Engineering Applications of Artificial Intelligence 100 (2021) 104170

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai

Coral reefs optimization algorithms for agent-based model calibration

Ignacio Moya [a,*], Enrique Bermejo [a], Manuel Chica [a,b], Óscar Cordón [a]

[a] Andalusian Research Institute DaSCI "Data Science and Computational Intelligence", University of Granada, 18071 Granada, Spain
[b] School of Electrical Engineering and Computing, The University of Newcastle, Callaghan NSW 2308, Australia

ARTICLE INFO

Keywords:
Evolutionary computation
Metaheuristics
Coral reefs optimization
Model calibration
Agent-based modeling

ABSTRACT

Calibrating agent-based models involves estimating multiple parameter values. This can be performed automatically using automatic calibration but its success depends on the optimization method's ability for exploring the parameter search space. This paper proposes to carry out this process using coral reefs optimization algorithms, a new branch of competitive bio-inspired metaheuristics that, beyond its novel metaphor, has shown its good behavior in other optimization problems. The performance of these metaheuristics for model calibration is evaluated by conducting an exhaustive experimentation against well-established and recent evolutionary algorithms, including their hybridization with local search procedures. The study analyzes the calibration accuracy of the metaheuristics using an integer coding scheme over a benchmark of 12 problem instances of an agent-based model with an increasing number of decision variables. The outstanding performance of the memetic coral reefs optimization is reported after performing statistical tests to the results.

1. Introduction

Agent-based modeling (ABM) (Epstein, 2006; Janssen and Ostrom, 2006; Wilensky and Rand, 2015) is a well-established methodology for designing and simulating computational models that relies on autonomous entities called agents. The behavior of these artificial agents is managed through the definition of simple rules and their interactions with other agents. The ability of ABM for recreating complex and emerging dynamics through the aggregation of the agents' rules and social interactions has made them receive increased attention in the last few years (Farmer and Foley, 2009; Waldrop, 2018; Coates et al., 2019). However, building agent-based models is difficult because the values of a large number of parameters must be set in order to design the model. In addition, the modeler is commonly forced to estimate many of those parameter values due to lack of appropriate data. The process of adjusting these parameter values to correctly replicate the desired dynamics is addressed as model calibration (Chica et al., 2017; Oliva, 2003).

Model calibration can be performed automatically using automatic calibration, a computationally intensive process that adjusts the model's parameters using a optimization method. This optimization method considers an error measure for comparing the model's output and the real data from the phenomena simulated by the model (Oliva, 2003; Sargent, 2005). Since the values of different model parameters are typically unrelated, it is desirable to use non-linear optimization methods such as metaheuristics (Talbi, 2009) that can search through the whole parameter space (Chica et al., 2017; Stonedahl and Rand, 2014). Nevertheless, the selection of the metaheuristic for carrying out the calibration process heavily determines the quality of the resulting model parameter configuration as the model's accuracy relies on the ability of the method for exploring the parameter search space.

The current manuscripts proposes to carry out the calibration process using novel and competitive bio-inspired metaheuristics based on coral reefs: coral reefs optimization (CRO) (Salcedo-Sanz et al., 2014) and coral reefs optimization with substrate layers (CRO-SL) (Salcedo-Sanz et al., 2016b). Coral reefs-based metaheuristics emulate the formation and reproduction of coral reefs, resulting in an optimization algorithm with a powerful trade-off between exploration and exploitation of the search space. CRO-SL is an enhanced version of CRO that also includes a cooperative co-evolution scheme and has shown outstanding performance tackling some complex optimization problems, such as (Bermejo et al., 2018; Camacho-Gómez et al., 2019; Garcia-Hernandez et al., 2020a,b; Salcedo-Sanz et al., 2019). Taking these good results as a base, the performance of the coral reefs-based metaheuristics should be extensively analyzed in the significantly complex problem of calibrating ABM. The authors acknowledge that the recent surge of novel bio-inspired algorithms has been subject of controversy due to the lack of scientific rigor behind some of these algorithms (Sörensen, 2015). However, CRO-SL is not one of those waste-of-time "novel" algorithms because it can be justified getting past the novel metaphor and focusing on its actual design and performance (see Bermejo et al. (2018)).

