NEVO
A Neuromorphic EVolutionary Optimiser with Spike-Driven Cortico-Basal-Thalamic Coordination

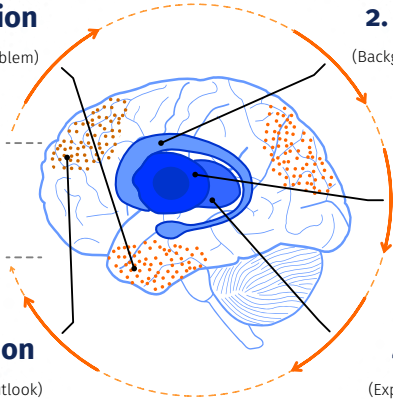
Jorge M. Cruz-Duarte • El-Ghazali Talbi

1. Perception

(Motivation & Problem)

2. Reasoning

(Background & Foundations)



3. Decision

(Proposed Approach)

4. Action

(Experiments & Results)

5. Reflection

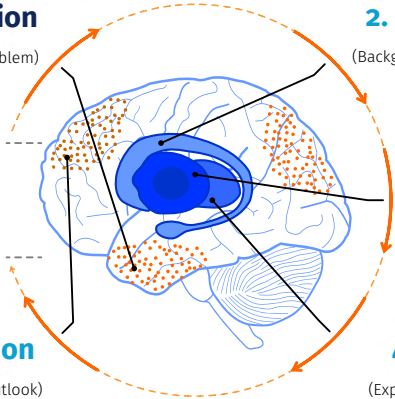
(Conclusions & Outlook)

1. Perception

(Motivation & Problem)

2. Reasoning

(Background & Foundations)



3. Decision

(Proposed Approach)

4. Action

(Experiments & Results)

5. Reflection

(Conclusions & Outlook)

Motivation I

Modern AI is consuming energy at a rate we cannot sustain!!!

- Energy and latency constraints increasingly limit where optimisation can be deployed,
- Embedded and autonomous systems are tightly budgeted.
- Separation between processing and memory makes data movement a dominant cost.
- Iterative BBO amplifies this cost via repeated evals. and pop. updates[1].

But this is the von Neumann realm!

[1] A. Eiben and J. Smith, "Introduction to evolutionary computing", Assembly Automation **24**, 324–324 (2004).

Motivation II

- Evolutionary Computation (EC) remains effective for problems where gradients are unavailable or unreliable[1].
- Its operators are local, its updates can be asynchronous, and its search tolerates finite precision.
- However, conventional implementations still rely on synchronous execution and repeated memory-access patterns that are not aligned with this structure.

[1] J. Liu et al., "Large-scale evolutionary optimization: a review and comparative study", Swarm and Evolutionary Computation **85**, 101466 (2024).

Motivation III

- Neuromorphic Computing (NC) provides event-driven architectures with collocated memory and computation.
- Platforms such as Intel Loihi, IBM TrueNorth, SpiNNaker, and BrainScaleS support sparse communication and distributed parallelism.
- While software frameworks such as Nengo implement the Neural Engineering Framework (NEF) for principled construction of spiking neural circuits[1].

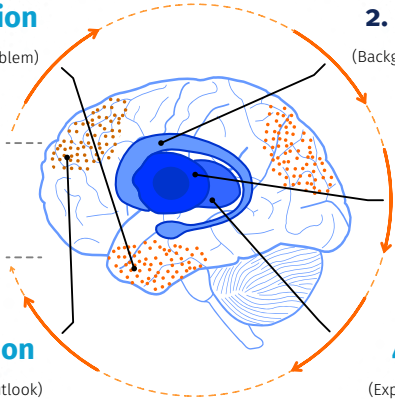
[1] C. Eliasmith and T. Stewart, "Nengo and NEF: Connecting Cognitive Theory to Neuroscience", in Proc. Annu. Meet. Cogn. Sci. Soc. Vol. 33 (2011).

1. Perception

(Motivation & Problem)

2. Reasoning

(Background & Foundations)



3. Decision

(Proposed Approach)

4. Action

(Experiments & Results)

5. Reflection

(Conclusions & Outlook)

Related Work I

Existing work supports neuromorphic optimisation, but it is commonly problem-bound or hybrid, leaving the optimisation loop outside the scope of spike-driven dynamics.

[1] H. Kim et al., "Organic memristor-based flexible neural networks with bio-realistic synaptic plasticity for complex combinatorial opt.", Adv. Sci. **10**, 2300659 (2023).

[2] A. Pierro et al., "Solving QUBO on the Loihi 2 Neuromorphic Processor", arXiv, 10.48550/arxiv.2408.03076 (2024).

[3] C. D. Schuman et al., "Evolutionary optimization for neuromorphic systems", in Proc. NICE Workshop (2020).

[4] D. Kudithipudi et al., "Neuromorphic computing at scale", Nature **637**, 801–812 (2025).

Related Work I

Existing work supports neuromorphic optimisation, but it is commonly problem-bound or hybrid, leaving the optimisation loop outside the scope of spike-driven dynamics.

First strand: Exploiting the device physics or hardwired setup.

- Implementations report promising efficiency in tackling combinatorial problems.
- E.g., ferroelectric/organic memristor nets[1] and SA on Loihi 2[2]

[1] H. Kim et al., "Organic memristor-based flexible neural networks with bio-realistic synaptic plasticity for complex combinatorial opt.", Adv. Sci. **10**, 2300659 (2023).

[2] A. Pierro et al., "Solving QUBO on the Loihi 2 Neuromorphic Processor", arXiv, 10.48550/arxiv.2408.03076 (2024).

[3] C. D. Schuman et al., "Evolutionary optimization for neuromorphic systems", in Proc. NICE Workshop (2020).

[4] D. Kudithipudi et al., "Neuromorphic computing at scale", Nature **637**, 801–812 (2025).

Related Work I

Existing work supports neuromorphic optimisation, but it is commonly problem-bound or hybrid, leaving the optimisation loop outside the scope of spike-driven dynamics.

First strand: Exploiting the device physics or hardwired setup.

- Implementations report promising efficiency in tackling combinatorial problems.
- E.g., ferroelectric/organic memristor nets[1] and SA on Loihi 2[2]
- But, extending them to other domains typically requires redesigning neural circuitry.

Second strand: Evolving spiking models, but retains a conventional search.

- EC algorithms can configure SNN architectures and learning rules[3,4] (**neuroevolution**).

[1] H. Kim et al., "Organic memristor-based flexible neural networks with bio-realistic synaptic plasticity for complex combinatorial opt.", Adv. Sci. **10**, 2300659 (2023).

[2] A. Pierro et al., "Solving QUBO on the Loihi 2 Neuromorphic Processor", arXiv, 10.48550/arxiv.2408.03076 (2024).

