global performance surrogate, as employed in model-learning and

surrogate-assisted approaches [43, 45, 48]; its quadratic models serve local optimization. The application-level contribution is the joint tuning of search, insertion, and recovery parameters under uncertainty, coupled with independent adoption checks and a specified fallback. The robotic results assess this complete pipeline within the synthetic simulator. A matched PSO-only robotic ablation would be required to isolate the contribution of the TR safeguard.

Figure 2 combines the schematic planar peg–rim geometry with the conditional search operations of PSO+TR. The left panel is illustrative, not a measured trajectory. After a complete non-improving PSO generation, a TR invocation may begin if both the global and per-invocation evaluation allowances permit. Before the first accepted TR step, an unsuccessful attempt may be retried at half the working radius if both allowances and the minimum radius permit. Accepted steps update the incumbent and reduced state and may be followed by further TR attempts. After any accepted step, the first subsequent unsuccessful attempt ends the invocation. The invocation also ends when the evaluation allowances do not permit continuation. Control returns to PSO if the global budget permits. Independent screening, frozen-policy validation, and paired final evaluation are described in Section 5.

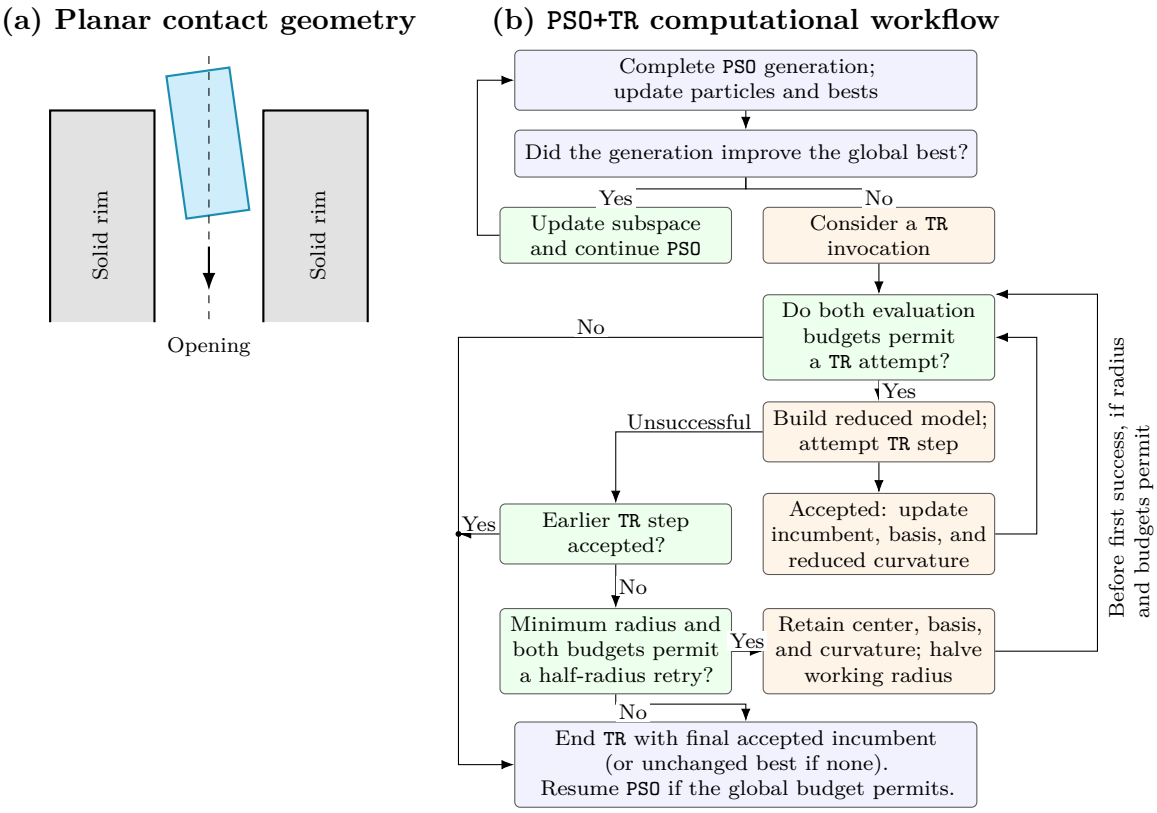

Figure 2: Illustrative planar peg–rim geometry and conditional PSO+TR workflow.

2 Tuning problem

To balance insertion cost and completion reliability under uncertainty, we formulate a DFPO problem over 14 parameters of a structured insertion controller. We minimize the larger of the expected penalized episode costs under the nominal and development scenario distributions, with fixed penalties for unsuccessful completion and benchmark-limit violations. The controller's prescribed phase architecture, feedback structure, and recovery rules remain fixed, while its numerical parameters are tuned using closed-loop simulation outcomes. This yields a finite-dimensional BBO problem, whose sample-based approximation we optimize using the proposed `PSO+TR` algorithm.

The feasible normalized parameter set for the reported experiment is $\Theta = [-3, 3]^{14} \subset \mathbb{R}^{14}$. Let $\theta \in \Theta$ denote the normalized controller parameter vector, and let π_θ represent the corresponding insertion policy. The components of θ determine the motion amplitudes, controller gains, timing parameters, and recovery thresholds.

Under the physical parameter map $p_j = c_j + r_j \tanh(\theta_j)$, $j = 1, \ldots, 14$, the centers c_j and positive radii r_j are specified in Tables 2 and 3, below. Consequently, each mapped parameter satisfies

$$p_j \in [c_j - r_j \tanh(3),\, c_j + r_j \tanh(3)], \qquad j = 1, \ldots, 14,$$

where $\tanh(3) \simeq 0.995054754$. The endpoints of these intervals are attained at $\theta_j = \pm 3$. The values $c_j \pm r_j$ are asymptotic limits of the mapping and are not attained.

For a scenario ω, let $C(\pi_\theta, \omega)$ denote the episode cost, and define $S(\pi_\theta, \omega) \in \{0, 1\}$ and $R(\pi_\theta, \omega) \in \{0, 1\}$ as the completion and benchmark-limit violation indicators, respectively. Completion and violation are recorded as separate outcomes: a successful episode may still involve a benchmark-limit violation.

Let P_0 and P_1 denote the nominal and development scenario distributions, respectively. To account for the different operating conditions, we formulate the bounded tuning problem as

$$\min_{\theta \in \Theta} \max_{j \in \{0,1\}} \mathbb{E}_{\omega \sim P_j} \Big[C(\pi_\theta, \omega) + 10\Big(1 - S(\pi_\theta, \omega)\Big) + 20 R(\pi_\theta, \omega) \Big]. \tag{1}$$

The objective combines episode cost, completion reliability, and benchmark-limit violations. Taking the maximum of the expected penalized costs over the two distributions discourages solutions that perform well under nominal conditions but poorly under the development distribution, or vice versa. Thus, the formulation seeks distribution-level robustness with respect to these two prescribed scenario families. It does not represent worst-case optimization over every possible contact configuration, nor does it guarantee satisfactory performance on distributions that are excluded from the tuning objective.

The coefficients 10 and 20 are fixed penalties associated with unsuccessful completion and benchmark-limit violations, respectively. Their purpose is to discourage solutions that achieve lower episode costs through unreliable insertion or excessive contact loading. These coefficients represent benchmark-specific preferences rather than physical actuator limits or formally calibrated safety requirements.

The penalties in (1) are additional to those already included in the episode cost C, defined in (6). All penalty weights remain fixed throughout the reported experiments; no independent penalty-weight sensitivity study is claimed.

Evaluating the objective requires executing the parameterized controller under uncertain contact conditions. Its dependence on nonlinear contact dynamics, discrete phase transitions, recovery decisions, and event-dependent termination makes analytical differentiation difficult. We therefore treat the objective as a BBO oracle and solve its sample-based approximation using `PSO+TR`. Within each independent optimization run, the same seed-specific scenario batch is used for candidate evaluations, including population updates and trust-region model construction. This permits consistent comparisons of observed objective values, while the resulting sample-average objective remains an approximation of the distribution-level formulation.

Finally, the expectation-based objective and its soft penalties do not impose hard probabilistic safety constraints. Accordingly, optimization is followed by independent candidate screening and frozen-policy validation, using scenarios excluded from the optimization batch. These procedures assess completion, violation, and paired-loss behavior before deciding whether a tuned controller is adopted. Final evaluation on additional nominal, development, and withheld stress scenarios provides a separate assessment of the resulting controller within the specified numerical benchmark.

3 Planar insertion model and controller family

The DFPO formulation in Section 2 requires a closed-loop simulation model for evaluating candidate controller parameters. We therefore consider a synthetic planar insertion benchmark that couples a four-corner compliant contact model with a structured, interpretable controller. This section specifies the dynamics and scenario uncertainty, the information available to the controller, its 14-parameter family, and the episode costs and outcome indicators used by the BBO oracle. The model provides the numerical setting for the subsequent `PSO+TR` experiments; it is not a model of a complete physical robot.

3.1 Dynamics, information, and uncertainty

The planar configuration is $q = (x, z, \varphi)$, where x and z denote the peg-center position and φ its orientation; the positive z direction points downward. The dynamics are

$$\mathrm{diag}(m, m, I)\ddot{q} = u + w_c(q, \dot{q}; \omega). \tag{2}$$

Here u is the applied control wrench and $w_c(q, \dot{q}; \omega)$ is the contact wrench for scenario ω. The mass is $m = 0.12$ kg, inertia $I = 2 \times 10^{-5}$ kg m^2, peg half-width $a = 5$ mm, and half-length $b = 12$ mm. The initial center is $(0, -17 \text{ mm})$ with zero orientation and velocity. These are synthetic benchmark parameters. The model represents local planar motion; articulated joints, gravity, perception, and actuator dynamics are not represented. At each physics step, semi-implicit Euler integration updates velocity using the current control and contact wrenches and then updates position using the new velocity. The contact wrench is

subsequently recomputed. The control command is held constant between 10 ms controller updates.

An episode scenario $\omega = (d_x, \alpha, \mu, g, k_c)$ specifies socket lateral position, orientation, friction, per-side clearance, and contact stiffness. Its components are independently sampled from uniform distributions with ranges in Table 1. The nominal and development distributions are used to construct the tuning objective in (1); the stress distribution is withheld from tuning. The controller receives measured pose, velocity, net contact wrench, and elapsed time, but not the true socket pose, friction coefficient, clearance, or stiffness. These scenario parameters are used internally by the simulator and for outcome assessment. The DFO algorithm receives only the resulting episode evaluations, not the hidden physical parameters. This reconstructed `MATLAB` implementation supplies the BBO oracle for the `PSO+TR` controller-tuning experiments reported in Section 6.

Table 1: Independent uniform scenario distributions. Nominal and development scenarios are used for tuning; stress scenarios are withheld for evaluation.

Parameter	Nominal	Development	Stress
d_x (mm)	$[-1.5, 1.5]$	$[-2.25, 2.25]$	$[-3, 3]$
α (rad)	$[-.04, .04]$	$[-.06, .06]$	$[-.08, .08]$
μ	$[.12, .35]$	$[.35, .55]$	$[.55, .80]$
g (mm)	$[.35, .70]$	$[.20, .35]$	$[.10, .20]$
k_c (N/m)	$[1200, 2000]$	$[1200, 2000]$	$[1800, 3000]$

3.2 Nearest-face contact

The contact model determines the wrench w_c appearing in the planar dynamics. Let

$$Q(\psi) = \begin{bmatrix} \cos\psi & -\sin\psi \\ \sin\psi & \cos\psi \end{bmatrix}, \qquad J = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}.$$

Index the four body-frame corners by $i = 1, \ldots, 4$, with

$$\eta_i \in \{(-a, -b)^\top, (a, -b)^\top, (-a, b)^\top, (a, b)^\top\}.$$

Their coordinates in the true socket frame are

$$\begin{pmatrix} \xi_i \\ \zeta_i \end{pmatrix} = Q(\alpha)^\top \left[\begin{pmatrix} x - d_x \\ z \end{pmatrix} + Q(\varphi)\eta_i \right].$$

For the stationary socket, the socket-frame moment arm measured from the peg center and the corner velocity are

$$r_i = Q(\alpha)^\top Q(\varphi)\eta_i, \qquad v_i = Q(\alpha)^\top \left[\begin{pmatrix} \dot{x} \\ \dot{z} \end{pmatrix} + \dot{\varphi} J Q(\varphi)\eta_i \right].$$

All corner normals, velocities, and forces below are expressed in the socket frame.

Let $p_i = |\xi_i| - a - g$. A corner penetrates a solid rim quadrant when $p_i > 0$ and $\zeta_i > 0$. The scalar penetration depth is

$$d_i = \max\{\min(p_i, \zeta_i), 0\}. \tag{3}$$

For active contact, the contact normal used by the force law is $n_i = (-\operatorname{sgn}\xi_i, 0)^\top$ if $p_i < \zeta_i$, and $(0, -1)^\top$ otherwise.

Writing $n_i = (n_{x,i}, n_{z,i})^\top$, define the tangent $t_i = (-n_{z,i}, n_{x,i})^\top$ and velocity components $v_{n,i} = n_i^\top v_i$, $v_{t,i} = t_i^\top v_i$. Throughout the paper, $[s]_+ = (s)_+ = \max\{s, 0\}$. For active contact, the nonnegative normal-force magnitude N_i and corner force f_i are

$$N_i = [k_c d_i - c_n v_{n,i}]_+, \qquad f_i = N_i n_i - \mu N_i \tanh(v_{t,i}/v_0) t_i, \tag{4}$$

where $c_n = 1.5$ N s/m and $v_0 = 0.001$ m/s. Inactive corner forces are zero.

Writing $r_i = (r_{x,i}, r_{z,i})^\top$ and $f_i = (f_{x,i}, f_{z,i})^\top$, the world-frame contact wrench is

$$w_c = \begin{pmatrix} Q(\alpha) \sum_{i=1}^4 f_i \\ \tau_c \end{pmatrix}, \qquad \tau_c = \sum_{i=1}^4 (r_{x,i} f_{z,i} - r_{z,i} f_{x,i}) .$$

Here, τ_c is the planar contact torque about the peg center. The monitored load $L = \sum_i \|f_i\|_2$ sums the magnitudes of all corner forces and therefore generally differs from the norm of the net contact force supplied to the controller. This distinction matters because the load-based violation threshold uses L, whereas controller transitions use components of the net contact wrench.

The nearest-face rule avoids penalizing a side-contacting corner according to its entire depth below the top plane. The rule remains a four-point compliant approximation, with discontinuous normal changes at the nearest-face switch. It does not resolve complete edge/face contact or three-dimensional geometry.

3.3 Controller structure and parameterization

The controller family π_θ introduced in Section 2 comprises approach, search, alignment, and insertion phases. Its structure and switching rules remain fixed while DFO selects the numerical parameters. Approach enters search when measured axial contact force is below -0.10 N or measured center depth exceeds -12 mm. Search enters alignment at depth -9.5 mm, latching the measured lateral coordinate and angle. Alignment enters insertion after an episode-fixed dwell parameter and depth exceeding -8 mm. The phase transitions use observable signals, without exact socket pose.