* Corresponding author.
E-mail addresses: imoya@ugr.es (I. Moya), enrique.bermejo@decsai.ugr.es (E. Bermejo), manuelchica@ugr.es (M. Chica), ocordon@decsai.ugr.es (Ó. Cordón).

https://doi.org/10.1016/j.engappai.2021.104170
Received 7 September 2020; Received in revised form 10 December 2020; Accepted 10 January 2021
Available online xxxx

This study analyzes the behavior of the coral reefs-based metaheuristics using an integer coding scheme for estimating the ABM model's parameters. This approach is selected because representing the values of real parameters following an integer-coded scheme allows the modeler to set the desired granularity for the parameter values (Chica et al., 2017). This increases the control of the modeler over the calibration process and eases the management of the calibrated parameters, since handling long-tailed real values could be troublesome. Additionally, CRO and CRO-SL are extended by hybridizing them with local search procedures with the goal of analyzing their improvement with respect to the original algorithms. Thus, memetic variants (Moscato, 1989; Moscato et al., 2004) are designed and implemented for the metaheuristics.

The performance of the coral reefs-based metaheuristics is evaluated by conducting an exhaustive comparison of CRO and CRO-SL against a well-known metaheuristic that have been previously applied for calibrating agent-based models such as differential evolution (DE) (Storn and Price, 1997). DE is one of the most commonly used metaheuristics because it obtains good results despite of being easy to use and simple to implement. However, there are newer and more advanced metaheuristics in the field of evolutionary computation that can obtain good results calibrating ABM. For example, success-history based adaptive differential evolution with linear population size reduction (L-SHADE) (Tanabe and Fukunaga, 2014) and restart CMA-ES with increasing population size (IPOP-CMA-ES) (Auger and Hansen, 2005) are two advanced and highly competitive metaheuristics that have established as a reference for evolutionary computation, where variants of these algorithms are constantly being developed (Molina et al., 2017). Thus, the performance of CRO and CRO-SL is compared against L-SHADE and IPOP-CMA-ES in addition to DE. On the one hand, L-SHADE is a recent metaheuristic extending the original DE design which is becoming the main DE variant due to its good performance across different benchmarks (Molina et al., 2017). On the other hand, IPOP-CMA-ES is a highly competitive evolutionary algorithm extending the original and very extended CMA-ES metaheuristic (Hansen, 1997) that has proven outstanding results solving complex problems (Biedrzycki, 2017; Biswas and Biswas, 2017; Marín, 2012; Molina et al., 2017).

In order to fairly benchmark the proposed coral reefs-based metaheuristics with the selected evolutionary methods, memetic variants for L-SHADE and IPOP-CMA-ES have also been implemented. The design of this study compares the performance of the metaheuristics when calibrating 12 scenarios. Each of these scenarios is defined by a different instance of an ABM for marketing, with every instance having different dimensionality (i.e., number of decision variables). Therefore, the mentioned calibration instances consider between 24 and 129 parameters to be calibrated, allowing the design of a test suite composed of several ABM calibration problem instances involving several large-dimension optimization problems where a significant number of parameters (beyond 100 parameters) are to be solved. The study also considers two baseline non-evolutionary methods: a hill climbing (HC) (Russell et al., 1995) and a random search procedure. Thus, the resulting battery of experiments is based on five global search metaheuristics, four memetic algorithms, and two baseline methods. Additionally, different statistical tests are performed for evaluating the significance of the results. Hence, the main contributions of the present study are:

- The evaluation of the performance of coral reefs-based metaheuristics when calibrating ABM scenarios using an integer coding scheme.
- The analysis of the improvement obtained by their hybridization with local search procedures.
- The design of an appropriate experimental setup for the study, which considers eleven calibration methods and composed of 12 ABM scenarios with up to 129 decision variables.

The paper is structured as follows. Section 2 reviews the background on model calibration and discusses the related work in the literature. Section 3 introduces the description of the ABM used in the experimentation. Section 4 details the coral reefs-based metaheuristics, the considered coding scheme, and the design of the memetic variants. Section 5 presents the problem instances, competing metaheuristics, and experimental setup. Finally, Section 6 reports the experimental analysis and Section 7 discusses the final remarks.

2. Background and related work

There is a need to carefully validate computational models before they can be used. In this regard, the calibration of the model is recognized as an important step during model validation (Chica et al., 2017; Oliva, 2003). The modeler can perform this process manually similarly to a global sensitivity analysis (ten Broeke et al., 2016), where the modeler repeatedly simulates the model and tunes its parameters based on the observed output. However, this approach is impracticable for many realistic models, which are characterized for considering many parameters. Instead, modelers tend to employ automatic calibration, which is an effective approach to model calibration (Chica et al., 2017; Oliva, 2003).