[3] C. D. Schuman et al., "Evolutionary optimization for neuromorphic systems", in Proc. NICE Workshop (2020).

[4] D. Kudithipudi et al., "Neuromorphic computing at scale", Nature **637**, 801–812 (2025).

Related Work I

Existing work supports neuromorphic optimisation, but it is commonly problem-bound or hybrid, leaving the optimisation loop outside the scope of spike-driven dynamics.

First strand: Exploiting the device physics or hardwired setup.

- Implementations report promising efficiency in tackling combinatorial problems.
- E.g., ferroelectric/organic memristor nets[1] and SA on Loihi 2[2]
- But, extending them to other domains typically requires redesigning neural circuitry.

Second strand: Evolving spiking models, but retains a conventional search.

- EC algorithms can configure SNN architectures and learning rules[3,4] (**neuroevolution**).
- Still, NC hardware is primarily used for inference or evaluation.

[1] H. Kim et al., "Organic memristor-based flexible neural networks with bio-realistic synaptic plasticity for complex combinatorial opt.", Adv. Sci. **10**, 2300659 (2023).

[2] A. Pierro et al., "Solving QUBO on the Loihi 2 Neuromorphic Processor", arXiv, 10.48550/arxiv.2408.03076 (2024).

[3] C. D. Schuman et al., "Evolutionary optimization for neuromorphic systems", in Proc. NICE Workshop (2020).

[4] D. Kudithipudi et al., "Neuromorphic computing at scale", Nature **637**, 801–812 (2025).

Related Work II

Neuromorphic ¿🤝? Optimisation

Three related directions:

[1] T. Sasaki and H. Nakano, "Swarm Intell. Alg. Spiking Neural-Osc. Netw., Coupling Interact. & Search Perform.", Nonlinear Theory Appl., IEICE **14**, 267–291 (2023).

[2] S. Snyder et al., "Neuromorphic bayesian optimization in Lava", in Proc. international conference on neuromorphic systems (icons) (2023).

[3] E. Talbi, *Neuromorphic-based metaheuristics: a new generation of low power, low latency and small footprint optimization algorithms*, 2025.

[4] J. M. Cruz-Duarte and E.-G. Talbi, "Neuroptimisation: the spiking way to evolve", arXiv, 10.48550/arxiv.2507.08320 (2025).

Related Work II

Neuromorphic ¿🤝? Optimisation

Three related directions:

- Swarm-inspired approaches provide spiking interpretations of established MHs[1].
  - Spiking dynamics for collective search, yet rarely exploiting all neuromorphic features.

[1] T. Sasaki and H. Nakano, "Swarm Intell. Alg. Spiking Neural-Osc. Netw., Coupling Interact. & Search Perform.", Nonlinear Theory Appl., IEICE **14**, 267–291 (2023).

[2] S. Snyder et al., "Neuromorphic bayesian optimization in Lava", in Proc. international conference on neuromorphic systems (icons) (2023).

[3] E. Talbi, *Neuromorphic-based metaheuristics: a new generation of low power, low latency and small footprint optimization algorithms*, 2025.

[4] J. M. Cruz-Duarte and E.-G. Talbi, "Neuroptimisation: the spiking way to evolve", arXiv, 10.48550/arxiv.2507.08320 (2025).

Related Work II

Neuromorphic ¿🤝? Optimisation

Three related directions:

- Swarm-inspired approaches provide spiking interpretations of established MHs[1].
  - Spiking dynamics for collective search, yet rarely exploiting all neuromorphic features.
- Probabilistic approaches are integrated into NC platforms through hybrid pipelines[2].
  - LavaBO exhibits scheduling and coordination, but the Bayesian core remains CPU-based.

[1] T. Sasaki and H. Nakano, "Swarm Intell. Alg. Spiking Neural-Osc. Netw., Coupling Interact. & Search Perform.", Nonlinear Theory Appl., IEICE **14**, 267–291 (2023).

[2] S. Snyder et al., "Neuromorphic bayesian optimization in Lava", in Proc. international conference on neuromorphic systems (icons) (2023).

[3] E. Talbi, *Neuromorphic-based metaheuristics: a new generation of low power, low latency and small footprint optimization algorithms*, 2025.

[4] J. M. Cruz-Duarte and E.-G. Talbi, "Neuroptimisation: the spiking way to evolve", arXiv, 10.48550/arxiv.2507.08320 (2025).

Related Work II

Neuromorphic ¿ 🤝 ? Optimisation

Three related directions:

- Swarm-inspired approaches provide spiking interpretations of established MHs[1].
  - Spiking dynamics for collective search, yet rarely exploiting all neuromorphic features.
- Probabilistic approaches are integrated into NC platforms through hybrid pipelines[2].
  - LavaBO exhibits scheduling and coordination, but the Bayesian core remains CPU-based.
- Recently, we formalised the design space of neuromorphic heuristic search.
  - Nheuristics[3] and NeurOptimiser[4] illustrate decentralised spike-triggered coordination.

[1] T. Sasaki and H. Nakano, "Swarm Intell. Alg. Spiking Neural-Osc. Netw., Coupling Interact. & Search Perform.", Nonlinear Theory Appl., IEICE **14**, 267–291 (2023).

[2] S. Snyder et al., "Neuromorphic bayesian optimization in Lava", in Proc. international conference on neuromorphic systems (icons) (2023).

[3] E. Talbi, *Neuromorphic-based metaheuristics: a new generation of low power, low latency and small footprint optimization algorithms*, 2025.

[4] J. M. Cruz-Duarte and E.-G. Talbi, "Neuroptimisation: the spiking way to evolve", arXiv, 10.48550/arxiv.2507.08320 (2025).

Related Work III

Hyphothesis

An evolutionary operator coordination can be realised as spike-driven action selection within a Cortico-Basal-Thalamic Loop (CBTL), while preserving a standard black-box interface for candidate generation and objective evaluation.

Related Work III

Hyphothesis

An evolutionary operator coordination can be realised as spike-driven action selection within a Cortico-Basal-Thalamic Loop (CBTL), while preserving a standard black-box interface for candidate generation and objective evaluation.

Objective

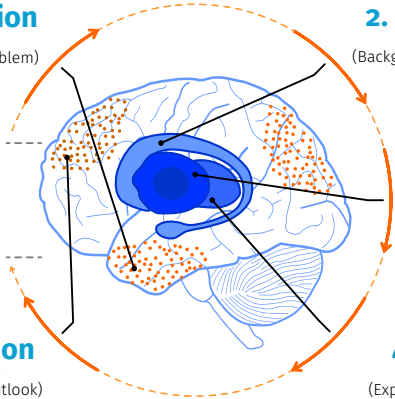
The objective is to confirm that spike-driven operator coordination is feasible with these interfaces and to identify the architectural and algorithmic constraints systematically. Such phenomena emerge when MH control is delegated to spiking dynamics.

