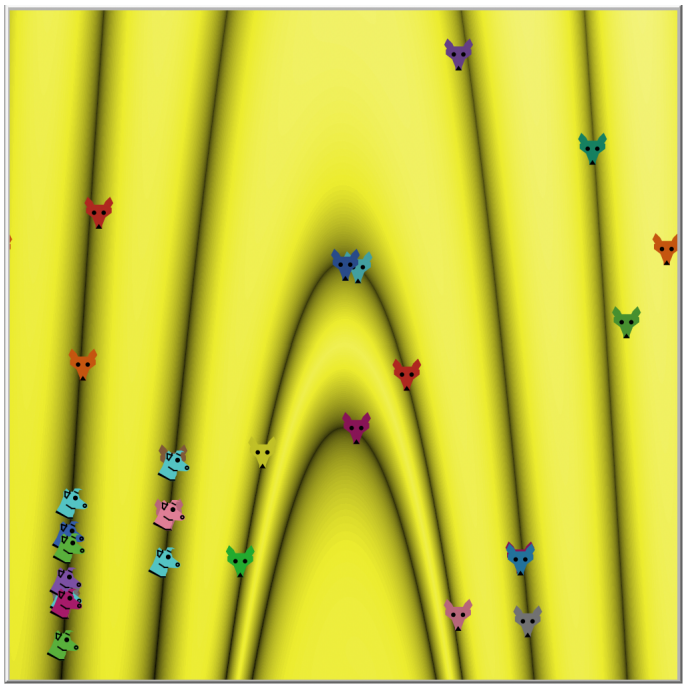
CIAO Optimization Tool
User Guide

Sergio Rojas-Galeano, Martha Garzón, Lindsay Álvarez

February 16, 2024

WHAT IS IT?

The CIAO (Collective Intelligence Algorithm for Optimization) metaheuristic simulates a collective intelligence approach to solving unconstrained continuous optimization problems. It involves agents known as solvers (wolves) and users (dogs), navigating a solution space to find the optimal coordinates that minimize a cost function associated with an optimization problem. This tool implements the algorithm as an agent-based model using the Netlogo language.



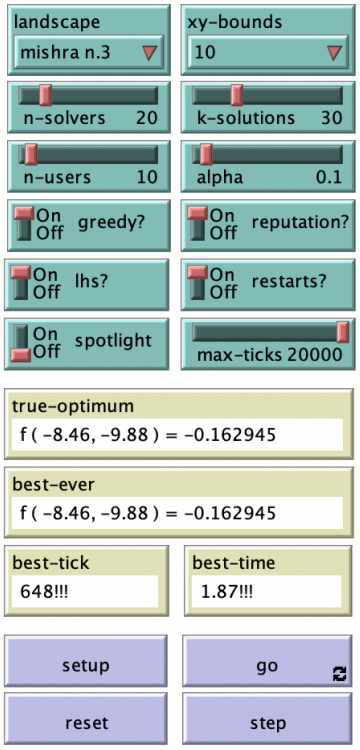
HOW IT WORKS

Solver agents maintain knowledge about promising sub-regions in the search space, represented as Gaussian distributions, involving their core expertise and their expertise dispersion. Users seek solutions from solvers, and the model incorporates learning and reputation mechanisms to refine the solver's expertise and reward effective solutions.

HOW TO USE IT

First configure the simulation parameters:

- LANDSCAPE: Chooses the optimization problem, visually represented in the view or world area. Selection influences the XY-BOUNDS. Refer to the APPENDIX: LIST OF BENCHMARK PROBLEMS section for a description of available benchmark functions.
- XY-BOUNDS: Sets the lower and upper bounds of the search space depending on the chosen landscape.
- N-SOLVERS: Defines the number of solver agents.