During approach and search, the lateral reference is $A \sin(2\pi(t - t_s)_+/P)$ and the angular reference is zero, where t_s denotes the reference-time origin; this origin is reset at the beginning of search. The oscillatory search motion can therefore also act during approach. Alignment holds its entry coordinates. Insertion sets lateral and angular targets to current measured coordinates, retaining damping while releasing their static position errors. The axial targets for the four phases are $\min(-.017 + v_a t, -.010)$, $-.008$, $-.005$, and $.016 + \Delta z$ m.

Each axis uses proportional–derivative (PD) wrench control,

$$u_j = \operatorname{clip}\left(K_j(q_{d,j} - q_j) - 2\sqrt{m_j K_j}\dot{q}_j, -\bar{u}_j, \bar{u}_j\right). \tag{5}$$

Here, $j \in \{1, 2, 3\}$ indexes x, z, φ, respectively; $q_{d,j}$ is the phase-dependent desired coordinate, and K_j is its proportional gain, as specified in Table 2. The effective inertias satisfy $(m_1, m_2, m_3) = (m, m, I)$. The clipping operator is $\mathrm{clip}(a, l, h) = \min\{h, \max\{l, a\}\}$. The wrench caps are $(\bar{u}_1, \bar{u}_2, \bar{u}_3) = (0.7, 0.6, 0.012)$ in N, N, and N m, respectively. Gains and motion parameters remain fixed within an episode, except for the specified phase-dependent switch. The isolated-axis damping formula does not establish stability of the switched contact loop.

The first ten components of θ determine bounded physical parameters through $p_j = c_j + r_j \tanh(\theta_j)$, $j = 1, \ldots, 10$. The row order of Table 2 defines the coordinate order of θ_j and the corresponding physical parameter p_j for these ten components. The four rows of Table 3 correspond, in order, to $j = 11, \ldots, 14$ and use the same mapping. In both tables, each row specifies the center c_j and radius $r_j > 0$ in the listed units; values in mm and mm/s are converted to m and m/s, respectively, for simulation. Setting the first ten components to zero gives the nominal parameter values $p_j = c_j$; disabling the additional probing/recovery guards then recovers the fixed staged reference controller.

Table 2: Staged-policy parameter map.

Parameter	Center c_j	Radius r_j	Unit
Search amplitude A	3.5	2.0	mm
Search period P	4	2	s
Approach speed v_a	10	4	mm/s
Lateral gain before alignment	70	30	N/m
Lateral gain after alignment	12	6	N/m
Axial gain	65	25	N/m
Angular gain before alignment	.025	.0125	N m/rad
Angular gain after alignment	.006	.003	N m/rad
Alignment dwell	.20	.10	s
Final axial offset Δz	1.0	.5	mm

3.4 Objective and outcome definitions

This subsection makes the cost and indicators in (1) explicit. Let T_e be the first recorded completion or protective-termination time, capped at the horizon of 8 s. If neither event occurs, the episode ends at $T_e = 8$ s. The finite-horizon, undiscounted cost is

$$C = \int_0^{T_e} \left(.3 + \frac{(.016 - z)_+}{.026} + .02 \left\| D_u^{-1} u \right\|^2 \right) dt + 12 I_{\text{abort}} + 6 I_{\text{unsuccessful}} + 2 \frac{(.016 - z(T_e))_+}{.026}. \tag{6}$$

The indicator I_{abort} equals one for protective termination and zero otherwise, and $I_{\text{unsuccessful}} = 1 - S$. A timeout without completion is unsuccessful even if no protective abort occurs. Since $I_{\text{unsuccessful}} = 1 - S$, the additional failure penalty in (1) combines with the penalty already included in C to give a total explicit unsuccessful-completion penalty of $16(1 - S)$. The protective-abort penalty $12 I_{\text{abort}}$ and violation penalty $20R$ are added separately; the violation penalty applies regardless of the completion outcome. The matrix $D_u = \mathrm{diag}(3, 3, .03)$ supplies fixed effort reference scales in N, N, and N m, distinct from the command caps. The composite cost is a benchmark objective, not a direct

measurement of physical energy. The progress terms in C use world-frame depth, whereas the completion indicator is evaluated in the true socket frame. The control period is 10 ms.

For a stationary socket centered laterally at d_x and oriented by α, define the socket-relative pose by

$$\begin{pmatrix} x_{\text{rel}} \\ z_{\text{rel}} \end{pmatrix} = Q(\alpha)^\top \begin{pmatrix} x - d_x \\ z \end{pmatrix}, \qquad \varphi_{\text{rel}} = \varphi - \alpha.$$

The corresponding axial speed is

$$\dot{z}_{\text{rel}} = -\sin\alpha\,\dot{x} + \cos\alpha\,\dot{z}.$$

The completion indicator S equals one if completion is recorded and zero otherwise. Completion requires all the following conditions to hold simultaneously for 0.10 s at the physics sampling rate: socket-frame center depth $z_{\text{rel}} \geq 16$ mm, axial speed $|\dot{z}_{\text{rel}}| < 0.02$ m/s, and

$$|x_{\text{rel}}| + a|\cos\varphi_{\text{rel}}| + b|\sin\varphi_{\text{rel}}| \leq a + g + 20 \times 10^{-6} \text{ m}. \tag{7}$$

The full peg must also be below the rim: $z_{\text{rel}} \geq a|\sin\varphi_{\text{rel}}| + b|\cos\varphi_{\text{rel}}|$. The 20 μm allowance accommodates small compliant penetration.

The violation indicator R records any $L > 1.5$ N, $|\tau_c| > 0.12$ N m, or protective workspace termination. Protection terminates at $L > 12$ N, $|\tau_c| > 0.18$ N m, $|x| > 25$ mm, $|z| > 50$ mm, or $|\varphi| > 0.7$ rad. Success and violation are separate labels. State and accumulated cost freeze after termination. Physics-rate checks cannot exclude an unobserved inter-step contact peak. These limits specify the numerical benchmark and are not hardware ratings.

3.5 History-triggered probing and retreat

To recover from stalled insertion, the controller supplements its staged motion with three insertion behaviors: release (mode 0), local probing (mode 1), and retreat (mode 2). The mode numbers identify controller states, not privileged physical contact states; in particular, they are distinct from the peg mass m. At each 10 ms control update, an exponentially weighted axial velocity is updated by

$$\bar{v}_{z,k} = \beta\bar{v}_{z,k-1} + (1-\beta)v_{z,k}, \qquad \beta = 0.935506985. \tag{8}$$

Here, $k = 0, 1, \ldots$ indexes control updates at $t_k = k\Delta t_c$, with $\Delta t_c = 0.01$ s. The quantity $v_{z,k} = \dot{z}(t_k)$ is the measured world-frame axial velocity, and $\bar{v}_{z,-1} = 0$ initializes the filter at the start of each episode. The decay time is $-\Delta t_c/\log\beta \simeq 0.15$ s.

During insertion, a stall age accumulates when $\bar{v}_z < v_{\text{stall}}$, measured axial wrench is below -0.10 N, and world center depth is less than 17 mm; otherwise it resets. Release enters probing when stall age $\geq t_{\text{probe}}$. Upon entering probing, the controller records the measured lateral position x_{entry} and entry time t_{entry}. The probe target is

$$x_d = x_{\text{entry}} + A_{\text{probe}}\sin(2\pi(t - t_{\text{entry}})/P_{\text{probe}}), \qquad \varphi_d = 0, \tag{9}$$

where $P_{\text{probe}} = 0.6$ s is the fixed probe period. The existing post-alignment gains are retained. During probing, release resumes when $\bar{v}_z \geq 2v_{\text{stall}}$, and stall age resets. A persistent stall

triggers retreat when stall age $\geq t_{\text{retreat}}$, provided fewer than three retreats have been completed. The target depth is latched 1.5 mm above the current measured depth. The retreat-age accumulator is reset to zero on entry and incremented by $\Delta t_c = 0.01$ s at every retreat control update, including the entry update. Search restarts when retreat age ≥ 0.25 s. The search-to-alignment depth guard is re-evaluated at the next 10 ms control update. If the measured world-frame center depth satisfies $z > -9.5$ mm at that update, the controller enters alignment. At most three retreats are allowed; later persistent stalls remain in probing. Command caps are unchanged. The guards react to observed wrench and motion and do not receive friction, clearance, or true socket pose.

Four additional components of θ parameterize the stall, probe, and retreat responses using the same bounded mapping $p_j = c_j + r_j \tanh(\theta_j)$ (Table 3), giving 14 tuning parameters in all.

Table 3: Additional hybrid-policy parameters. Total search dimension is 14.

Parameter	Center	Radius	Unit
v_{stall}	1.0	.9	mm/s
t_{probe}	.30	.15	s
A_{probe}	1.5	1.0	mm
t_{retreat}	.90	.40	s

For comparison, the fixed staged baseline disables probing and retreat and uses zero values for the first ten normalized parameters. The fixed-hybrid comparator enables these behaviors with all normalized parameters set to zero. Both are predefined reference controllers, not optimal recovery policies. Hence, the DFPO task is to choose the parameters of the structured hybrid controller, not its phase architecture or the contact geometry. The peg retains the noncircular, rectangular geometry specified above.

4 PSO+TR with a reduced-subspace TR safeguard

To supplement population exploration with local refinement, PSO+TR combines PSO with a conditional reduced-subspace TR procedure. A TR invocation is eligible only after a complete PSO generation fails to improve the incumbent and the remaining evaluation allowances permit an attempt.

The working TR radius bounds the trial displacement and the region in which the local model is used. Before the first accepted step, an unsuccessful attempt may be retried at half the working radius if the minimum radius and evaluation allowances permit. Trial acceptance requires a strict objective decrease and an actual-to-predicted reduction ratio meeting the prescribed threshold. Accepted steps may be followed by further local refinement; the first subsequent unsuccessful attempt ends the invocation, as does exhaustion of either evaluation allowance.

The general formulation is the bounded DFO problem

$$\min_{x \in [\ell, u]} f(x), \qquad x \in \mathbb{R}^n, \tag{10}$$

where f is evaluated through an oracle without using its derivatives, and $\ell, u \in \mathbb{R}^n$ are componentwise bounds.

In the robotic application, x represents the normalized controller parameter vector θ from (1), with $n = 14$ and feasible set $\Theta = [-3, 3]^{14}$. Accordingly, the bounds are $\ell = -3\mathbf{1}_{14}$ and $u = 3\mathbf{1}_{14}$, where $\mathbf{1}_{14} \in \mathbb{R}^{14}$ is the all-ones column vector. The same solver is evaluated on eligible `BCP` problems through the five-method comparison in Section 7, separately from its application to robotic controller tuning.

A complete `PSO` generation precedes each possible `TR` intervention. If the population improves its global best, the safeguard is skipped. Otherwise, an eligible intervention can solve multiple reduced `TR` subproblems. After an accepted step, the next model is constructed around the newly accepted point and the process continues while its evaluation allowance permits. A failure following at least one success ends the intervention; unsuccessful trials before the first success may instead be retried at smaller working radii. This successive-step mechanism is distinct from repeatedly invoking `TR` after every particle update.

For DFPO, the distribution-level objective in (1) is replaced during each optimization run by a sample-based BBO oracle. Feasible population, `TR` probe, and `TR` trial points are evaluated on the same fixed, seed-specific scenario batch. Thus, their objective values can be compared consistently within a run, although observed sample-average improvements need not generalize to new scenarios. Independent screening and validation are described in Section 5.

4.1 `PSO` population dynamics and conditional intervention

Let N be the swarm size. At generation $g = 0, 1, \ldots$, particle $i \in \{1, \ldots, N\}$ has position $x_{i,g} \in \mathbb{R}^n$, velocity $v_{i,g} \in \mathbb{R}^n$, and personal best $p_{i,g} \in \mathbb{R}^n$; the current swarm best is $g_g \in \mathbb{R}^n$. The symbol $\odot$ denotes componentwise multiplication, and $\mathbf{1}_n \in \mathbb{R}^n$ is the all-ones vector.

For a finite box $[\ell, u]$, draw mutually independent vectors $r_i^x, r_i^v \sim \mathcal{U}([0, 1]^n)$ and initialize

$$x_{i,0} = \ell + (u - \ell) \odot r_i^x, \qquad v_{i,0} = 0.1(u - \ell) \odot (2r_i^v - \mathbf{1}_n), \qquad i = 1, \ldots, N.$$

Replace $x_{1,0}$ by the prescribed initial candidate $x^0 \in [\ell, u]$, evaluate all N initial particles, and set

$$p_{i,0} = x_{i,0}, \qquad i = 1, \ldots, N, \qquad g_0 \in \operatorname*{argmin}_{x \in \{p_{i,0} : 1 \le i \le N\}} f(x).$$

All initialization evaluations are charged to the global evaluation budget. The robotic runs use $N = 10$ and $x^0 = \mathbf{0}_{14}$, the zero normalized parameter vector.

With independent vectors $r_{1,i,g}, r_{2,i,g}$ uniformly distributed on $[0, 1]^n$, the `PSO` update [8] is

$$v_{i,g+1} = \omega v_{i,g} + c_1 r_{1,i,g} \odot (p_{i,g} - x_{i,g}) + c_2 r_{2,i,g} \odot (g_g - x_{i,g}), \tag{11}$$

$$x_{i,g+1} = \Pi_{[\ell,u]}(x_{i,g} + v_{i,g+1}), \tag{12}$$

where $\Pi_{[\ell,u]}$ is the componentwise box projection. The inertia weight is ω, while c_1 and c_2 are the cognitive and social acceleration coefficients, respectively. The executed configuration uses $(\omega, c_1, c_2) = (0.7, 1.4, 1.4)$; this scalar inertia weight is distinct from the scenario

vector ω used in the robotic model. When projection changes a particle coordinate, the corresponding velocity component is set to zero. These initialization, update, and boundary-handling rules concern the proposed solver; the separate `PSO` comparator retains its own implementation.

Each updated particle is evaluated before the personal and global bests are recomputed:

$$\begin{aligned} p_{i,g+1} &= \begin{cases} x_{i,g+1}, & f(x_{i,g+1}) < f(p_{i,g}), \\ p_{i,g}, & \text{otherwise}, \end{cases} \\ g_{g+1}^{\mathrm{PSO}} &\in \operatorname*{argmin}_{x \in \{p_{i,g+1} : 1 \le i \le N\}} f(x). \end{aligned} \tag{13}$$

For robotic tuning, let $\Omega_j^{(s)}$ be the fixed scenario batch for distribution P_j and optimization seed s. The oracle is

$$\widehat{f}^{(s)}(\theta) = \max_{j \in \{0,1\}} \frac{1}{|\Omega_j^{(s)}|} \sum_{\xi \in \Omega_j^{(s)}} \Big[C(\pi_\theta, \xi) + 10\Big(1 - S(\pi_\theta, \xi)\Big) + 20 R(\pi_\theta, \xi) \Big]. \tag{14}$$

The robotic runs use $|\Omega_0^{(s)}| = |\Omega_1^{(s)}| = 8$; the training batches differ across seeds but remain unchanged between evaluations within each run.