1. Perception

(Motivation & Problem)

2. Reasoning

(Background & Foundations)



3. Decision

(Proposed Approach)

4. Action

(Experiments & Results)

5. Reflection

(Conclusions & Outlook)

Proposed Approach I

We present **Neuromorphic EVolutionary Optimiser (NEVO)**, which implements operator selection via spike-driven **Cortico-Basal-Thalamic Loop (CBTL)**.

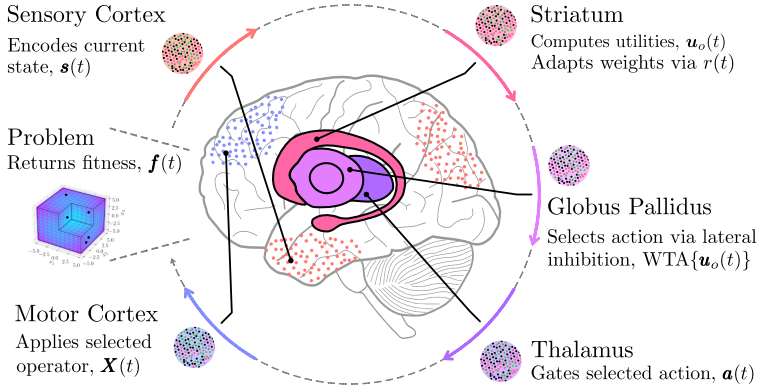


Figure: NEVO architecture with an operator selection based on a CBTL.

The **Sensory Cortex** encodes the state $s(t)$, the **Striatum** computes utilities $u_o(t)$, and the **Basal Ganglia** implement Winner-Take-All (WTA) to gate an approximately one-hot action vector $a(t)$ via the **Thalamus**.

The **Motor Cortex** triggers the selected operator to generate $X(t)$ and obtain fitness $f(t)$, and the resulting reward $r(t) \propto \Delta f(t)$ updates the memory and Striatum weights, closing the cycle.

Proposed Approach II

The controller follows the **Neural Engineering Framework (NEF)**[1] paradigm.

[1] C. Eliasmith and C. Anderson, *Neural engineering: computation, representation, and dynamics in neurobiological systems*, (MIT press, 2003)

Proposed Approach II

The controller follows the **Neural Engineering Framework (NEF)**[1] paradigm.

- It **encodes** continuous state variables in **spiking ensembles**.
- It **decodes** their activity into **control signals** for online operator selection.

[1] C. Eliasmith and C. Anderson, *Neural engineering: computation, representation, and dynamics in neurobiological systems*, (MIT press, 2003)

Proposed Approach II

The controller follows the **Neural Engineering Framework (NEF)**[1] paradigm.

- It **encodes** continuous state variables in **spiking ensembles**.
- It **decodes** their activity into **control signals** for online operator selection.

Why CBTL architecture?

[1] C. Eliasmith and C. Anderson, *Neural engineering: computation, representation, and dynamics in neurobiological systems*, (MIT press, 2003)

Proposed Approach II

The controller follows the **Neural Engineering Framework (NEF)**[1] paradigm.

- It **encodes** continuous state variables in **spiking ensembles**.
- It **decodes** their activity into **control signals** for online operator selection.

Why CBTL architecture?

- It provides a **biologically grounded WTA** mechanism.

[1] C. Eliasmith and C. Anderson, *Neural engineering: computation, representation, and dynamics in neurobiological systems*, (MIT press, 2003)

Proposed Approach II

The controller follows the **Neural Engineering Framework (NEF)**[1] paradigm.

- It **encodes** continuous state variables in **spiking ensembles**.
- It **decodes** their activity into **control signals** for online operator selection.

Why CBTL architecture?

- It provides a **biologically grounded WTA** mechanism.
- It operates in **continuous time** with distributed representations.

[1] C. Eliasmith and C. Anderson, *Neural engineering: computation, representation, and dynamics in neurobiological systems*, (MIT press, 2003)

Proposed Approach II

The controller follows the **Neural Engineering Framework (NEF)**[1] paradigm.

- It **encodes** continuous state variables in **spiking ensembles**.
- It **decodes** their activity into **control signals** for online operator selection.

Why CBTL architecture?

- It provides a **biologically grounded WTA** mechanism.
- It operates in **continuous time** with distributed representations.
- It supports **online adaptation** via **local synaptic** weight updates.

[1] C. Eliasmith and C. Anderson, *Neural engineering: computation, representation, and dynamics in neurobiological systems*, (MIT press, 2003)

Proposed Approach III: Algorithmic Architecture

- NEVO follows the **generalised MH** model[1] and adopts the low- and high-level view commonly used in AADCS.
- It can be seen as a state-based Adaptive Operator Section or HH controller[2,3].
- At the low level:
  - **Problem**: Find $\boldsymbol{x}_* = \text{argmin}_{\boldsymbol{x}\in\mathfrak{X}} f(\boldsymbol{x})$, where $\mathfrak{X} \subseteq \mathbb{R}^D$ and $f : \mathfrak{X} \to \mathbb{R}$
  - **Solver**: A sequence of search operators from a collection, $h_o \triangleq h_s \circ h_g \in \mathfrak{H}_o \subset \mathfrak{H}$
- At the high level:
  - **Problem**: Find $h^k_* = \text{argmax}_{h_o\in\mathfrak{H}_o\subset\mathfrak{H}} \{U(h_o, \boldsymbol{s}^k)\}$, where $\boldsymbol{s}^k$ is search state and U the expected utility
  - **Solver**: A sequence of ..., or NEVO!

[1] J. M. Cruz-Duarte et al., "Towards a generalised metaheuristic model for continuous optimisation problems", Mathematics **8**, 2046 (2020).

[2] J. Pei et al., "Adaptive operator selection for meta-heuristics: a survey", IEEE Transactions on Artificial Intelligence (2025).

[3] A. Hassan and N. Pillay, "Dynamic heuristic set selection for cross-domain selection hyper-heuristics", in Int. Conf. Theory Pract. Nat. Comput. (2021), pp. 33–44.

Proposed Approach III: Algorithmic Architecture

Domain Transformation

An affine transform $\mathcal{T} : \mathfrak{V} \to \mathfrak{X}$ with $\mathfrak{V} = [-1, 1]^D$ and $\mathfrak{X} = [\boldsymbol{l}_b, \boldsymbol{u}_b]$.

Memory Management

A fixed-size archive of capacity M storing pairs $(\boldsymbol{v}_i, f_i)$.