Automatic calibration techniques have been applied for calibrating the parameters of computational non-linear models from different areas. A few examples would be market modeling (Chica and Rand, 2017; North et al., 2010), crowd modeling (Zhong and Cai, 2015), traffic simulation (Kim and Rilett, 2003; Ngoduy and Maher, 2012), or growth modeling (Trejo-Zúñiga et al., 2014). Some approaches consider the use of exact methods like simplex-based (Kim and Rilett, 2003) or gradient-based (Thiele et al., 2014) methods. However, these approaches are employed for calibrating a relatively low number of parameters (i.e., no more than 20 calibration parameters) and its application to models involving more than 100 parameters seems computationally prohibitive. Another approach is the cross entropy method (Ngoduy and Maher, 2012), a stochastic optimization algorithm which results effective dealing with calibration problems with multi-local optima, but it is also unclear how this approach can perform when dealing with more than 20 calibration parameters.

In contrast, the use of metaheuristics is more convenient for the calibration problem when having higher dimensionality. There are several contributions addressing the application of metaheuristics for model calibration and parameter estimation. For instance, the interested readers can find genetic algorithms (Dai et al., 2009), evolution strategies (Muraro and Dilão, 2013; Trejo-Zúñiga et al., 2014), and several versions of differential evolution (LaTorre et al., 2019; Zhong and Cai, 2015; Trejo-Zúñiga et al., 2014). In terms of the ABM calibration, the use of metaheuristics is clearly the most extended approach. Thus, there are examples of metaheuristics for calibrating ABM that tackle models designed for different areas, such as social and biological sciences (Calvez and Hutzler, 2006; Canessa and Chaigneau, 2015; Chica et al., 2017; Fabretti, 2013; Herrmann and Savin, 2015; Malleson et al., 2014; Moya et al., 2017, 2019). However, one can see from these examples that different versions of genetic algorithms (Back et al., 1997) are usually the default approach, which could be improved by employing recent and more powerful evolutionary metaheuristics. Unfortunately, the state of the art in ABM calibration does not consider a reference benchmark for comparing these methods. Moreover, none of these previous efforts consider an exhaustive comparison of several metaheuristics for ABM calibration, as it is done in this manuscript.

Additionally, efficiently calibrating ABM is troublesome since the model needs to be simulated in order to evaluate the quality of a given set of parameter values, thus leading to a simheuristic approach (Chica et al., 2020). It can be noted the use of surrogate models for reducing the computational cost of estimating these values (van der Hoog, 2019; Lamperti et al., 2018). Specifically, machine learning algorithms have been employed for training a fast surrogate model that responds

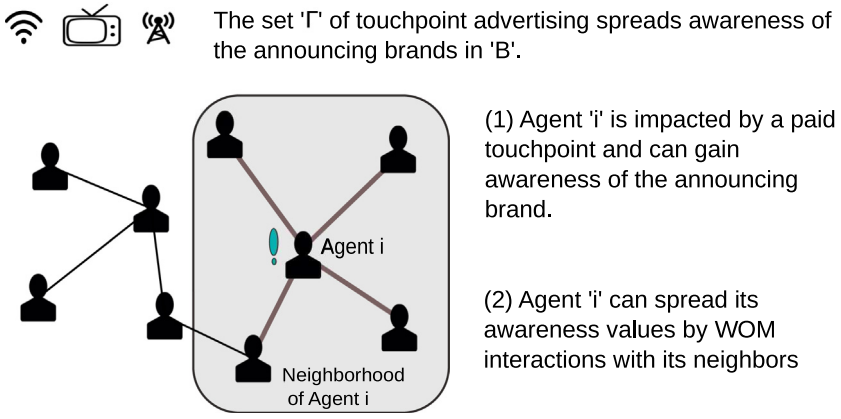


Fig. 1. Summary and main elements of the ABM model. Agents exposed to the advertising of the paid touchpoints can gain awareness of the brand announced and talk about it to their neighbors.

similarly to the changes of parameter values of the original model. Nevertheless, it is unknown how this approach could perform with ABM having a high number of parameters, but authors of the latter publications claimed it can work with more than 30 parameters.

Finally, the use of surrogate fitness functions for tackling expensive optimization has also been addressed in evolutionary optimization (Bhattacharya, 2013; Branke et al., 2017). In this case, the authors propose to replace the original fitness function by an approximate (faster) function. This surrogate function is designed relaxing the conditions of the original fitness function. For example, a fitness function for an ABM could be relaxed by reducing the simulation steps of the model or reducing the number of agents. However, this approach can be problematic from the model calibration point of view because the use of an appropriate surrogate function is problem-dependent (i.e., specific for each ABM) and its validation is not straightforward.