Let $f_g^{\mathrm{old}} = f(g_g)$ be the incumbent value immediately before generation $g+1$. The nonnegative $\varepsilon_{\mathrm{imp}}$ is an absolute improvement tolerance in objective units; it is zero in all reported runs. The reduced dimension used below is $m = \min\{n, 10\}$. A completed generation improves the incumbent when

$$f(g_{g+1}^{\mathrm{PSO}}) < f_g^{\mathrm{old}} - \varepsilon_{\mathrm{imp}}. \tag{15}$$

In that case, the best point is retained, the successful displacement updates the persistent subspace, and no `TR` intervention occurs. Otherwise, `TR` may be invoked only after every particle in that generation has been evaluated and at least $m+1$ objective calls remain in both the global budget and the intervention allowance. There is at most one **invocation** per eligible generation, but that invocation may contain several reduced `TR` subproblems.

4.2 Initialization and success-driven update of the search subspace

The reduced models are motivated by subspace DFO methods [11, 23]. Our implementation retains an orthonormal basis across iterations and modifies it using successful displacement directions rather than independently resampling an entire subspace at each call. Set $m = \min\{n, 10\}$ and, once per run, generate

$$G_0 = (\gamma_{ij}) \in \mathbb{R}^{n \times m}, \qquad \gamma_{ij} \overset{\mathrm{iid}}{\sim} \mathcal{N}(0,1), \qquad G_0 = S_0 R_0. \tag{16}$$

Here $R_0 \in \mathbb{R}^{m \times m}$ is the upper-triangular factor of the thin QR factorization, and $i = 1, \ldots, n$ and $j = 1, \ldots, m$ index the rows and columns of G_0, respectively. Throughout this section, I_d is the $d \times d$ identity, $\|\cdot\|_2$ is the Euclidean vector norm, and $\|\cdot\|_F$ is the Frobenius matrix

norm. Two-pass Gram–Schmidt orthogonalization produces $S_0 = [s_{1,0}, \ldots, s_{m,0}] \in \mathbb{R}^{n\times m}$ with $S_0^\top S_0 = I_m$, up to floating-point error. A separate random stream prevents subspace construction from consuming the random draws used by `PSO`.

At a successful population or `TR` step, let x^- and x^+ be the incumbent before and after acceptance, and let S be the current basis. Define the displacement and the column selected for replacement by

$$d = x^+ - x^-, \qquad j = \min\left(\operatorname*{argmax}_{1\le i\le m} |s_i^\top d|\right). \tag{17}$$

Thus, ties in the maximum-correlation selection are resolved by the first maximizing index. The replacement is skipped, leaving S unchanged, if

$$\|d\|_2 \le 10^{-12}\max\{1, \|x^-\|_2\}.$$

Otherwise, normalize $u = d/\|d\|_2$ and let S_{-j} contain all columns except s_j. Compute the candidate residual $\widetilde{s}_j$ by reorthogonalizing u against the columns of S_{-j} in two passes. The candidate is retained only if $\|\widetilde{s}_j\|_2 > 10^{-10}$; otherwise, S remains unchanged. For a retained candidate, the exact-arithmetic formulas are

$$\widetilde{s}_j = (I_n - S_{-j}S_{-j}^\top)u, \qquad s_j^+ = \frac{\widetilde{s}_j}{\|\widetilde{s}_j\|_2}. \tag{18}$$

All other columns are preserved, and in exact arithmetic,

$$s_i^+ = s_i \quad (i \ne j), \qquad (S^+)^\top S^+ = I_m. \tag{19}$$

The computed basis satisfies the orthonormality condition up to floating-point error.

A performed replacement preserves $m-1$ previous basis columns and ensures that d lies in the span of S^+ in exact arithmetic. Population displacements with a nonzero component outside the current subspace can therefore change its span. For an accepted `TR` displacement $d = Sz$, with $z \in \mathbb{R}^m$, however,

$$(I_n - S_{-j}S_{-j}^\top)d = z_j s_j.$$

Since the selected coefficient z_j is nonzero for $d \ne 0$, a performed replacement reduces to $s_j^+ = \operatorname{sgn}(z_j)s_j$ in exact arithmetic. Thus, success-driven updates after accepted `TR` steps can change the sign of the selected column but do not introduce a new subspace direction. Such updates can occur several times within one successive-`TR` invocation, immediately after its accepted steps. The sign changes can reverse subsequent forward-probe directions even though the subspace is unchanged.

If the global best does not improve for five consecutive completed generations, the last basis column is replaced by a newly sampled Gaussian direction orthogonalized against the others. This refresh requires no objective evaluation and is separate from the success-driven update. When $m = n$, the basis spans $\mathbb{R}^n$, and both the success-driven updates and these one-column random refreshes reduce to possible column-sign changes in exact arithmetic. Neither the random initialization nor the basis-update rule alone provides a convergence guarantee.

4.3 Direct reduced-curvature construction and incremental updating

We construct an $m \times m$ curvature surrogate directly in the current search subspace, without forming an ambient $n \times n$ curvature matrix. At the first eligible `TR` invocation, take a snapshot of the current swarm positions X, velocities V, and personal bests P, each in $\mathbb{R}^{N\times n}$, and form

$$D = \begin{bmatrix} X - \mathbf{1}g^\top \\ V \\ P - X \end{bmatrix} \in \mathbb{R}^{3N\times n}. \tag{20}$$

Here, $\mathbf{1} \in \mathbb{R}^N$ is the all-ones column vector, and the rows of X, V, P are $x_i^\top, v_i^\top, p_i^\top$ from the snapshot, respectively. The vector $g \in \mathbb{R}^n$ is the swarm best at that snapshot, distinct from the generation index. Put $s = \max_{i,j} |D_{ij}|$ and $\widetilde{D} = D/s$ when $s > 10^{-15}$; otherwise take $\widetilde{D} = 0$. Let $\|\cdot\|_F$ denote the Frobenius matrix norm. For $\widetilde{D} \in \mathbb{R}^{3N\times n}$,

$$\|\widetilde{D}\|_F = \left(\sum_{i=1}^{3N}\sum_{j=1}^{n} |\widetilde{D}_{ij}|^2\right)^{1/2}.$$

The stored geometry factor and regularization weight are

$$(F, \alpha) = \begin{cases} \left(\sqrt{0.95n/\|\widetilde{D}\|_F^2}\,\widetilde{D},\ 0.05\right), & \widetilde{D} \neq 0, \\ \left(\mathbf{0}_{0\times n},\ 1\right), & \widetilde{D} = 0. \end{cases} \tag{21}$$

Here, $\mathbf{0}_{0\times n}$ denotes the empty matrix with zero rows and n columns. The scalar α in this subsection is a regularization weight, distinct from the socket angle used in the contact model. This normalization preserves direction information in the frozen swarm snapshot, while the second case avoids a degenerate normalization.

The operator $C_{\text{geo}} = F^\top F + \alpha I_n$ is useful for interpreting the construction, but is never formed in the implementation. Instead, store

$$T = FS, \qquad B = T^\top T + \alpha I_m. \tag{22}$$

Because $\alpha > 0$, the reduced matrix is positive definite in exact arithmetic. It is a regularized swarm-geometry surrogate, not an objective-Hessian estimate. The factor F and weight α are frozen after their initial construction; subsequent changes to B result only from changes to S.

In this subsection, $t_i = T(:, i)$ denotes the ith factor column, distinct from the contact tangent used in the planar contact model. For a nonempty factor, $T \in \mathbb{R}^{3N\times m}$ and $t_i \in \mathbb{R}^{3N}$. In the empty-factor case, $T \in \mathbb{R}^{0\times m}$; each column has length zero, its inner products are zero, and $B = I_m$. The Kronecker delta satisfies $\delta_{ij} = 1$ for $i = j$ and zero otherwise.

When column j of S is replaced, update only the corresponding column of T:

$$t_j^+ = Fs_j^+, \qquad t_i^+ = t_i \quad (i \neq j). \tag{23}$$

The affected row and column of the reduced curvature matrix then satisfy

$$(B^+)_{ij} = (t_i^+)^\top t_j^+ + \alpha\delta_{ij}, \qquad (B^+)_{ji} = (B^+)_{ij}, \quad 1 \le i \le m. \tag{24}$$

The $(m-1)\times(m-1)$ submatrix indexed by $i,\ell \neq j$ remains unchanged. The same update applies to a random-column refresh. This exact one-row/one-column modification relies on the frozen factor and regularization weight: updating the swarm snapshot itself would generally change every entry of B.

The matrices F, S, T, and B require $O(Nn+nm+m^2)$ storage, apart from constant factors. Replacing a basis column and updating the reduced curvature requires $O(nm+Nn+Nm)$ arithmetic. No ambient $n\times n$ curvature matrix is formed. In exact arithmetic, $S^\top S = I_m$ gives $B = S^\top C_{\text{geo}} S$; for the computed basis, this relation holds up to floating-point error. It does not assert that the frozen geometry surrogate tracks later swarm configurations.

4.4 Derivative-free quadratic models and successive TR steps

Consider an eligible TR invocation after a non-improving PSO generation. Index its reduced subproblems by $k = 0, 1, \ldots$. Let $x_k \in \mathbb{R}^n$ be the current accepted center, and let $S_k \in \mathbb{R}^{n\times m}$ and $B_k \in \mathbb{R}^{m\times m}$ be the current persistent basis and reduced curvature matrix. Initially $x_0 = g_{g+1}^{\text{PSO}}$ and the working radius is $\Delta_0 > 0$. The vector $z \in \mathbb{R}^m$ contains reduced step coordinates, so the full-space displacement is $S_k z$; this vector is distinct from the physical depth in the planar insertion model. For $j = 1, \ldots, m$, $e_j \in \mathbb{R}^m$ is the jth canonical coordinate vector. The positive Δ_k is the working TR radius, and h_k is the forward-probe length defined below.

At inner step k, define the reduced quadratic model

$$q_k(z) = f(x_k) + c_k^\top z + \tfrac{1}{2} z^\top B_k z, \qquad \|z\|_2 \le \Delta_k. \tag{25}$$

Only $c_k \in \mathbb{R}^m$ is fitted from fresh function values; B_k is the reduced surrogate described above. For a minimum radius $\Delta_{\min} > 0$, choose

$$h_k = \min\{\Delta_k, \max\{\Delta_k/2, \Delta_{\min}\}\}.$$

The forward probes and curvature-corrected differences are

$$f_{k,j} = f(x_k + h_k S_k e_j), \qquad y_{k,j} = f_{k,j} - f(x_k) - \tfrac{1}{2} h_k^2 (B_k)_{jj}, \quad j = 1, \ldots, m. \tag{26}$$

These m probes use the current basis and the current accepted center. Any basis-column update following an accepted TR step is therefore reflected in the next probes and fitted model.

With $A_k = h_k I_m$, $y_k = (y_{k,1}, \ldots, y_{k,m})^\top$, and a nonnegative ridge weight λ_{ridge}, the fitting problem is

$$c_k = \operatorname*{arg\,min}_{c\in\mathbb{R}^m} \left\{ \|A_k c - y_k\|_2^2 + \lambda_{\text{ridge}} h_k^2 \|c\|_2^2 \right\}. \tag{27}$$

Its solution is explicit (with $c_k = \nabla_z q_k(0)$ denoting the model gradient in reduced coordinates):

$$(c_k)_j = \frac{f(x_k + h_k S_k e_j) - f(x_k) - \frac{1}{2} h_k^2 (B_k)_{jj}}{h_k(1+\lambda_{\text{ridge}})}. \tag{28}$$

Each inner subproblem therefore requires at most m fresh forward probes and, when a valid descent model is obtained, one trial evaluation. Invalid or out-of-box probes are charged

to the evaluation budget and invalidate that particular model. The optimization is over a Euclidean ball in reduced coordinates, not the exact intersection with the original box; the candidate trial is checked by the objective oracle, which assigns an invalid result to an infeasible point. No clipping of a TR trial back into the box is assumed.

The reduced quadratic subproblem is

$$z_k^* \in \operatorname*{arg\,min}_{\|z\|_2 \le \Delta_k} \left(c_k^\top z + \tfrac{1}{2} z^\top B_k z \right). \tag{29}$$

Since $B_k \succ 0$ in exact arithmetic, this is a strictly convex problem with a unique minimizer. The TR optimality conditions [10] state that a multiplier $\lambda_k \ge 0$ exists such that

$$\begin{aligned} &(B_k + \lambda_k I_m) z_k^* = -c_k, \\ &\|z_k^*\|_2 \le \Delta_k, \qquad \lambda_k(\|z_k^*\|_2 - \Delta_k) = 0. \end{aligned} \tag{30}$$

These conditions are necessary and sufficient for the reduced quadratic problem. Writing

$$B_k = U_k \operatorname{diag}(\beta_{k,1}, \dots, \beta_{k,m}) U_k^\top, \qquad a_k = U_k^\top c_k, \tag{31}$$

with $U_k^\top U_k = I_m$ and $\beta_{k,i} > 0$, the unconstrained minimizer $-B_k^{-1} c_k$ is used if it lies inside the radius. Otherwise $\lambda_k > 0$ and

$$z_k^*(\lambda_k) = -U_k \operatorname{diag}\left(\frac{1}{\beta_{k,1} + \lambda_k}, \dots, \frac{1}{\beta_{k,m} + \lambda_k} \right) a_k. \tag{32}$$

The multiplier is the unique positive root of the secular equation

$$\Phi_k(\lambda) = \sum_{i=1}^{m} \frac{(a_k)_i^2}{(\beta_{k,i} + \lambda)^2} - \Delta_k^2 = 0, \qquad \lambda > 0. \tag{33}$$

The implementation brackets and bisects this root and applies a final radius projection when needed. Its numerical kernel also handles singular or indefinite input defensively, although these cases are not expected for the regularized matrix in (22). Solving the reduced quadratic problem does not imply global optimality for the original BBO objective.

For a candidate $x_k^{\mathrm{tr}} = x_k + S_k z_k^*$, the predicted decrease is

$$\mathrm{pred}_k = -c_k^\top z_k^* - \tfrac{1}{2} (z_k^*)^\top B_k z_k^*. \tag{34}$$

The trial is evaluated only when the predicted decrease is finite and positive. The actual-to-predicted reduction ratio is

$$\rho_k = \frac{f(x_k) - f(x_k^{\mathrm{tr}})}{\mathrm{pred}_k}. \tag{35}$$

The TR ratio ρ_k measures agreement between the actual objective reduction and the reduction predicted by the model. The step is accepted if its objective value and ρ_k are finite and

$$f(x_k^{\mathrm{tr}}) < f(x_k) - \varepsilon_{\mathrm{imp}}, \qquad \rho_k \ge \eta_{\mathrm{accept}}, \tag{36}$$

where $\varepsilon_{\mathrm{imp}} \ge 0$ is the absolute improvement tolerance and η_{accept} is the acceptance threshold.