Utility Modelling

Each operator has utility $u_o : [0, 1]^3 \to \mathbb{R}$ and adaptive weight w_{h_o}. Thus, utilities are computed as $\boldsymbol{u}_o^k = \boldsymbol{W}(\boldsymbol{A}\boldsymbol{s}^{k+1} + \boldsymbol{b})$.

Search State

It is given by features (diversity, improv. rate, and conv.), as

$$\boldsymbol{s}^k = (\phi_d^k, \phi_i^k, \phi_c^k)^\intercal \in [0, 1]^3$$

where

$$\phi_d^k = \sqrt{3} D^{-1} \sum_{j=1}^{D} \mathrm{Std}\{\boldsymbol{v}_{:,j}^k\},$$

$$\phi_i^k = W^{-1} \sum_{q=k-W+1}^{k} \mathbb{I}(f_\star^q < f_\star^{q-1}),$$

$$\phi_c^k = \left(1 + \lambda_c \Delta f^k / (\,|f_\star^k| + \eta_c\, \Delta f^k + \epsilon)\right).$$

Proposed Approach IV: Algorithmic Architecture

Search Operator Collection

Thirteen search operators (each generates N candidate solutions) from well-known MHs a priori categorised* as for:

Exploration

- Random Search (RS)
- Lévy Flight (LF)
- Genetic Crossover (GX)
- Differential Evolution (DE)
- Central Force Optimisation (CF)
- Firefly Algorithm (FA)
- Gravitational Search (GS)

Exploitation

- Local Random Walk (LW)
- Genetic Mutation (GM)
- Simulated Annealing (SA)
- Tabu Search (TS)
- Particle Swarm Optimisation (PS)
- Spiral Optimisation (SO)

*This is a coarse description of the intended role of each implementation and does not imply a strict dichotomy.

Proposed Approach V: CBTL

The operator selection is implemented as a **Cortico-Basal-Thalamic Loop (CBTL)**[1] circuit.

- Utility ensembles (one per operator) compute $u_o(\boldsymbol{s}(t))$
- Basal ganglia WTA network performs competition
- Thalamus gates action vector $\boldsymbol{a}(t) \in \mathbb{R}^{13}$ for operator selection
- After each evaluation, reward $\boldsymbol{r}(t)$ updates weights and the action selection applies ε-greedy policy over basal ganglia output[2]
- The closed loop

$$\boldsymbol{s}(t) \to \boldsymbol{u}_o(t) \to \boldsymbol{a}(t) \to \boldsymbol{X}(t) \to r(t)$$

updates every Δt_{eval}.

[1] D. Joseph and R. S. Bapi, "What do the basal ganglia do? A modeling perspective", Biological cybernetics **103**, 237–253 (2010).

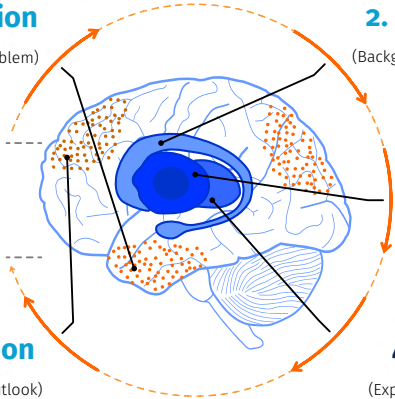
[2] R. S. Sutton and A. G. Barto, *Reinforcement learning: an introduction*, (A Bradford Book, Cambridge, MA, USA, 2018).

1. Perception

(Motivation & Problem)

2. Reasoning

(Background & Foundations)



3. Decision

(Proposed Approach)

4. Action

(Experiments & Results)

5. Reflection

(Conclusions & Outlook)

Experimental Setup

NEVO implementation:

- Nengo[1] (in Python v3.10) with LIF neurons
- Population Size of $N = 50$, Memory Size of $M = 25$,
- Ensemble Size of $N_{\text{neurons}} = 50$, and Simulation Step of $\Delta t = 1\,\text{ms}$

[1] C. Eliasmith and T. Stewart, "Nengo and NEF: Connecting Cognitive Theory to Neuroscience", in Proc. Annu. Meet. Cogn. Sci. Soc. Vol. 33 (2011).

[2] N. Hansen et al., "COCO: a platform for comparing continuous optimizers in a black-box setting", Optimization Methods and Software **36**, 114–144 (2021).

[3] J. de Nobel et al., "IOHexperimenter: Benchmarking Platform for Iterative Optimization Heuristics", arXiv (2021).

Experimental Setup

NEVO implementation:

- Nengo[1] (in Python v3.10) with LIF neurons
- Population Size of $N = 50$, Memory Size of $M = 25$,
- Ensemble Size of $N_{\text{neurons}} = 50$, and Simulation Step of $\Delta t = 1$ ms

Benchmarking:

- BBOB noiseless suite from COCO[2] and and IOHexperimenter[3]
- 24 problems across six dimensions (2, 3, 5, 10, 20, 40) with 15 instances each
- Five categories: Separable (`separ`), Unimodal Low- and High-Conditioning (`lcond` & `hcond`), and Multimodal Adequate and Weak Structure (`multi` & `mult2`)

[1] C. Eliasmith and T. Stewart, "Nengo and NEF: Connecting Cognitive Theory to Neuroscience", in Proc. Annu. Meet. Cogn. Sci. Soc. Vol. 33 (2011).

[2] N. Hansen et al., "COCO: a platform for comparing continuous optimizers in a black-box setting", Optimization Methods and Software **36**, 114–144 (2021).

[3] J. de Nobel et al., "IOHexperimenter: Benchmarking Platform for Iterative Optimization Heuristics", arXiv (2021).

Experimental Protocol

Preliminary Experiment

- Simulation time: 5 s
- Functions: f1, f6, f10, f15, f20
- Instances: 1st
- Dimensions: 10

[1] AMD-Penguin Computing with AMD EPYC 7642 (48 cores, 512 GB RAM), part of the Grid'5000 project: `https://www.grid5000.fr`.

Experimental Protocol

Preliminary Experiment

- Simulation time: 5 s
- Functions: f1, f6, f10, f15, f20
- Instances: 1st
- Dimensions: 10

Confirmatory Experiment

- Simulation time: 20 s
- Functions: f1–f24
- Instances: 1–15
- Dimensions: 2, 3, 5, 10, 20, 40

[1] AMD-Penguin Computing with AMD EPYC 7642 (48 cores, 512 GB RAM), part of the Grid'5000 project: `https://www.grid5000.fr`.