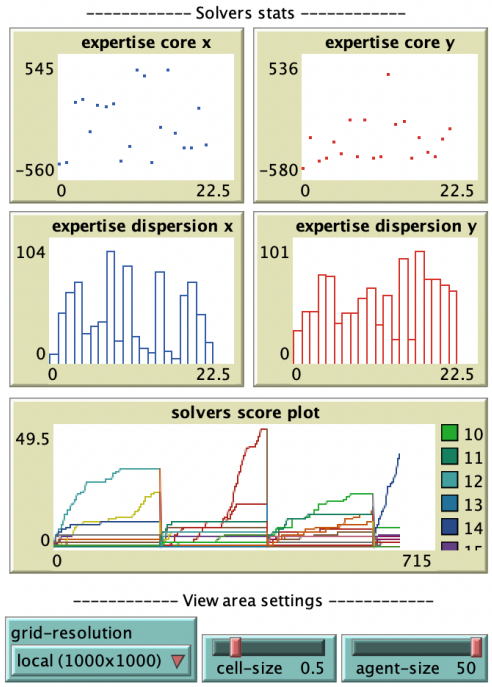
- K-SOLUTIONS: Defines the number of solutions a chosen solver attempts to generate.
- N-USERS: Defines the number of user agents.
- ALPHA: Sets the learning rate for solver adaptation.
- GREEDY?: Enables or disables the greedy update mode of solver expertise.
- REPUTATION?: Enables or disables choosing solvers based on their scores using roulette wheel selection.
- LHS?: Enables or disables Latin Hypercube Sampling of the initial solver population.
- RESTARTS?: Enables or disables random resets to prevent stagnation due to premature convergence to local minima.
- SPOTLIGHT?: Enables or disables highlighting the global minima in the view or world area.
- MAX-TICKS: Sets the maximum number of iterations of the algorithm main search routine.
- GRID-SIZE: Adjusts the resolution of the view area. Choose "*web (200x200)*" if running on the model online in the modeling commons website, as server memory constraints limit the amount of cells in the search space. Choose "*local (1000x1000)*" for a higher resolution of 1000x1000 cells if running on a desktop machine, allowing for better discretization of the search space.
- CELL-SIZE: Specifies the size of each grid cell in the view area. Can be adjusted from 0.1 to 2 with a step increment of 0.1. This control enables closer or further inspection of the cells in the view area.
- AGENT-SIZE: Controls the size of agent representations in the view area. Adjusts from 10 to 50 with a step increment of 10.

Next click the SETUP button to initialize the model with chosen parameters. And then click the GO button to start the simulation. Observe the movement and interaction of solvers and users in the view area of the simulator. You can control the execution of the simulation using the control panel buttons:

- SETUP: Computes and visualizes the landscape and initializes agents and global variables of the simulation, according to the given parameters.
- RESET: Only initializes agents and global variables of the simulation, according to the given parameters. Also clears plots from previous runs.
- GO: Executes the main search routine until stopping conditions are met.
- STEP: Executes one iteration of the main search routine.

THINGS TO NOTICE

- Observe how solvers adapt their expertise to the search space, guiding users towards promising regions.
- Observe how regions with low (black) and high (yellow) values in the landscape are visualized in the view area based on the selected problem (LANDSCAPE).
- Watch the TRUE-OPTIMUM value and notice how the BEST-EVER approaches to it.
- Monitor the BEST-EVER patch, BEST-TICK, and BEST-TIME to understand when and where the best solution is discovered.
- Observe how the EXPERTISE CORE and EXPERTISE DISPERSION parameters of solvers evolve over time in the corresponding plots.
- Analyze the SOLVERS SCORE plot to see how solvers' reputation change during the optimization process.
- Notice the periodic changes in the SOLVERS SCORE plot when RESTARTS? is enabled.



THINGS TO TRY

- Experiment with different numbers of SOLVERS and USERS to observe how the collective intelligence adapts to problem complexity.
- Observe how adjusting the learning rate (ALPHA) affects the adaptation of solver expertise. Higher values (closer to 1) make solvers more resistant to exploring new solutions and cling to their currently known best solution. Lower values make them more susceptible to learning from new information and exploring alternative solutions.
- Evaluate the impact of greedy (utilizing the single best new solution) and non-greedy (leveraging the average performance of multiple new solutions) learning strategies on solver adaptation using the GREEDY? switch.
- Explore the effects on solvers scores and on user decisions, of enabling or disabling reputation-based solver selection (REPUTATION?).
- Test the impact of Latin Hypercube Sampling of initial solver locations (LHS?).
- Observe the behavior when random restarts are enabled or disabled (RESTARTS?).
- Toggle the spotlight (SPOTLIGHT?) to visually track the global minimum in the landscape, and how user agents approach to it.

EXTENDING THE MODEL

Here are some ideas to extend the model:

- Extend the list of landscape functions of optimization problems.
- Implement additional user or solver behaviors to enhance the complexity of the collective intelligence dynamics. Techniques such as temporal memory, tabu lists, collaboration mechanisms, or more advanced expertise representation models like a mixture of Gaussians.
- Extend the model to incorporate alternative solver selection strategies to compare their impact on the optimization process.
- Generalize the model to handle continuous optimization problems in more than two dimensions ($d > 2$).
- Investigate the adaptation of the model for binary domain problems, exploring how the dynamics change in this context.