Upon acceptance, set $x_{k+1} = x_k^{\mathrm{tr}}$, retain $\Delta_{k+1} = \Delta_k$, and apply the success-driven basis-column update using $x_{k+1} - x_k$, as described in Section 4.2. If a column changes, update the corresponding row and column of the reduced matrix before fitting the next quadratic model. Accepted steps can therefore be followed by further `TR` subproblems within the same invocation.

If a subproblem is unsuccessful **before any `TR` success** in the current invocation, the incumbent and reduced state are retained:

$$x_{k+1} = x_k, \qquad S_{k+1} = S_k, \qquad B_{k+1} = B_k.$$

If the minimum radius and the remaining per-invocation and global evaluation allowances permit another attempt, set $\Delta_{k+1} = \Delta_k/2$ and continue; otherwise, the invocation terminates. Once at least one `TR` step has succeeded, the **first subsequent unsuccessful subproblem** terminates the invocation without accepting its trial. Exhaustion of the per-invocation or global evaluation allowance also terminates the inner sequence. The configured initial radius Δ_0 is not modified by these internal reductions.

If the invocation contains accepted steps, its final accepted point is installed as the `PSO` global best. The particle with the worst personal-best objective value has both its position and personal best replaced by this point, and its velocity is reset to zero. If there is no accepted `TR` step, the pre-invocation global best is retained. The next major iteration resumes the usual `PSO` population updates while the global evaluation budget permits; there is no blending between swarm and `TR` points.

The experimental configurations and evaluation allowances are reported in Section 7 and Table 8.

5 `PSO+TR` robotic experimental design, controller selection, and validation

To test whether sampled training gains transfer to new scenarios, we separate optimization, candidate screening, frozen-policy validation, and withheld final testing. The proposed `PSO+TR` solver optimizes a prescribed 14-parameter controller, while a separate screen–freeze–validate protocol determines whether a tuned policy should replace the fixed staged reference. This separation is essential: an improvement in the sampled training objective does not, by itself, establish reliable performance on independent scenarios. The complete procedure comprises optimization, candidate screening, frozen-policy validation, and withheld final tests.

5.1 Reconstructed simulator and optimization protocol

The BBO oracle is implemented in `MATLAB` using the four-corner, nearest-face contact model and the controller defined in Section 3. The simulator uses semi-implicit Euler integration with a 0.250 ms physics step for training and a 0.125 ms step for screening, validation, and final testing. Control commands are updated every 10 ms, and each episode has a maximum

duration of 8 s. This is a synthetic numerical implementation of the stated planar model; it is not a hardware experiment or a verified reproduction of an independently implemented simulator.

Eight independent optimizer runs use seeds $0, \ldots, 7$, ten swarm particles, a 14-dimensional normalized parameter vector, and a budget of 120 charged objective-evaluation attempts per seed. Each feasible evaluation executes eight nominal and eight development scenarios, which are drawn once for a given seed and held fixed throughout that optimizer run. The training oracle is the maximum of the two sample means of the penalized episode cost in (1).

The charged training workload is $8 \times 120 \times 16 = 15{,}360$ rollout equivalents, or 1,920 per seed. An infeasible attempted evaluation is charged by the optimizer without executing its scenario batch. The saved optimizer logs record 109 such infeasible attempts across the eight runs, leaving $960 - 109 = 851$ feasible evaluations and $851 \times 16 = 13{,}616$ executed training episodes. Thus, charged rollout equivalents and executed simulator episodes are reported separately.

Initialization evaluations, subsequent particle evaluations, `TR` model probes, and `TR` trial evaluations all count against the same 120-attempt budget, including infeasible attempts. The successive-`TR` mechanism described in Section 4 may solve more than one reduced subproblem within a conditional `TR` invocation; it receives no additional unaccounted objective-evaluation allowance.

The `PSO+TR` hyperparameters are transferred from the separate `BCP` tuning archive without selecting or readjusting them using the robotic final-test results. The controller parameters themselves are optimized independently for each seed. The `BCP` solver comparison is reported separately in Section 7; it is not a matched-simulator robotic comparison among optimizers.

5.2 Candidate screening and policy freezing

Following optimization within each seed, the returned incumbent is placed first in the candidate list, followed by archived population points with finite training scores in increasing order of $\widehat{f}^{(s)}$ from (14). The archive contains finite evaluations of the initial and subsequent swarm positions, recorded in evaluation order. Equal training scores preserve this archive order. `TR` probes and trials are not appended to the population archive; the returned incumbent is included separately even if it results from an accepted `TR` step. Exact duplicate parameter vectors are removed from the combined list while preserving their first occurrence. The first 13 distinct vectors are screened, or all available vectors if fewer remain.

Within each seed, every collected candidate and the fixed staged reference are evaluated on the same newly sampled 64 nominal and 64 development scenarios. These scenarios are independent of the optimizer's fixed training batch.

For a candidate θ, let $\widehat{v}_j(\theta)$ be its empirical violation fraction on screening distribution $j \in \{0, 1\}$, and let $\widehat{\ell}_j(\theta)$ be the fraction of paired screening scenarios on which the fixed staged controller succeeds but π_θ fails. A candidate is eligible only if

$$\widehat{v}_j(\theta) \leq 0.05, \qquad \widehat{\ell}_j(\theta) \leq 0.03, \qquad j \in \{0, 1\}. \tag{37}$$

Among eligible candidates, the method selects the one with the lowest balanced empirical mean episode cost, $\frac{1}{2}(\overline{C}_0(\theta) + \overline{C}_1(\theta))$, where

$$\overline{C}_j(\theta) = \frac{1}{64} \sum_{r=1}^{64} C(\pi_\theta, \omega_{j,r}^{\mathrm{scr}}), \qquad j \in \{0, 1\}.$$

Here $\omega_{j,r}^{\mathrm{scr}}$ is the rth screening scenario from distribution P_j within the current seed. If eligible candidates have exactly equal balanced costs, the first one in the candidate-list order is retained; no tie tolerance is used.

This selection criterion excludes the additional penalties $10(1-S)+20R$ used in the training oracle. The episode cost C retains its own unsuccessful-completion and protective-abort penalties. The screening inequalities separately restrict the observed violation and paired-loss frequencies. The selected candidate is then frozen; no validation observation is used to change its parameters or to reselect a different candidate. If no candidate is eligible, the fixed staged reference is retained without candidate validation.

5.3 Independent frozen-policy validation

The frozen candidate and the fixed staged reference are evaluated on a new, common batch of 512 nominal and 512 development scenarios. For k occurrences of a binary event in n independent scenarios drawn from a given validation distribution, we use the one-sided exact Clopper–Pearson upper confidence bound

$$U(k, n, a_\star) = \begin{cases} \mathrm{Beta}^{-1}(1 - a_\star; k+1, n-k), & k < n, \\ 1, & k = n. \end{cases} \tag{38}$$

Here, $\mathrm{Beta}^{-1}(q; a, b)$ is the q quantile of the beta distribution with shape parameters $a, b > 0$. The fixed per-bound tail probability is $a_\star = 0.05/256$. Each upper bound therefore has nominal confidence level $1 - a_\star$, with coverage at least this level under the binomial model. In validation, $n = 512$ and k counts occurrences of the event being bounded within the relevant distribution. The denominator 256 is a prescribed allocation factor; it is not presented as the number of simultaneous tests in the eight-seed experiment.

For **each** validation distribution, adoption requires the bound for the paired event "fixed staged succeeds, candidate fails" to be at most 0.03, the candidate's violation bound to be at most 0.05, and the fixed staged reference's violation bound to be at most 0.05. In addition, the candidate's balanced mean validation cost must be strictly below that of the reference. All these conditions must hold for the **same frozen candidate**. If any condition fails, or no candidate passes screening, the deployed policy is the fixed staged reference. Rejected candidates are never replaced by the zero-parameter hybrid controller, because enabling its probing and retreat rules does not reproduce the fixed staged baseline.

The exact binomial bound quantifies uncertainty about the specified binary event probabilities under independent draws from the validation distributions. It does not give a guarantee for the withheld stress distribution, other contact models, or physical hardware; the empirical cost-improvement condition is not itself a confidence-bound guarantee.

5.4 Final tests, evaluation accounting, and provenance

After deployment is determined, four controller configurations are evaluated: the deployed `PSO+TR` selection pipeline (the adopted candidate or its fixed staged fallback), fixed staged, continued search, and fixed hybrid. The continued-search reference uses the nominal staged parameters and disables probing and retreat, but continues the lateral sinusoid during insertion instead of setting its lateral target to the measured current position; its insertion angular target remains the measured angle. Each configuration is evaluated on the **same** 400 fresh scenarios per seed for each of the nominal, development, and withheld stress distributions. Scenarios are paired across controllers within a seed, and fresh draws are used for different seeds. The resulting final-test ledger contains $8 \times 4 \times 3 \times 400 = 38{,}400$ controller episodes, not 38,400 independent optimizer runs.

The archive includes the seed-specific optimizer checkpoints, function-evaluation histories, screening and validation decisions, and episode-level final-test outcomes. The supplied results also contain an auditable correction of an earlier deployment error. Seed 4 produced no eligible screening candidate, so the prescribed protocol required the fixed staged fallback without candidate validation. The original runner instead evaluated the zero-parameter hybrid for this seed. Its reported deployed-policy final-test rows were deterministically replaced by the **already measured** fixed staged rows on exactly matched scenarios. The original records and a fallback-correction ledger are retained; this correction does not represent additional simulations.

The complete offline episode budget and final-test count are summarized in Table 4; the table is regenerated from the experimental manifest and decision records:

Table 4: Native `PSO+TR` reconstructed-simulator episode accounting across all eight optimizer seeds. Executed training episodes include feasible `TR` probes and trials. The charged training workload is 15,360 rollout equivalents; infeasible attempts do not execute episodes. Validation is skipped for seed 4, which has no eligible screening candidate. The final-test count refers to corrected archive records.

Stage	Count
Training: 851 executed objective calls $\times$ 16 episodes	13616
Candidate screening: 104 candidates $\times$ 2 groups $\times$ 64	13312
Fixed staged screening: 8 seeds $\times$ 2 groups $\times$ 64	1024
Frozen-candidate validation: 7 seeds $\times$ 2 groups $\times$ 512	7168
Fixed staged validation: 7 seeds $\times$ 2 groups $\times$ 512	7168
Offline total: executed episodes	42288
Final-test records: 8 seeds $\times$ 4 policies $\times$ 3 groups $\times$ 400	38400

The numerical results, controller comparisons, and `TR` diagnostics are reported next. An adopted controller is a policy that satisfies the independent deployment criteria, whereas an accepted `TR` step is an internal optimization event; the two statistics measure different outcomes. Demonstrating the incremental benefit of the `TR` safeguard would require a matched robotic `PSO`-only ablation with the same simulation model, evaluation budget, scenario design, and selection protocol.

6 PSO+TR robotic controller-tuning results

To assess the complete tuning and deployment pipeline, we compare its final-test outcomes with three fixed controllers on paired nominal, development, and withheld stress scenarios.

Eight independent optimization runs tune the same 14-parameter hybrid controller on the reconstructed MATLAB planar contact simulator. The historical robotic MAES measurements are not pooled with these PSO+TR results. Each run uses ten particles, the independently selected TR configuration, and a fixed training batch of eight nominal and eight development scenarios shared across feasible candidate evaluations within that run.

The optimization budget is 120 charged evaluation attempts per seed, or 1,920 charged training rollout equivalents. Across the eight runs, 109 of the 960 charged attempts are infeasible and do not execute the simulator batch. The remaining 851 feasible evaluations execute 13,616 training episodes. Initialization evaluations, particle evaluations, TR model probes, and TR trials share this budget, including infeasible attempts.

Each seed screens 13 distinct candidates selected from the returned incumbent and the finite population archive. All candidates and the fixed staged reference use the same 64 fresh scenarios per nominal and development distribution within that seed. When screening selects an eligible candidate, that frozen candidate and the reference are validated on 512 new paired scenarios per distribution. Final testing uses 400 new scenarios per seed, distribution, and controller. The final-test archive contains 38,400 records across four controllers. Scenario streams are independent across seeds, while controller comparisons within each seed use identical test scenarios.

Seed 4 produced no eligible screening candidate and therefore used the fixed staged fallback without candidate validation. The original run evaluated the zero-parameter hybrid controller as the fallback. The published dataset replaces the corresponding PSO+TR records with the already measured fixed staged outcomes on the identical paired final-test scenarios. The unmodified seed checkpoints and deterministic correction audit are retained. The corrected MATLAB runner explicitly implements the fixed staged fallback.

Table 5 reports final-test outcomes as means and sample standard deviations across the eight optimizer seeds. Figure 4 displays the corresponding controller comparisons. Cost averages include successful, unsuccessful, and fallback outcomes.

Under development conditions, the deployed PSO+TR pipeline achieves 97.78% completion versus 91.22% for fixed staged. On the stress scenarios, completion is 58.00% versus 31.12%, a paired difference of 26.88 percentage points. The corresponding stress violation fractions are 2.19% and 20.69%, and mean costs are 10.094 and 13.124, respectively. These are descriptive contrasts across eight optimization seeds, not simultaneous confidence bounds or a direct PSO-only robotic ablation.

Table 6 lists the seed-level optimization and deployment decisions, while Figure 6 displays the TR intervention counts. Seven of the eight optimization runs produced a screening-eligible candidate. All seven frozen candidates were adopted after independent validation; seed 4 used the fixed staged fallback. Across all seeds, PSO+TR attempted 11 conditional TR invocations and accepted 2 TR steps, consuming 226 charged TR evaluation attempts. The accepted steps occurred in seeds 1 and 5. Adoption of a final controller and acceptance of a

Table 5: Robotic final-test outcomes across eight optimizer seeds.

Group	Method	Completion (%)	Violation (%)	Cost
nominal	PSO+TR	100.00 ± 0.00	0.00 ± 0.00	3.050 ± 0.581
nominal	Fixed staged	100.00 ± 0.00	0.00 ± 0.00	4.414 ± 0.032
nominal	Continued search	100.00 ± 0.00	0.00 ± 0.00	4.728 ± 0.036
nominal	Fixed hybrid	100.00 ± 0.00	0.00 ± 0.00	4.414 ± 0.032
development	PSO+TR	97.78 ± 3.52	0.00 ± 0.00	4.140 ± 1.033
development	Fixed staged	91.22 ± 1.22	0.00 ± 0.00	6.415 ± 0.176
development	Continued search	81.88 ± 2.19	1.25 ± 0.65	7.345 ± 0.213
development	Fixed hybrid	91.22 ± 1.22	0.00 ± 0.00	6.482 ± 0.177
stress	PSO+TR	58.00 ± 13.05	2.19 ± 5.99	10.094 ± 1.789
stress	Fixed staged	31.12 ± 1.87	20.69 ± 2.54	13.124 ± 0.312
stress	Continued search	23.84 ± 1.89	34.59 ± 2.22	13.628 ± 0.302
stress	Fixed hybrid	29.66 ± 2.20	21.28 ± 2.61	13.391 ± 0.333

TR step are separate events. These sparse TR improvements do not identify a causal benefit of TR relative to a matched PSO-only robotic experiment.