Experimental Protocol

Preliminary Experiment

- Simulation time: 5 s
- Functions: f1, f6, f10, f15, f20
- Instances: 1st
- Dimensions: 10

Confirmatory Experiment

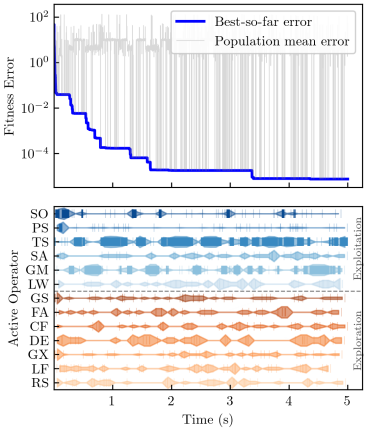
- Simulation time: 20 s
- Functions: f1–f24
- Instances: 1–15
- Dimensions: 2, 3, 5, 10, 20, 40

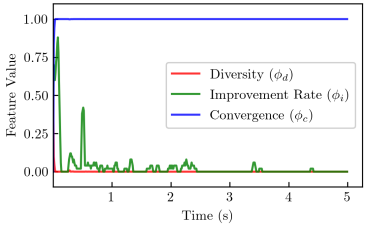
Platform: A machine with 48 cores and 512 GB RAM part of the Grid'5000[1]

[1] AMD-Penguin Computing with AMD EPYC 7642 (48 cores, 512 GB RAM), part of the Grid'5000 project: `https://www.grid5000.fr`.

Preliminary Experiment I

Sphere (fo1) in 10D: Best: 79.5, Error: 7.57×10^{-6}





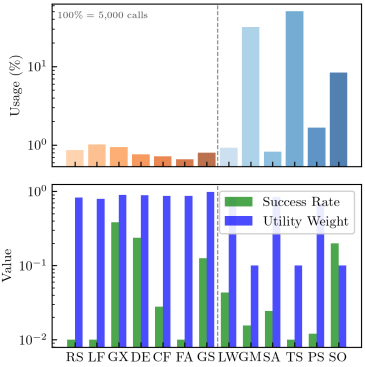
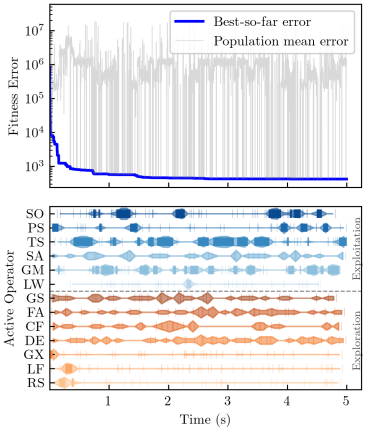


Figure: Representative single-run optimisation dynamics of the CBTL arch. on **Sphere (fo1)** in 10D.

Left: Fitness Error / Active Operator
Centre: Search-State Features
Right: Operator Usage / Success Rates & Weights

Preliminary Experiment II

Ellipsoidal Rotated (f10) in 10D: Best: 365, Error: 420



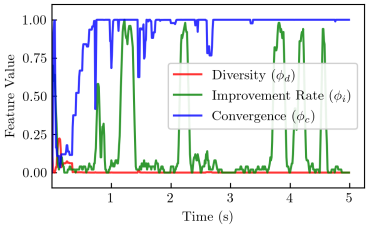
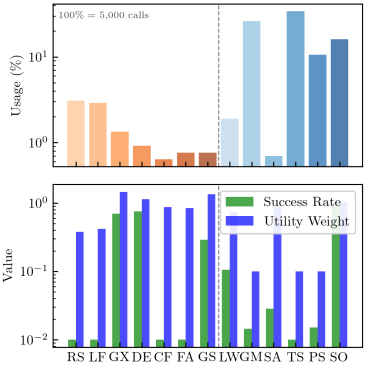


Figure: Representative single-run optimisation dynamics of the CBTL arch. on **Ellipsoidal Rotated (f10)** in 10D.

Left: Fitness Error / Active Operator
Centre: Search-State Features
Right: Operator Usage / Success Rates & Weights



Confirmatory Experiment I

Fraction of function-target pairs for 2D, 3D, 5D

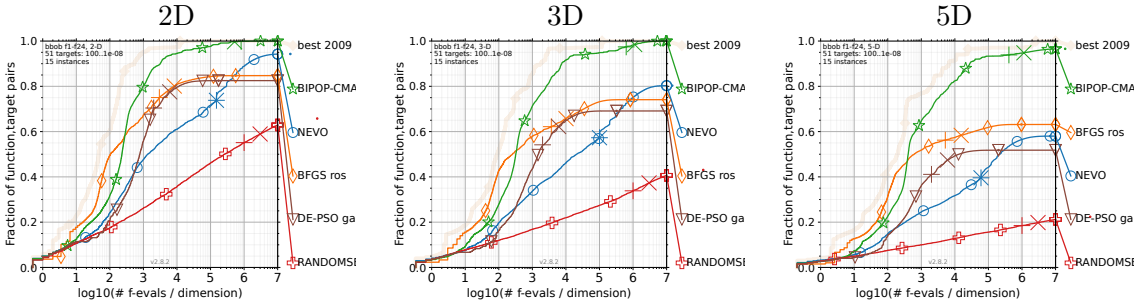


Figure: Bootstrapped Empirical Cumulative Distribution Functions (ECDFs) of the number of f-evaluations divided by dimension for 51 targets in $10^{[-8..2]}$ across all function subgroups, with 15 instances per dimension, shown for dimensions 2–40. The best 2009 (upper bound ECDFs) is shown as a light, thick reference line.

Legend: ◯: NEVO (this work), ☆: BIPOP-CMA-ES, ◇: BFGS, ▽: DE-PSO, ✚: RANDOMSEARCH. Comparison datasets were obtained from COCO v2.8.2.

Confirmatory Experiment II

Fraction of function-target pairs for 10D, 20D, 40D

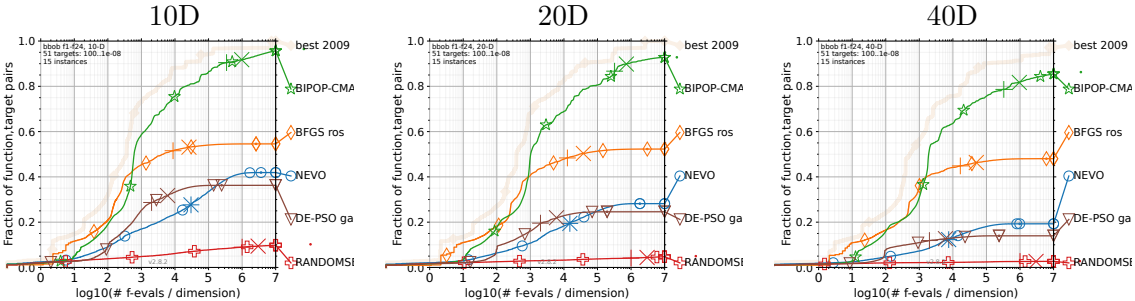


Figure: Bootstrapped Empirical Cumulative Distribution Functions (ECDFs) of the number of f-evaluations divided by dimension for 51 targets in $10^{[-8..2]}$ across all function subgroups, with 15 instances per dimension, shown for dimensions 2–40. The best 2009 (upper bound ECDFs) is shown as a light, thick reference line.

Legend: ◯: NEVO (this work), ☆: BIPOP-CMA-ES, ◇: BFGS, ▽: DE-PSO, ✚: RANDOMSEARCH. Comparison datasets were obtained from COCO v2.8.2.

Confirmatory Experiment III

Mean operator weight vs mean success rate

- Partial alignments between weights & improv.
- □ gets higher weights than ◯
- Current rule overvalues sporadic large improv. and undervalues consistent incremental progress.
- It remains insensitive to diversity and delayed benefits.

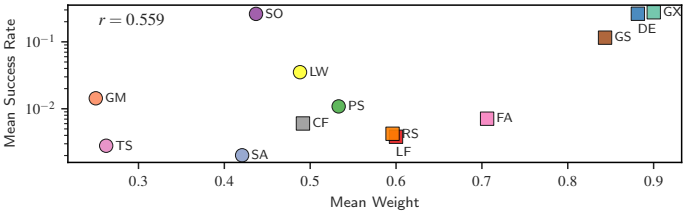
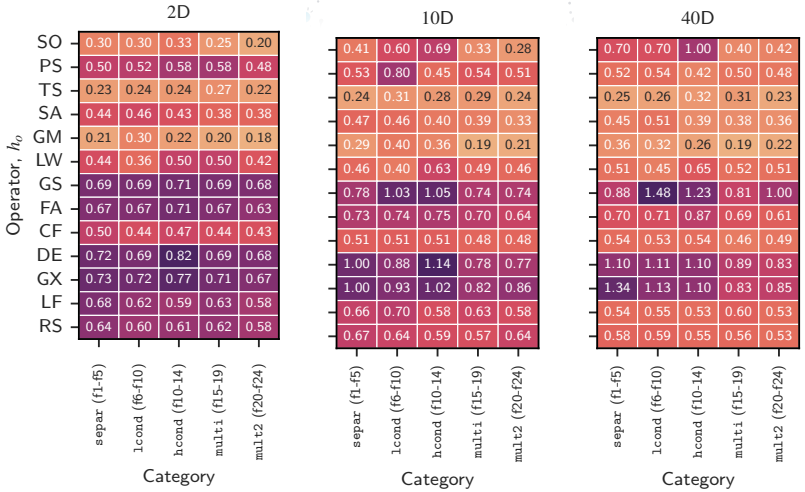


Figure: Relationship between mean operator weight and mean success rate. Each marker represents a search operator.
r corresponds to the Pearson correlation coefficient.

Classification (a priori): Exploration (□) and Exploitation (◯)

Confirmatory Experiment IV



Operator weights across dims & BBOB categories

Figure: Operator weight heatmaps faceted by problem dimension, showing how operator preferences vary across BBOB problem categories.

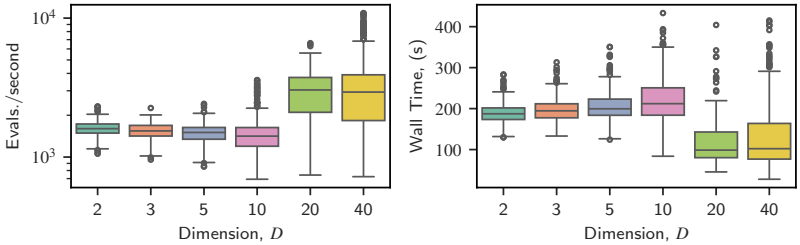
Classification (a priori): Exploration (RS–GS) and Exploitation (LW–SO)

Confirmatory Experiment V

Evaluation throughput and wall-clock time per experiment

Figure: Performance summary by problem dim.

Left: Number of evaluations per second
Right: Wall-clock time required per exp.

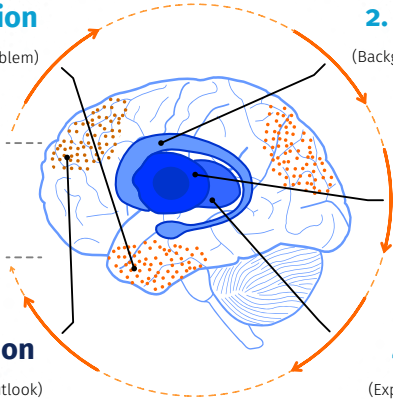


1. Perception

(Motivation & Problem)

2. Reasoning

(Background & Foundations)



3. Decision

(Proposed Approach)

4. Action

(Experiments & Results)

5. Reflection

(Conclusions & Outlook)

Conclusions

- We introduced an NC-based evolutionary optimiser in which operator selection is implemented via spike-driven coordination within a CBTL, realised via the NEF paradigm.
- Experiments revealed that spike-driven action selection can sustain coherent operator sequences and achieve strong to acceptable target attainment in 2, 3, and 5D.
- Performance deteriorates from 10D forth under the fixed-resource setting.
- The weight statistics indicate a systematic bias toward exploration operators, driven by a reward function that mainly responds to immediate improvements in best fitness.

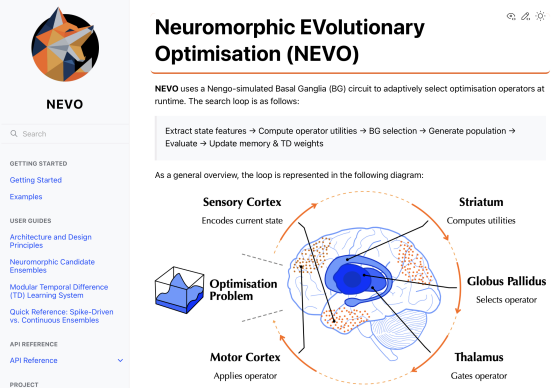
Future Work

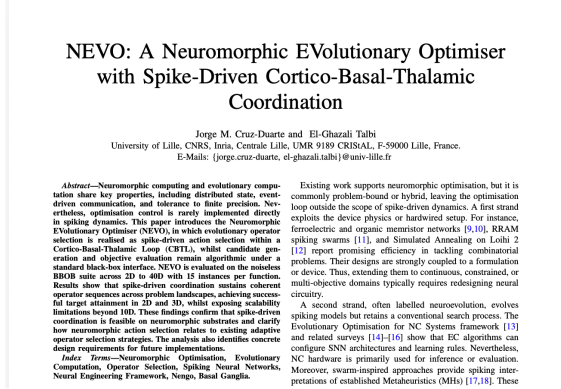
- Address scalability by revising the **reward** and **weight** updates to account for incremental progress, population-level improvements, and diversity maintenance.
- Make the **utility mapping** be learned online rather than fixed.
- Extend the **state encoding** to capture landscape anisotropy and rotation sensitivity without increasing decoding burden.
- Reduce **hybridity** by developing NC representations for solution memory, candidate generation, and selection; e.g., mapping these to deployable substrates.
- Make **evaluation protocols** jointly report solution quality, latency, and energy, and include **hardware-backed measurements**.