Table 6: Seed-level optimization and controller-adoption decisions.

Seed	Adopted	Charged evals	TR calls	TR accepted	Charged TR evals
0	1	120	1	0	20
1	1	120	2	1	50
2	1	120	1	0	10
3	1	120	2	0	40
4	0	120	2	0	40
5	1	120	3	1	66
6	1	120	0	0	0
7	1	120	0	0	0
Total	7	960	11	2	226

Table 7 reports differences between the deployed pipeline and fixed staged, formed within each seed before computing their means and sample standard deviations across seeds. Here, pp denotes percentage points; positive completion and negative violation or cost differences favor PSO+TR. Nominal completion is unchanged, while mean cost differences favor the deployed pipeline in all three distributions.

Table 7: Paired final-test differences: deployed PSO+TR minus fixed staged.

Distribution	Δ completion (pp)	Δ violation (pp)	Δ cost
nominal	0.00 ± 0.00	0.00 ± 0.00	-1.365 ± 0.587
development	6.56 ± 2.97	0.00 ± 0.00	-2.275 ± 0.965
stress	26.88 ± 12.04	-18.50 ± 7.73	-3.030 ± 1.584

Figure 3 shows training incumbent scores versus charged training rollout equivalents, computed as 16 times the number of charged evaluation attempts. Infeasible attempts advance this count without executing the simulator. Each seed uses its own fixed training batch, so the scores refer to different sampled objectives across seeds. Figure 5 complements the mean-cost comparisons with empirical episode-cost distributions by controller and

scenario group, using all 38,400 final-test records, including unsuccessful and fixed staged fallback outcomes.

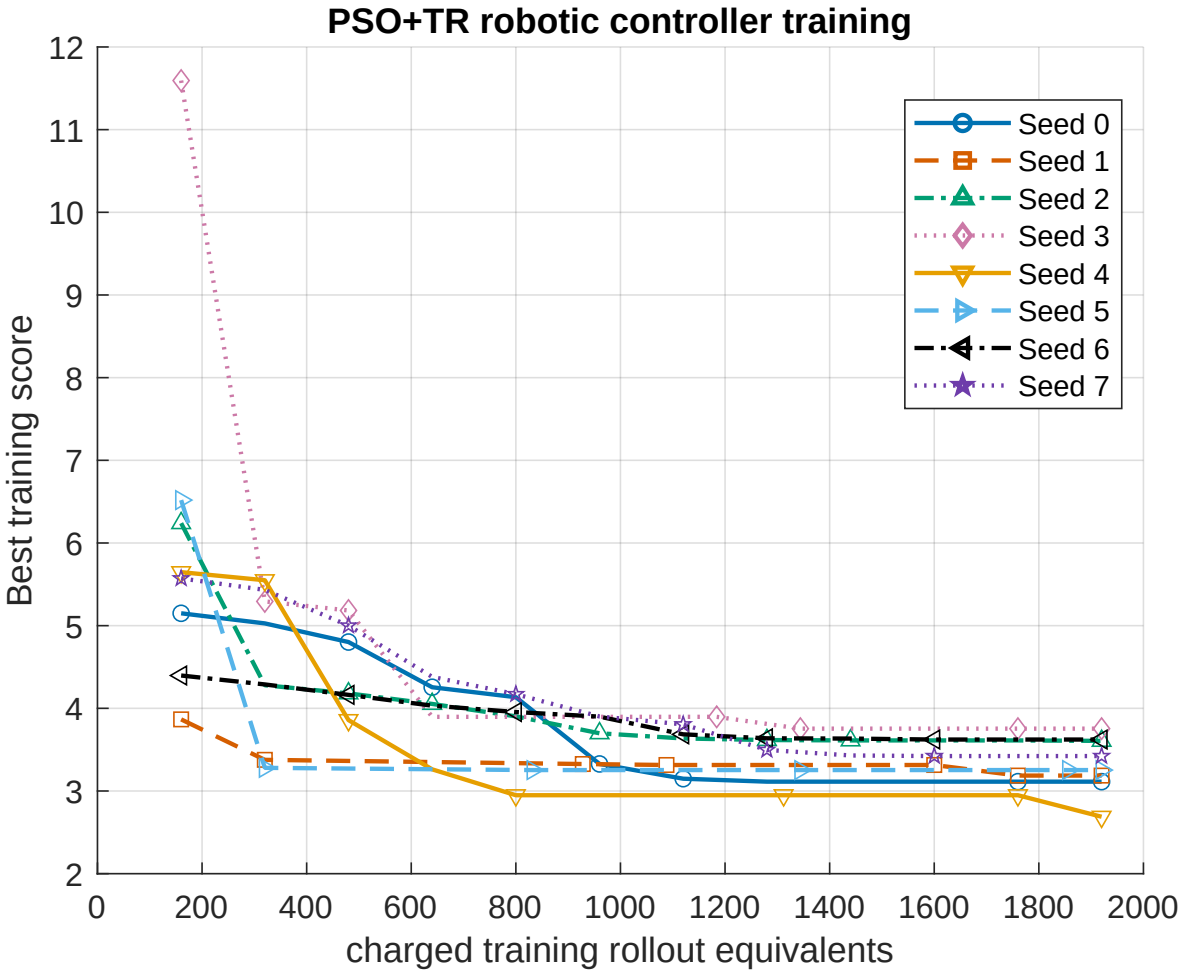

Figure 3: PSO+TR training scores versus charged training rollout equivalents.

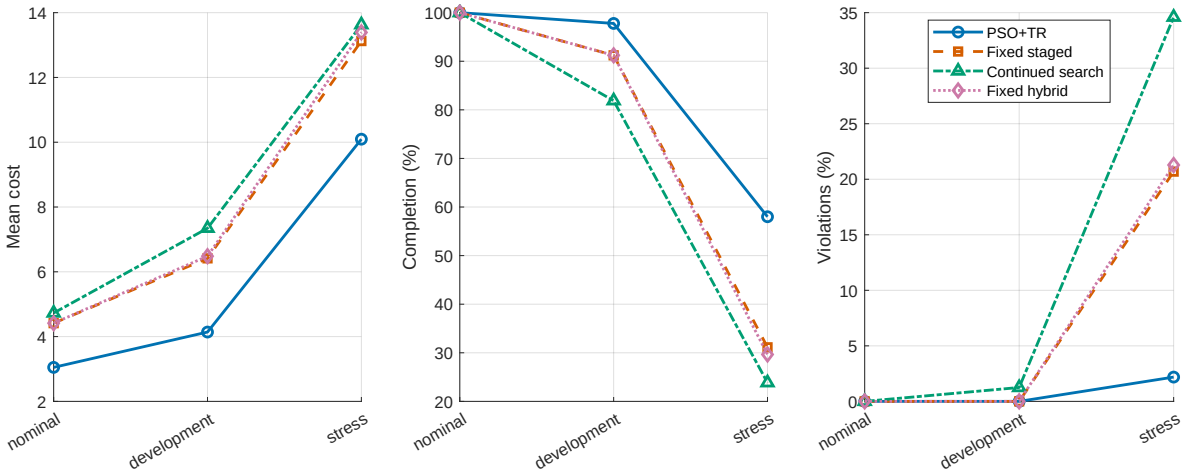

Figure 4: Deployed-policy outcomes on paired final-test scenarios.

The simulator is synthetic and represents local planar motion: it omits articulated joints, gravity, sensor and actuator dynamics, and three-dimensional contact. The tuned controller is not certified on physical hardware. Fixed training batches support consistent comparisons of TR center, probe, and trial values within each seed; the stress distribution is reserved

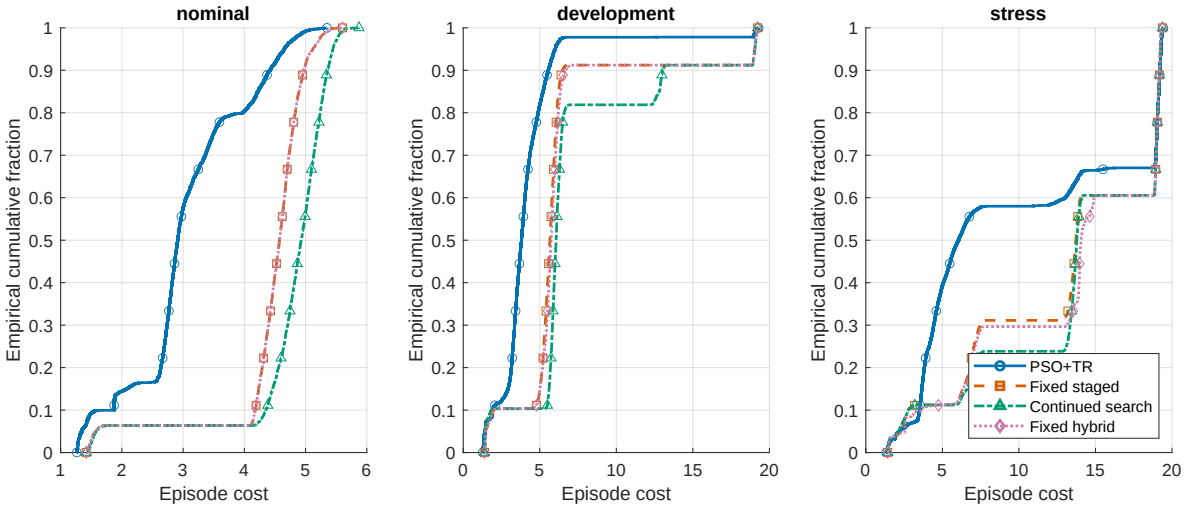

Figure 5: Empirical final-test cost distributions by controller and scenario group.

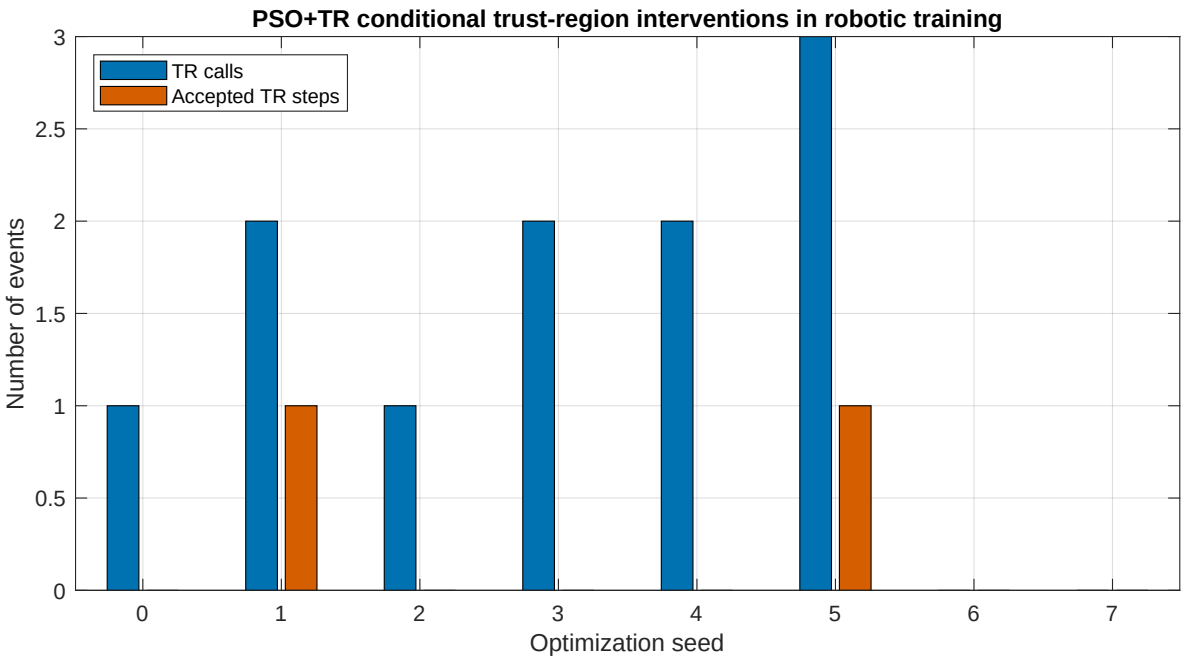

Figure 6: Conditional TR invocations and accepted steps by optimizer seed.

for final evaluation. The results assess the complete tuning and deployment pipeline on this simulator. They do not isolate the contribution of TR or establish robustness to other contact models.

7 Numerical optimizer comparison on BCP problems

To assess target attainment and evaluation efficiency, we compare PSO+TR with four other optimization methods on bound-constrained BCP problems under matched objective-evaluation budgets. We specify the benchmark eligibility criteria, solver configurations, evaluation accounting, and empirical target definitions used in the comparison. This experiment assesses numerical optimization performance separately from robotic controller selection and validation.

Benchmark and comparison protocol. The BCP collection contains 326 problem definitions. We evaluate 93 eligible problems with dimensions $1 \leq n \leq 12$. These problems have continuous decision variables, finite componentwise bounds, no additional nonlinear constraints, and a finite, real objective value at the repaired, box-projected initial point. Five methods are compared: PSO+TR, PSO [8], VRBBO [6], MAES [36], and DE [52].

Each method is tested with three independent random seeds on every problem, yielding $93 \times 3 = 279$ matched problem–seed instances per method and $5 \times 279 = 1{,}395$ solver runs in total. For a given instance, all methods receive the same prescribed initial candidate and the same maximum number of objective evaluations,

$$N_{\max}(n) = \max\{100, \lceil 100n \rceil\}. \tag{39}$$

Every solver evaluation attempt is charged to this allowance, including initialization, model probes, infeasible attempts, and unsuccessful trials of the conditional TR safeguard. An infeasible attempt can be charged without calling the underlying objective. The initial eligibility check is performed before the solver runs and is outside their evaluation budgets. The common budget does not imply identical internal initialization or identical evaluation overhead across solvers.

Evaluation accounting and solver configurations. Particle evaluations, forward model probes, and all TR trial evaluations share the same global evaluation budget. A TR invocation is eligible only after a complete non-improving generation and is limited to 32 charged evaluation attempts within that budget. It may consume fewer attempts when a model fails early or either evaluation allowance is exhausted.

Table 8 collects the shared PSO+TR configuration and fixed implementation constants. The archived parameter-selection record reports a budget factor of 1,000; its configuration was transferred without retuning to this BCP comparison, which uses factor 100, and to the reconstructed MATLAB controller experiment. Not every constant in the table was varied in the parameter-selection study. The present comparison therefore measures performance under its stated budget rather than parameter optimality for that budget. Robotic controller selection and outcomes are reported in Sections 5 and 6. The combined stochastic method is assessed empirically; no global or stationary-point convergence guarantee is asserted.

In the numerical eigenvalue-threshold row, β_i are the eigenvalues of the reduced curvature matrix, and $a = U^\top c$ contains the model's linear coefficients in its orthonormal eigenbasis U.