Resources

https://jcrvz.github.io/nevo

cec_pap596





NEVO: A Neuromorphic EVolutionary Optimiser with Spike-Driven Cortico-Basal-Thalamic Coordination

Jorge M. Cruz-Duarte and El-Ghazali Talbi
University of Lille, CNRS, Inria, Centrale Lille, UMR 9189 CRIStAL, F-59000 Lille, France.
E-Mails: {jorge.cruz-duarte, el-ghazali.talbi}@univ-lille.fr

***Abstract*—Neuromorphic computing and evolutionary computation share key properties, including distributed state, event-driven communication, and tolerance to finite precision. Nevertheless, optimisation control is rarely implemented directly in spiking dynamics. This paper introduces the Neuromorphic EVolutionary Optimiser (NEVO), in which evolutionary operator selection is realised as spike-driven action selection within a Cortico-Basal-Thalamic Loop (CBTL), whilst candidate generation and objective evaluation remain algorithmic under a standard black-box interface. NEVO is evaluated on the noiseless BBOB suite across 2D to 40D with 15 instances per function. Results show that spike-driven coordination sustains coherent operator sequences across problem landscapes, achieving successful target attainment in 2D and 3D, whilst exposing scalability limitations beyond 10D. These findings confirm that spike-driven coordination is feasible on neuromorphic substrates and clarify how neuromorphic action selection relates to existing adaptive operator selection strategies. The analysis also identifies concrete design requirements for future implementations.**

***Index Terms*—Neuromorphic Optimisation, Evolutionary Computation, Operator Selection, Spiking Neural Networks, Neural Engineering Framework, Nengo, Basal Ganglia.**

Existing work supports neuromorphic optimisation, but it is commonly problem-bound or hybrid, leaving the optimisation loop outside the scope of spike-driven dynamics. A first strand exploits the device physics or hardwired setup. For instance, ferroelectric and organic memristor networks [9,10], RRAM spiking swarms [11], and Simulated Annealing on Loihi 2 [12] report promising efficiency in tackling combinatorial problems. Their designs are strongly coupled to a formulation or device. Thus, extending them to continuous, constrained, or multi-objective domains typically requires redesigning neural circuitry.

A second strand, often labelled neuroevolution, evolves spiking models but retains a conventional search process. The Evolutionary Optimisation for NC Systems framework [13] and related surveys [14]–[16] show that EC algorithms can configure SNN architectures and learning rules. Nevertheless, NC hardware is primarily used for inference or evaluation. Moreover, swarm-inspired approaches provide spiking interpretations of established Metaheuristics (MHs) [17,18]. These

The convergence of Evolutionary Computation and Neuromorphic Computing is timely, technically plausible, and open for us, as a community, to shape.

Bedankt!
Thanks!
Merci!
Grazie!
¡Gracias!
Obrigado!
Mulţmesc!

NeurOptim

neuroptim.org

Contact information

Jorge M. Cruz-Duarte
jorge.cruz-duarte@univ-lille.fr

El-Ghazali Talbi
el-ghazali.talbi@univ-lille.fr

France 2030 · Liberté Égalité Fraternité · RÉPUBLIQUE FRANÇAISE · Inria · Université de Lille · cnrs · CRIStAL Centre de Recherche en Informatique, Signal et Automatique de Lille

(This slide is intentionally left blank.)

Complexity Analysis

Total per-cycle cost:

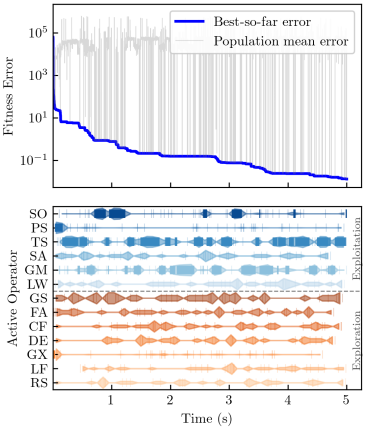
$$\mathcal{O}\Big(\underbrace{NC_f}_{\text{Obj. Eval.}} + \underbrace{C_{\mathrm{o}}(N, D, M)}_{\text{Search Operator}} + \underbrace{MD}_{\text{State Feat.}} + \underbrace{NM}_{\text{Mem. Update}} + \underbrace{K}_{\text{Utility}} + \underbrace{(E + C)\,\Delta t_{\mathrm{eval}}/\Delta t}_{\text{Neuromorphic Controller}} \Big)$$

$$C_{\mathrm{o}}(N, D, M) = \begin{cases} \mathcal{O}(ND), & \text{Simple perturbation} \\ \mathcal{O}(NM \log M), & \text{Parent selection with sorting} \\ \mathcal{O}(N^2 D), & \text{Distance-based perturbation} \\ \mathcal{O}(ND^2), & \text{Coordinate plane rotations} \end{cases}$$

- It can be reduced to $\mathcal{O}(NC_f)$ if C_f dominates
- $\mathcal{O}((E + C)\,\Delta t_{\mathrm{eval}}/\Delta t)$ is a fixed overhead independent of D
  - It can become significant in software-based implementations.
  - It would become substantially reduced on dedicated NC hardware.

Preliminary Experiment III

Attactive Sector (f06) in 10D: Best: 37.6, Error: 1.72



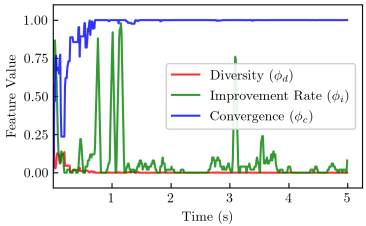
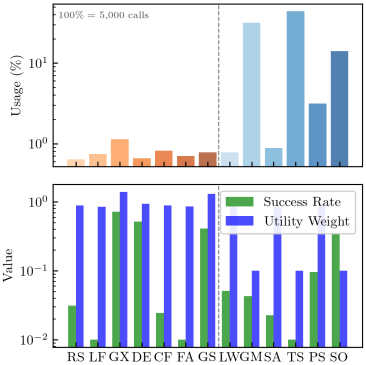


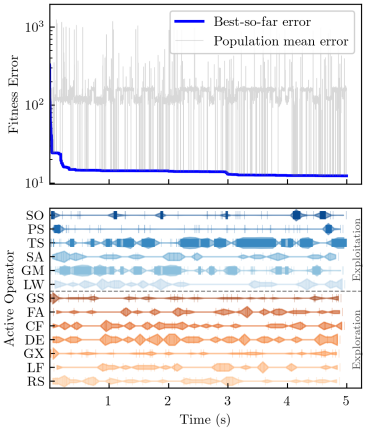
Figure: Representative single-run optimisation dynamics of the CBTL arch. on **Attactive Sector (f06)** in 10D.

Left: Fitness Error / Active Operator
Centre: Search-State Features
Right: Operator Usage / Success Rates & Weights



Preliminary Experiment IV

Rastrigin Rotated (f15) in 10D: Best: 1060, Error: 63.5



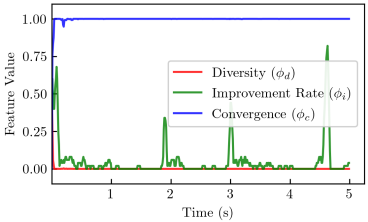
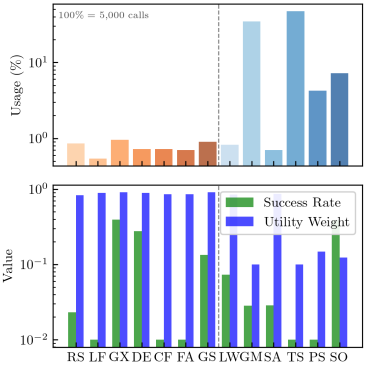


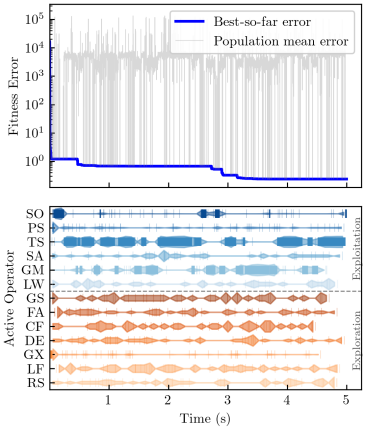
Figure: Representative single-run optimisation dynamics of the CBTL arch. on **Rastrigin Rotated (f15)** in 10D.

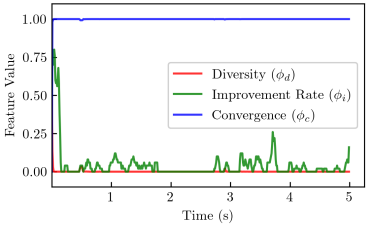
Left: Fitness Error / Active Operator
Centre: Search-State Features
Right: Operator Usage / Success Rates & Weights



Preliminary Experiment V

Schwefel (f20) in 10D: Best: -545, Error: 1.73





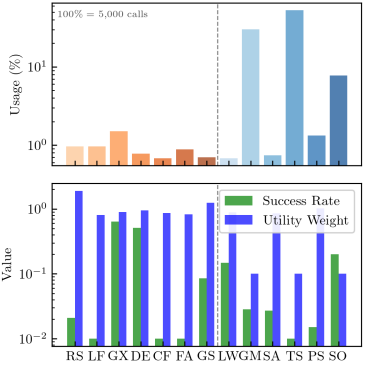


Figure: Representative single-run optimisation dynamics of the CBTL arch. on **Schwefel (f20)** in 10D.

Left: Fitness Error / Active Operator
Centre: Search-State Features
Right: Operator Usage / Success Rates & Weights

Confirmatory Experiment IV

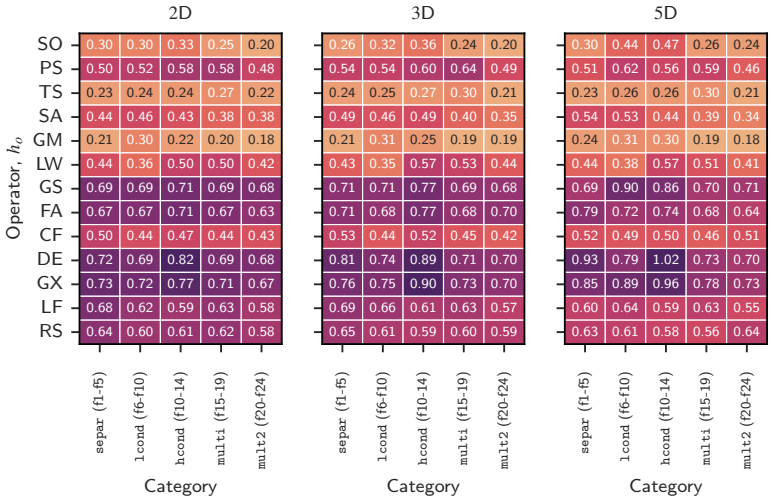


Figure: Operator weight heatmaps faceted by problem dimension, showing how operator preferences vary across BBOB problem categories.

Classification (a priori): Exploration (RS–GS) and Exploitation (LW–SO)

Confirmatory Experiment V

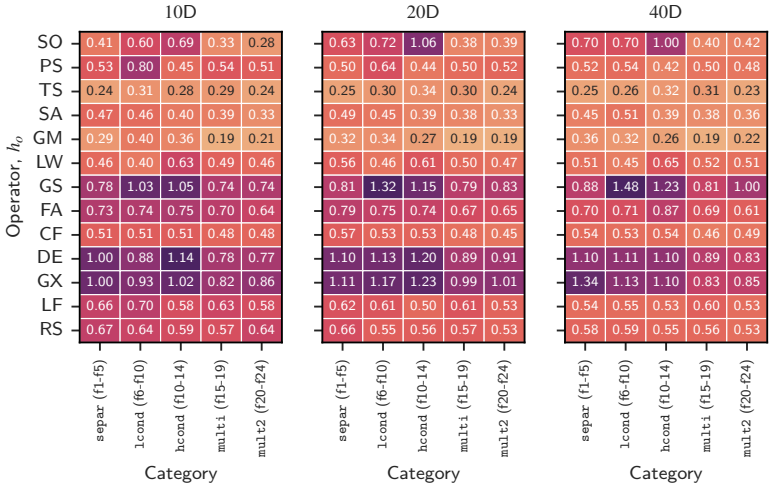


Figure: Operator weight heatmaps faceted by problem dimension, showing how operator preferences vary across BBOB problem categories.

Classification (a priori): Exploration (RS–GS) and Exploitation (LW–SO)