In the BCP runs, PSO+TR, PSO, and DE use $N = \max\{12, \min\{40, 5n\}\}$ population members. Their initial positions are sampled uniformly in the box, with the first position replaced by the prescribed initial candidate x^0. All initialization evaluations are charged. The robotic runs instead use $N = 10$, $x^0 = \mathbf{0}_{14}$, and 120 charged attempts per seed; the shared settings in Table 8 are unchanged.

The BCP random seeds are $1, 2, 3$. PSO+TR uses separate Mersenne Twister streams for swarm and subspace draws. For solver seed s, the subspace stream uses $(s{+}1{,}000{,}003) \bmod (2^{32}{-}1)$; the robotic runner uses solver seed $5000 + s$ for experiment labels $s = 0, \ldots, 7$. Maximum-correlation ties in basis-column selection use the first maximizing index. If the initial Gaussian basis requires rank completion, at most $20 \max\{n, m\}$ additional Gaussian draws are attempted before canonical coordinate directions are used.

Table 8: Shared PSO+TR settings in the reported BCP and robotic experiments. The first group records the transferred configuration; the second lists fixed implementation constants.

Setting	Value or rule
Transferred configuration	
Inertia weight ω	0.7
Cognitive and social coefficients (c_1, c_2)	$(1.4, 1.4)$
Reduced dimension m	$\min\{n, 10\}$
Initial radius Δ_0	0.2
Refresh interval	Five non-improving generations
Acceptance threshold η_{accept}	0.2
Ridge weight λ_{ridge}	10^{-12}
Per-invocation evaluation cap	32 charged attempts
Fixed implementation constants	
Minimum radius $\Delta_{\min}$	10^{-8}
Absolute improvement tolerance ε_{imp}	0
Radius multiplier on a permitted retry	1/2
Initial velocity scale relative to box width	0.1
Forward-probe length h_k	$\min\{\Delta_k, \max\{\Delta_k/2, \Delta_{\min}\}\}$
Geometry-factor normalization	$\|F\|_F^2 = 0.95n$ for nonempty F
Regularization weight α	0.05; 1 when F is empty
Zero-geometry cutoff $\max_{i,j} \|D_{ij}\|$	10^{-15}
Displacement rejection threshold	$10^{-12} \max\{1, \|x^-\|_2\}$
Minimum retained residual norm	Strictly greater than 10^{-10}
Reorthogonalization passes	2
Basis orthogonality-check tolerance	10^{-8} in Frobenius norm
Eigenvector orthogonality-check tolerance	10^{-6} in Frobenius norm
Numerical eigenvalue threshold	$10^{-12} \max\{1, \max_i \|\beta_i\|, \|a\|_2\}$
Secular-root bracketing and bisection limits	200 and 100 iterations
Maximum Gaussian draws per random refresh	10

Table 9 records the principal settings of the four comparator implementations. The separate PSO comparator starts with zero velocities and retains velocity components when positions are projected onto the box. PSO+TR uses randomly initialized velocities and zeros the affected velocity components after projection. Thus the comparison includes implementation differences beyond the presence of the TR safeguard.

Table 9: Comparator configurations in the supplied BCP runner. Each method uses the common budget in (39).

Method	Configuration
PSO	$(\omega, c_1, c_2) = (0.7, 1.4, 1.4)$; zero initial velocities; componentwise position projection.
DE	DE/rand/1 mutation with scale 0.7; binomial crossover probability 0.85, with one forced coordinate; componentwise trial projection; strict-improvement replacement.
MAES	Offspring count $\lambda = \max\{8, 4 + \lfloor 3 \log n \rfloor\}$; parent count $\mu = \lfloor \lambda/2 \rfloor$; initial mean x^0, transformation I_n, and zero evolution path; initial scale $0.15\,\text{median}_j(u_j - \ell_j)$, with floor 10^{-12}; projected offspring and recombined mean; no TR rescue.
VRBBO	Archived implementation with its default tuning structure (alg=5); initial point x^0; evaluation limit $N_{\max}$; no finite time limit or objective target. The counted oracle enforces the box by penalizing invalid attempts.

For the MAES implementation, the normalized recombination weights are

$$w_j = \frac{\log(\mu + 1/2) - \log j}{\sum_{\ell=1}^{\mu}[\log(\mu + 1/2) - \log \ell]}, \qquad j = 1, \ldots, \mu, \qquad \mu_{\text{eff}} = \frac{1}{\sum_{j=1}^{\mu} w_j^2}.$$

Its path, rank-one, rank-μ, and step-scale coefficients are

$$c_\sigma = \frac{\mu_{\text{eff}} + 2}{n + \mu_{\text{eff}} + 5}, \qquad c_{\text{one}} = \frac{2}{(n + 1.3)^2 + \mu_{\text{eff}}},$$
$$c_\mu = \min\left\{1 - c_{\text{one}}, \frac{2(\mu_{\text{eff}} - 2 + 1/\mu_{\text{eff}})}{(n + 2)^2 + \mu_{\text{eff}}}\right\}, \qquad d_\sigma = 1 + c_\sigma + 2\max\left\{0, \sqrt{\frac{\mu_{\text{eff}} - 1}{n + 1}} - 1\right\}.$$

The step-scale update uses $\chi_n = \sqrt{2}\,\Gamma((n + 1)/2)/\Gamma(n/2)$, where Γ is the gamma function. These dimension-dependent coefficients follow the recorded implementation and were not retuned for this comparison.

Target attainment and evaluation-count profiles. Let $\mathcal{P}$ be the set of 279 matched problem–seed instances and $\mathcal{A}$ the five solvers. Here $p \in \mathcal{P}$ indexes an instance and $s \in \mathcal{A}$ indexes a solver, rather than a random seed. For attempted evaluation e, let $x_{p,s}^{(e)}$ be its trial point and $\widehat{f}_{p,s}^{(e)}$ the recorded oracle value, which equals $f_p(x_{p,s}^{(e)})$ for a valid finite evaluation. Define the recorded best-value trace

$$F_{p,s}(r) = \min_{1 \le e \le r} \widehat{f}_{p,s}^{(e)}, \qquad f_{p,s}^{\text{final}} = F_{p,s}(N_{p,s}^{\text{used}}), \qquad f_p^\star = \min_{s \in \mathcal{A}} f_{p,s}^{\text{final}}.$$

Here, $N_{p,s}^{\text{used}}$ is the charged evaluation count at termination, distinct from the first-hit count $N_{p,s}$. Invalid attempts receive $+\infty$ in PSO+TR and the finite penalty realmax/10^6 in the four comparator implementations. The comparator box checks allow a tolerance of 10^{-10}; PSO+TR uses strict box checks. The initial value is $f_p^0 = f_p(x_p^0)$. The trace includes finite model probes and rejected TR trials, so its best value can differ from the accepted incumbent returned by the solver. In Eq. (41), # denotes set cardinality. We use the instance-dependent target

$$t_p = f_p^\star + 0.01\max\left\{f_p^0 - f_p^\star, \epsilon_{\text{mach}}\max(1, |f_p^\star|)\right\}. \tag{40}$$

Thus an algorithm reaches the target when at most 1% of the observed initial-to-best gap remains, up to the indicated numerical tolerance. Here, $\epsilon_{\text{mach}} = 2^{-52} \simeq 2.220446049250313 \times 10^{-16}$ is the spacing at one in IEEE binary64 arithmetic, as implemented by eps(1). These targets are empirical references rather than certified global optima.

Let $N_{p,s}$ denote the number of charged evaluation attempts needed by solver s to reach t_p for the first time, with $N_{p,s} = +\infty$ when the target is not attained. Define $H_p = \min_{a \in \mathcal{A}} N_{p,a}$. Following Dolan and Moré [53] and Moré and Wild [4], the performance and data profiles are

$$\begin{aligned} \rho_s(\tau) &= \frac{1}{279}\#\left\{p : H_p < +\infty,\ N_{p,s} \le \tau H_p\right\}, \\ \delta_s(\kappa) &= \frac{1}{279}\#\left\{p : N_{p,s} \le \kappa(n_p + 1)\right\}. \end{aligned} \tag{41}$$

Here, $\tau \geq 1$, $\kappa \geq 0$, and n_p is the problem dimension. Instances without target attainment by any method contribute zero to both profiles while remaining in the common denominator. The explicit condition $H_p < +\infty$ avoids an undefined ratio for such instances. Final solution quality is further summarized by the instance-normalized regret

$$r_{p,s} = \frac{\max\{0, f_{p,s}^{\text{final}} - f_p^\star\}}{\max\{1, |f_p^\star|\}}. \tag{42}$$

The regret normalization allows comparison across problems with different objective scales but can still produce large values on a few instances.

Figure 7 compares relative first-hit efficiency in panel (a) with target attainment under a dimension-normalized evaluation allowance in panel (b). At $\tau = 1$, `PSO+TR` has the largest fraction of fastest recorded first hits, including ties, at 35.13%. Both profiles approach the target-attainment fractions reported below.

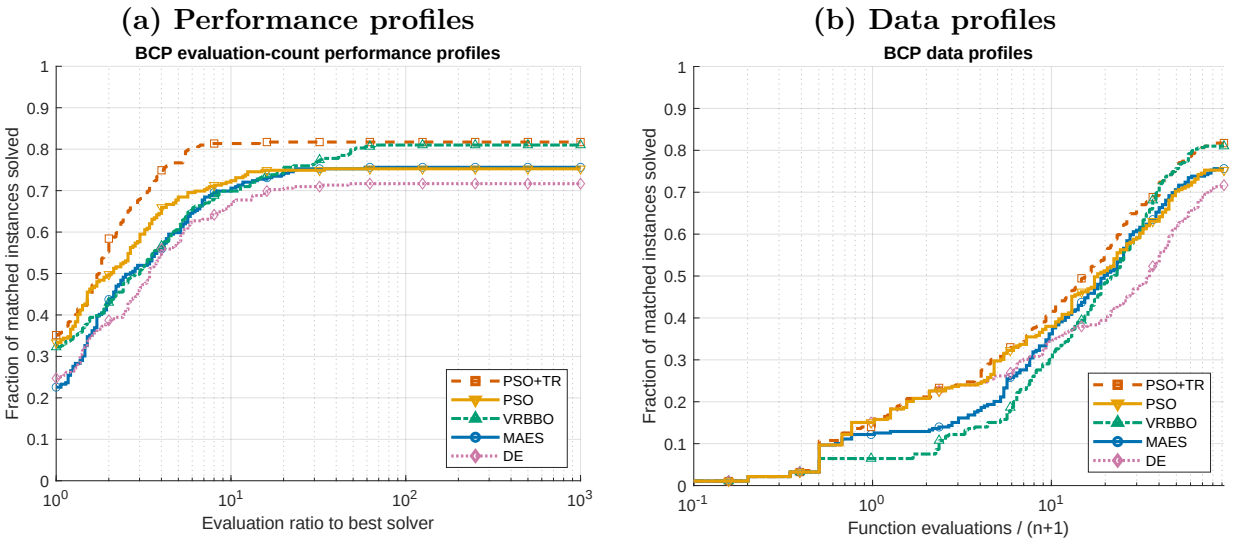

Figure 7: Evaluation-count profiles for five DFO methods on 279 matched `BCP` problem–seed instances (93 problems, three seeds). (a) Target attainment within a factor τ of the smallest observed first-hit count on each instance. (b) Target attainment within $\kappa(n_p + 1)$ charged evaluations. Higher curves indicate greater attainment at the stated allowance. Unsolved instances remain in the common denominator, and all methods use matched evaluation budgets.

Comparison results. Table 10 presents the five-solver results on 279 matched problem–seed instances. All methods use identical objective-evaluation limits on each matched instance, with targets allowing 1% of the initial-to-best observed objective gap. The proposed `PSO+TR` reaches the common target on 81.72% of the instances, compared with 81.00% for `VRBBO`, 75.63% for `MAES`, 75.27% for `PSO`, and 71.68% for `DE`. Table 11 summarizes the pairwise comparison, with a target-attainment advantage of 6.45 percentage points for `PSO+TR` over `PSO`.

The median first-hit evaluation counts, computed separately on each method's successful instances, are 41.5 for `PSO+TR`, 54 for `VRBBO`, 44 for `MAES`, 36 for `PSO`, and 46 for `DE`. Thus, the smaller conditional median for `PSO` coexists with its lower overall target-attainment fraction.

Mean regret averages instance-normalized regret values rather than raw objective values. "Tied best" counts instances satisfying $|f_{p,s}^{\text{final}} - f_p^\star| \leq 10^{-10} \max\{1, |f_p^\star|\}$; several methods may be counted on the same instance. VRBBO records the largest number of tied-best final values, whereas PSO has a smaller mean regret than PSO+TR. The observed ranking therefore depends on the criterion.

These differences are descriptive: three stochastic repetitions of the same 93 problems do not constitute 279 independent problem families, and no statistical significance or universal superiority is inferred. Moreover, the separate PSO implementation differs from PSO+TR in swarm initialization and boundary-velocity handling, so their comparison does not isolate the contribution of the TR safeguard.

Table 10: Five-solver BCP results on 279 matched problem–seed instances.

Method	Solved (%)	Median evaluations	Mean regret	Tied best
MAES	75.63	44	1.494×10^4	59
PSO+TR	81.72	41.5	15.27	89
VRBBO	81.00	54	7.127×10^3	148
DE	71.68	46	10.95	63
PSO	75.27	36	0.9065	72

Table 11: PSO+TR and baseline PSO results on the same 279 matched BCP instances.

Method	Solved (%)	Median evaluations	Tied best
PSO+TR	81.72	41.5	89
PSO	75.27	36	72

Conditional TR and subspace diagnostics. Over the 279 matched instances, PSO+TR makes 1,409 conditional TR invocations, accepts 1,687 TR steps, and expends 32,988 objective evaluations on TR operations, including model construction. These quantities are compatible with a successive-TR mechanism: one invocation may accept more than one local step. Consequently, 119.7 accepted steps per 100 invocations is a frequency, not a probability or percentage of accepted invocations. The logs record 3,046 basis-column updates and 77 refreshes. An update following an accepted TR step preserves the subspace span and can only change the selected column's sign in exact arithmetic; these update counts therefore do not count distinct changes of span. The maximum observed orthogonality residual is $\|S^\top S - I\|_F = 6.99 \times 10^{-16}$. These data document the safeguard's computational activity, but do not by themselves identify its incremental benefit relative to an otherwise identical PSO run.

Table 12 collects these intervention counts and numerical orthogonality diagnostics. Figure 8 displays the target-attainment fractions: PSO+TR solves 228 instances and VRBBO solves 226, so their observed difference corresponds to two instances. Figure 9 exposes problem-level variation in the mean transformed regret; the lowest plotted values occur for different methods on different problems.

Exploratory hyperparameter sensitivity. The accompanying sensitivity figures summarize a separate exploratory training grid in which the initial TR radius and subspace

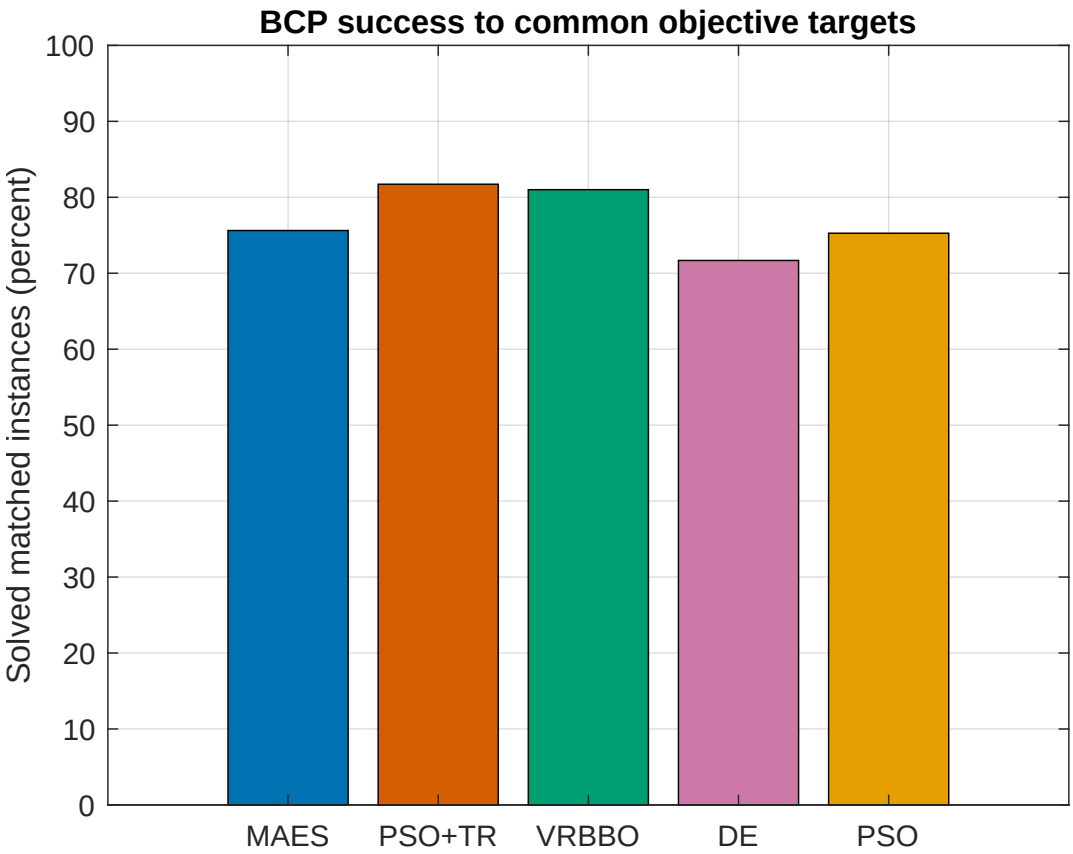

Figure 8: Observed proportion of the 279 matched `BCP` instances meeting the same instance-specific objective target for each of the five methods.

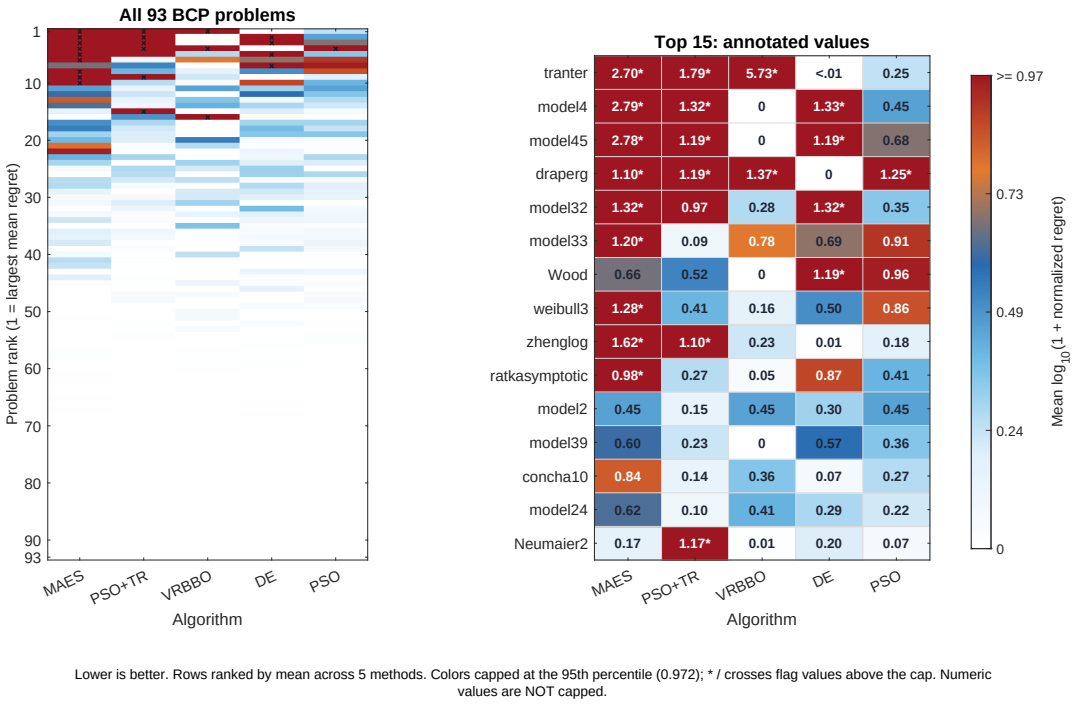

Figure 9: Problem-level mean $\log_{10}(1 + \text{normalized regret})$ over three seeds. Each row is one of 93 eligible `BCP` problems; `MAES` and `PSO+TR` remain separate methods.

Table 12: PSO+TR safeguard diagnostics for the BCP comparison.

Metric	Value
TR invocations	1409
Accepted TR steps	1687
Accepted TR steps per 100 invocations	119.7
TR objective evaluations	32988
Subspace updates	3046
Subspace refreshes	77
Maximum $\|S^\top S - I\|_F$	6.99e-16

refresh interval were varied, while the other displayed parameters were held fixed. The saved parameter-selection record reports a budget factor of 1,000, whereas the final BCP comparison uses a factor of 100. The plotted grid illustrates sensitivity within its own experimental setting; its fixed-parameter panels provide no information about varying those parameters. No final-comparison seed was used to select the configuration reported above.

Figures 10 and 11 show the lowest mean training regret at the tested initial radius $\Delta_0 = 0.2$ for both tested refresh intervals, three and five generations. Figure 12 presents the coordinate marginals; its six fixed-parameter panels describe the tested settings and do not establish sensitivity to those parameters.

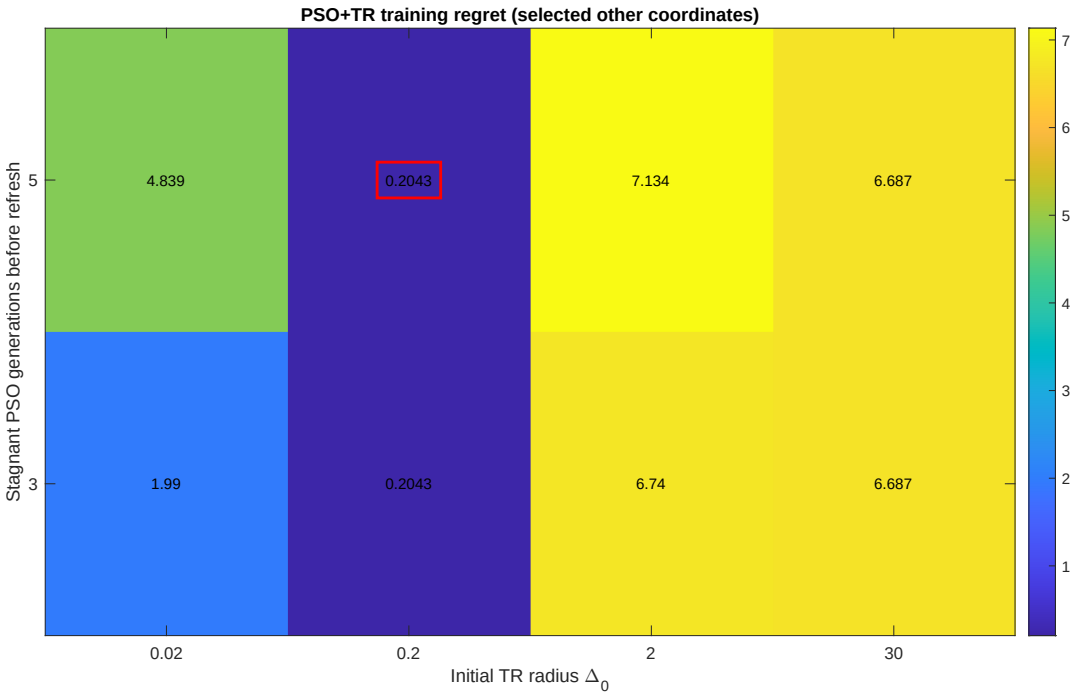

Figure 10: PSO+TR training normalized-regret grid as a function of the initial TR radius and subspace refresh interval; other grid coordinates are held at their selected values.

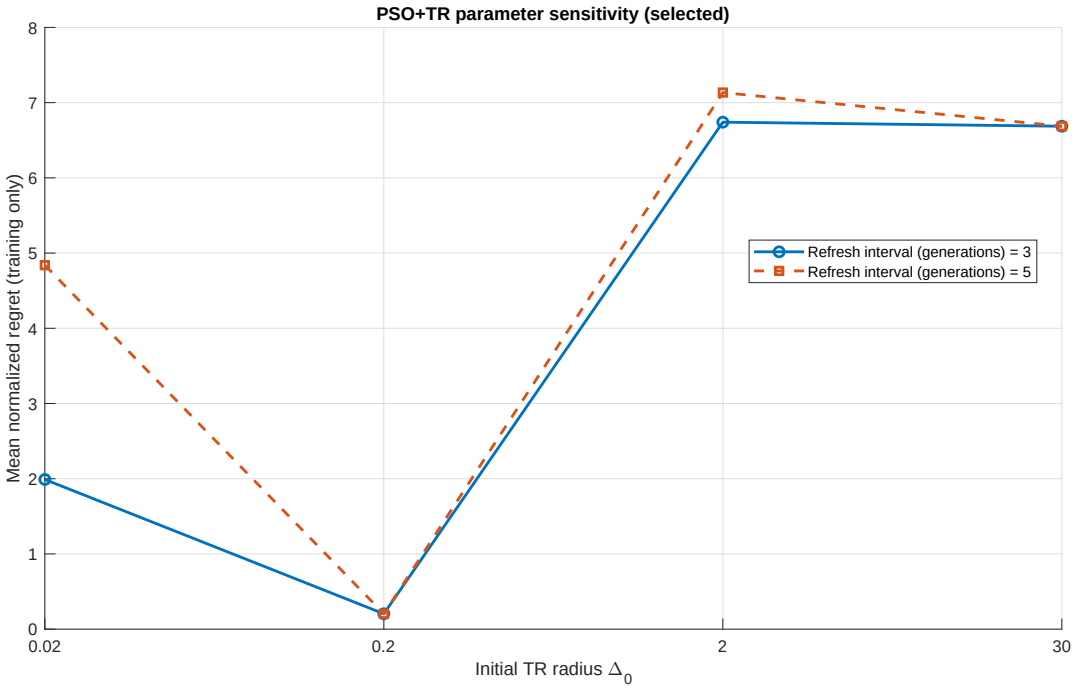

Figure 11: PSO+TR training-grid sensitivity, without holdout-based model selection.

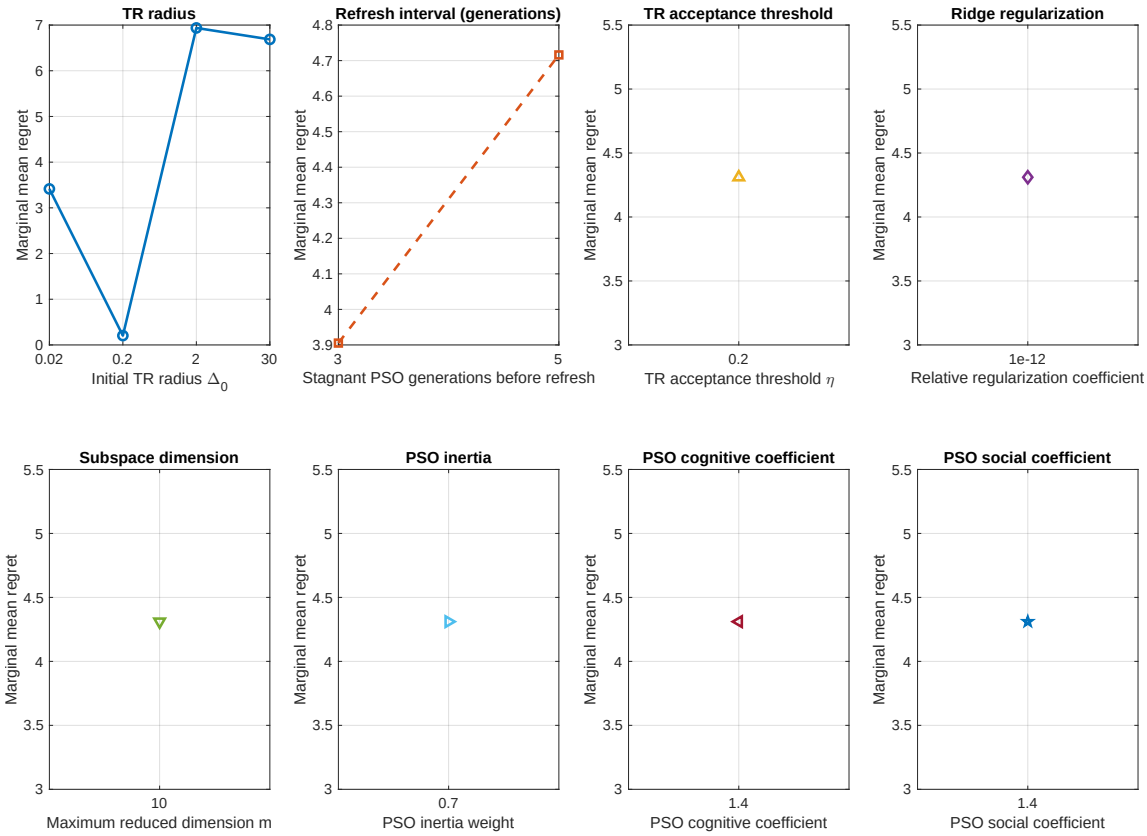

Figure 12: Marginal training-grid regret by parameter. The six parameters held fixed in the archived eight-configuration grid have only one value, so those panels are not evidence of sensitivity to those parameters.

8 Discussion and Conclusion

To distinguish supported conclusions from remaining questions, we interpret the BCP optimizer comparison and robotic controller results together and identify the controlled ablations still needed. The benchmark evaluates target attainment and evaluation efficiency, while the robotic experiment assesses the complete tuning and deployment pipeline.

PSO+TR combines population exploration with conditional, successive reduced-subspace TR steps. Its persistent basis and regularized swarm-geometry surrogate support local refinement without constructing a full-dimensional curvature matrix. All attempted evaluations are charged to a common objective-evaluation budget. Controlled ablations are needed to isolate the contributions of the TR safeguard, subspace updates, and curvature construction.

On the 93 BCP problems, PSO+TR achieves the highest target-attainment rate among the five compared methods, reaching the prescribed target on 81.72% of the 279 matched instances, compared with 75.27% for PSO and 81.00% for VRBBO. Nevertheless, PSO has a lower median first-hit evaluation count among successful instances and a lower mean normalized regret, indicating that no method dominates across all performance measures. The TR safeguard records 1,409 invocations and 1,687 accepted steps, consuming 32,988 objective evaluations. These findings demonstrate the practical occurrence of successive local improvements, although differences between the compared implementations prevent attributing the overall performance gain solely to the TR component.

In the robotic DFPO study, the validated PSO+TR deployment pipeline achieves completion rates of 97.78% and 58.00% under development and withheld stress conditions, respectively, compared with 91.22% and 31.12% for the fixed staged controller. Under stress conditions, the violation rate decreases from 20.69% to 2.19%, while the mean episode cost decreases from 13.124 to 10.094. Seven of eight optimized controllers pass independent validation. However, only two local steps are accepted across 11 robotic TR invocations. Consequently, the observed improvements characterize the complete tuning and validation pipeline rather than the isolated contribution of the TR safeguard.

Overall, the results establish competitive performance on the considered DFO benchmarks and demonstrate the potential of PSO+TR for tuning interpretable robotic insertion controllers. Nevertheless, the robotic experiments rely on a synthetic planar contact model, and their results do not establish physical-robot performance or safety guarantees. Future research will focus on controlled PSO-only ablations, sensitivity to algorithmic parameters and contact uncertainty, more realistic three-dimensional assembly models, and experimental validation on physical robots.

A Seed-level outcomes of the robotic controller-tuning experiment

To expose variation hidden by aggregate results, Table 13 reports completion and violation counts, mean episode cost, and mean termination time for each of the eight independent optimizer seeds. Within each seed, the four controller configurations use matched final-test scenarios, with 400 episodes per distribution and controller.

Seed 4 produced no eligible screening candidate and therefore deployed the fixed staged reference without candidate validation. Its reported deployed-policy outcomes are the previously observed fixed staged outcomes on those same paired scenarios. No additional simulation episodes were generated for this correction.

Averaging the deployed-policy completion fractions across the eight seeds gives 97.78% for the development distribution and 58.00% for the stress distribution, consistent with the aggregate results. Variation across seeds combines differences in the independent optimization runs and in the separately drawn final-test scenarios; it therefore does not isolate variability due to optimization alone.

Table 13: Native `PSO+TR` deployed-policy results for all eight seeds on the reconstructed simulator (400 paired cases per group and seed). Here, mean time is the average recorded termination time T_e over all 400 episodes, including timeouts and protective terminations.

Group	Seed	Successes	Violations	Mean cost	Mean time (s)
nominal	0	400	0	2.709	2.165
nominal	1	400	0	3.263	2.585
nominal	2	400	0	2.835	2.447
nominal	3	400	0	2.930	2.373
nominal	4	400	0	4.401	3.547
nominal	5	400	0	2.573	2.105
nominal	6	400	0	2.795	2.268
nominal	7	400	0	2.890	2.471
development	0	398	0	3.591	3.290
development	1	396	0	4.038	3.334
development	2	400	0	3.573	3.589
development	3	398	0	3.952	3.795
development	4	358	0	6.650	5.351
development	5	394	0	3.613	3.409
development	6	386	0	4.019	3.409
development	7	399	0	3.686	3.638
stress	0	233	0	10.333	5.914
stress	1	197	2	11.836	6.412
stress	2	265	0	8.634	5.439
stress	3	253	0	9.346	5.639
stress	4	119	68	13.467	7.226
stress	5	282	0	7.988	4.858
stress	6	246	0	9.946	5.753
stress	7	261	0	9.202	5.732

Funding

M.K. acknowledges financial support of the Austrian Science Fund (FWF) under https://doi.org/10.55776/PAT2747625.

References

[1] C. Audet and W. Hare. Derivative-Free and Blackbox Optimization. Springer Series in Operations Research and Financial Engineering. Springer International Publishing, 2017.

[2] A. R. Conn, K. Scheinberg, and L. N. Vicente. Introduction to Derivative-Free Optimization. MOS–SIAM Series on Optimization. SIAM, Philadelphia, 2009.

[3] J. Larson, M. Menickelly, and S. M. Wild. Derivative-free optimization methods. Acta Numerica, 28:287–404, 2019.

[4] J. J. Moré and S. M. Wild. Benchmarking derivative-free optimization algorithms. SIAM Journal on Optimization, 20(1):172–191, 2009.

[5] L. M. Rios and N. V. Sahinidis. Derivative-free optimization: A review of algorithms and comparison of software implementations. Journal of Global Optimization, 56(3):1247–1293, 2013.

[6] M. Kimiaei and A. Neumaier. Efficient unconstrained black box optimization. Mathematical Programming Computation, 14:365–414, 2022.

[7] M. Kimiaei. An improved randomized algorithm with noise level tuning for large-scale noisy unconstrained DFO problems. Numerical Algorithms, 2026.

[8] J. Kennedy and R. Eberhart. Particle swarm optimization. In Proceedings of ICNN'95 – International Conference on Neural Networks, volume 4, pages 1942–1948, 1995.

[9] A. I. F. Vaz and L. N. Vicente. A particle swarm pattern search method for bound constrained global optimization. Journal of Global Optimization, 39:197–219, 2007.

[10] J. J. Moré and D. C. Sorensen. Computing a trust region step. SIAM Journal on Scientific and Statistical Computing, 4(3):553–572, 1983.

[11] C. Cartis and L. Roberts. Scalable subspace methods for derivative-free nonlinear least-squares optimization. Mathematical Programming, 199:461–524, 2023.

[12] M. J. D. Powell. UOBYQA: unconstrained optimization by quadratic approximation. Mathematical Programming, 92:555–582, 2002.

[13] M. J. D. Powell. The NEWUOA software for unconstrained optimization without derivatives. In G. Di Pillo and M. Roma, editors, Large-Scale Nonlinear Optimization, volume 83 of Nonconvex Optimization and Its Applications, pages 255–297. Springer, 2006.

[14] M. J. D. Powell. The BOBYQA algorithm for bound constrained optimization without derivatives. Technical Report DAMTP 2009/NA06, Department of Applied Mathematics and Theoretical Physics, University of Cambridge, 2009.

[15] Serge Gratton, Ph. L. Toint, and Anke Tröltzsch. An active-set trust-region method for derivative-free nonlinear bound-constrained optimization. Optimization Methods and Software, 26(4-5):873–894, 2011.

[16] Giampaolo Liuzzi, Stefano Lucidi, Francesco Rinaldi, and Luis Nunes Vicente. Trust-region methods for the derivative-free optimization of nonsmooth black-box functions. SIAM Journal on Optimization, 29(4):3012–3035, 2019.

[17] M. Kimiaei and A. Neumaier. Effective matrix adaptation strategy for noisy derivative-free optimization. Mathematical Programming Computation, 16:459–501, 2024.

[18] M. Kimiaei and A. Neumaier. MATRS: heuristic methods for noisy derivative-free bound-constrained mixed-integer optimization. Mathematical Programming Computation, 17:505–546, 2025.

[19] M. Kimiaei, A. Neumaier, and B. Azmi. LMBOPT: A limited memory method for bound-constrained optimization. Mathematical Programming Computation, 14:271–318, 2022.

[20] C. Cartis, J. Fowkes, and Z. Shao. Randomised subspace methods for non-convex optimization, with applications to nonlinear least-squares, 2022. Preprint.

[21] M. Kimiaei and A. Neumaier. A new limited memory method for unconstrained nonlinear least squares. Soft Computing, 26:465–490, 2022.

[22] M. Kimiaei, A. Neumaier, and P. Faramarzi. New subspace method for unconstrained derivative-free optimization. ACM Transactions on Mathematical Software, 49(4), 2023.

[23] C. Cartis and L. Roberts. Randomized subspace derivative-free optimization with quadratic models and second-order convergence. Optimization Methods and Software, pages 1–28, 2026.

[24] C. Cartis, E. Massart, and A. Otemissov. Bound-constrained global optimization of functions with low effective dimensionality using multiple random embeddings. Mathematical Programming, 198:997–1058, 2023.

[25] C. Cartis, E. Massart, and A. Otemissov. Global optimization using random embeddings. Mathematical Programming, 200:781–829, 2023.

[26] M. H. Raibert and J. J. Craig. Hybrid position/force control of manipulators. Journal of Dynamic Systems, Measurement, and Control, 103(2):126–133, 1981.

[27] N. Hogan. Impedance control: An approach to manipulation: Part I—theory. Journal of Dynamic Systems, Measurement, and Control, 107(1):1–7, 1985.

[28] T. Inoue, G. De Magistris, A. Munawar, T. Yokoya, and R. Tachibana. Deep reinforcement learning for high precision assembly tasks. In 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 819–825, 2017.

[29] C. C. Beltran-Hernandez, D. Petit, I. G. Ramirez-Alpizar, and K. Harada. Variable compliance control for robotic peg-in-hole assembly: A deep-reinforcement-learning approach. Applied Sciences, 10(19):6923, 2020.

[30] T. Johannink, S. Bahl, A. Nair, J. Luo, A. Kumar, M. Loskyll, J. A. Ojea, E. Solowjow, and S. Levine. Residual reinforcement learning for robot control. In 2019 International Conference on Robotics and Automation (ICRA), pages 6023–6029, 2019.

[31] X. Zhang, C. Wang, L. Sun, Z. Wu, X. Zhu, and M. Tomizuka. Efficient sim-to-real transfer of contact-rich manipulation skills with online admittance residual learning. In Proceedings of the 7th Conference on Robot Learning, volume 229 of Proceedings of Machine Learning Research, pages 1621–1639, 2023.

[32] B. Tang et al. IndustReal: Transferring contact-rich assembly tasks from simulation to reality. In Robotics: Science and Systems, 2023.

[33] M. Noseworthy et al. FORGE: Force-guided exploration for robust contact-rich manipulation under uncertainty. IEEE Robotics and Automation Letters, 10(5):4436–4443, 2025.

[34] X. Guo, X. Li, and R. Xu. Fast policy learning for linear-quadratic control with entropy regularization. SIAM Journal on Control and Optimization, 64(1):124–151, 2026.

[35] N. Hansen. The CMA evolution strategy: A tutorial, 2016. Revised 2023.

[36] H.-G. Beyer and B. Sendhoff. Simplify your covariance matrix adaptation evolution strategy. IEEE Transactions on Evolutionary Computation, 21(5):746–759, 2017.

[37] A. Jafari-Tabrizi, D. P. Gruber, and A. Gams. Exploiting image quality measure for automatic trajectory generation in robot-aided visual quality inspection. The International Journal of Advanced Manufacturing Technology, 132:4885–4901, 2024.

[38] H. Sago, R. Ariizumi, T. Asai, and S.-I. Azuma. Multi-objective path integral policy improvement for learning robotic motion. Artificial Life and Robotics, 30:534–545, 2025.

[39] D. Blanck-Kahan, G. Ortiz-Cervantes, V. Martínez-Gama, H. Cervantes-Culebro, J. E. Chong-Quero, and C. A. Cruz-Villar. Neural-optimal tuning of a controller for a parallel robot. Expert Systems with Applications, 236:121184, 2024.

[40] A. Rodríguez-Molina, M. G. Villarreal-Cervantes, J. S. Pantoja-García, A. G. Rojas-López, E. Hernández-Castillo, and R. Mejía-Rodríguez. Metaheuristic adaptive control based on polynomial regression and differential evolution for robotic manipulators. Applied Soft Computing, 151:111116, 2024.

[41] N. Hasan, H. Khan, S. Khatoon, and M. Sajid. Intelligent controller design of an autonomous system using a social spider optimizer for path navigation and obstacle avoidance. Automatika, 65(4):1432–1446, 2024.

[42] H. Huang, A. Arogbonlo, S. S. Yu, L. C. Kwek, and C. P. Lim. Adaptive neuro-fuzzy inference system based active force control with iterative learning for trajectory tracking of a biped robot. International Journal of Systems Science, 56(6):1171–1188, 2025.

[43] K. Chatzilygeroudis, R. Rama, R. Kaushik, D. Goepp, V. Vassiliades, and J.-B. Mouret. Black-box data-efficient policy search for robotics. In IEEE/RSJ International Conference on Intelligent Robots and Systems, 2017.

[44] Y. Lei, J. Chen, S. E. Li, and S. Zheng. Performance-driven controller tuning via derivative-free reinforcement learning. In 2022 IEEE 61st Conference on Decision and Control (CDC), pages 115–122, 2022.

[45] S. He, A. von Rohr, D. Baumann, J. Xiang, and S. Trimpe. Simulation-aided policy tuning for black-box robot learning. IEEE Transactions on Robotics, 41:2533–2548, 2025.

[46] R. Khalili Amirabadi and O. Solaymani Fard. Combining hybrid metaheuristic algorithms and reinforcement learning to improve the optimal control of nonlinear continuous-time systems with input constraints. Computers and Electrical Engineering, 116:109179, 2024.

[47] J. Chen, W. Ying, Z. Hua, B. Cao, and J. Zhao. Efficient impedance controller tuning of robot–environment interaction with improved transient and steady-state performance. IEEE Transactions on Instrumentation and Measurement, 73:7508310, 2024.

[48] M. Nobar, J. Keller, A. Rupenyan, M. Khosravi, and J. Lygeros. Guided bayesian optimization: Data-efficient controller tuning with digital twin. IEEE Transactions on Automation Science and Engineering, 22:11304–11317, 2025.

[49] A. Kudva, M. T. Huynh, A. Mesbah, and J. A. Paulson. Efficient performance-based mpc tuning in high dimensions using bayesian optimization over sparse subspaces. IFAC-PapersOnLine, 58(14):458–463, 2024.

[50] M. Zagórowska, C. König, H. Yu, E. C. Balta, A. Rupenyan, and J. Lygeros. Efficient safe learning for controller tuning with experimental validation. Engineering Applications of Artificial Intelligence, 143:109894, 2025.

[51] Y. Zhang and N. V. Sahinidis. Solving continuous and discrete nonlinear programs with BARON. Computational Optimization and Applications, 92(3):1123–1161, 2025.

[52] R. Storn and K. Price. Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces. Journal of Global Optimization, 11:341–359, 1997.

[53] E. D. Dolan and J. J. Moré. Benchmarking optimization software with performance profiles. Mathematical Programming, 91:201–213, 2002.