3. Model description

This section introduces the composition of the ABM for marketing employed in the following experiments (Moya et al., 2019). First, the general structure of the model and the behavior of the agents are presented in Section 3.1. Section 3.2 introduces the features of the artificial social network where the agents are embedded. Section 3.3 presents the modeling of external influences as paid touchpoints. Finally, the parameters selected for calibration and the fitness function for the problem are summarized in Section 3.4.

3.1. General structure and definition of agents' behavior

The ABM simulates a given number of weeks (t_{max}) of a market composed of $|B|$ competing brands. It considers the behavior of z agents exposed to the advertising of a set of Γ paid touchpoints and the social interaction between the agents. The model uses a time-step of a week and involves the computation of two outputs or key performance indicators (KPI163(Ns)-363(brand)-363(awareness)-363(and)-323(number)3463(of)3463twod-e o-m(ouch)-917(WOBM))-917(interactios1)-917amoing the conumters, t(at)-917(s1)-917referrted evolme.r(Thuse)-36(0KPIe)-36((are)-36((selected)-36(dure)-36((to)-367(their)-367iomprtmance)-367(with) TJ0 0 0 rg 0 0 0 RG0 0 0 rg 0 0 0 RG 0 -10.504 T

3.3. Modeling of paid touchpoints

The external influences of the agents are modeled as paid touchpoints (i.e., global mass media González-Avella et al., 2007) and mainly act as brand advertising. Using a similar modeling approach to that already applied for social interactions, different parameters are considered for representing the natural differences among the touchpoints (i.e., television, radio, and press). Paid touchpoints γ from Γ can influence any agent of the model at random. The maximum percentage of reached agents depends on the amount invested by the brand and the touchpoints' capability for reaching the agents, which is bounded by the properties of the touchpoint itself. For example, an ad included in the press will impact the population percentage that reads the press at best. These different properties are modeled by a reach parameter ($r_\gamma \in [0,1], \forall \gamma \in \Gamma$), which bounds the maximum number of agents that a touchpoint γ can try to influence at a single step.

Touchpoint advertising is modeled according to an investment measure called gross rating points (GRPs). In advertising (Farris et al., 2010), a GRP measures the potential of the campaigns scheduled in mass media assuming that one GRP represent enough impressions for reaching 1% of the target population. The variable $\chi_\gamma^b(t)$ models the investment units in GRPs for touchpoint γ by brand b and time step t. Because increasing the awareness or the number of conversations of the target population using paid touchpoints implies a monetary cost, the brands need to define their marketing mix considering that each touchpoint has a different cost for the invested GRP units. Touchpoint advertising is scheduled by computing the number of actual impressions for each band and step using the GRP values and the size of the agent population. Then, these impressions are individually assigned at random between the agents respecting the reach restrictions for the touchpoint.

Similarly to the social interactions, each touchpoint is assigned an awareness impact parameter ($\alpha_\gamma \in [0,1], \forall \gamma \in \Gamma$) that models the probability of the agent to activate its awareness of the ad brand after a single touchpoint impact. In addition, the effect of the advertising transmitted by paid touchpoints can create a viral effect in the reached agent, as done in Moya et al. (2017). The buzz effect created by the touchpoints produces an increment of the number of conversations regarding the announcing brand by increasing the talking probability (p_i^b) of the impacted agents. This effect is modeled using a buzz increment parameter (τ_γ) defined for each touchpoint $\gamma \in \Gamma$. The increment produced on the agents' talking probability is calculated as a percentage increment over the initial talking probability ($p_i^b(0)$) of the agent. Similarly to the awareness, buzz effect decays over time if it is not reinforced. This effect is modeled by Eq. (1), where the action of the buzz decay parameter ($d\tau_\gamma$) erodes the increment of talking probability (σ_γ) previously supplied to the agent by touchpoint γ. Finally, a summary of the variables and parameters of the calibrated ABM is shown at Table 1.

$$p_i^b(t+1) = p_i^b(t) - \sigma_{i\,\gamma}^b(t) \cdot d\tau_\gamma + p_i^b(0) \cdot \tau_\gamma, \tag{1}$$

$$\text{where } \sigma_{i\,\gamma}^b(t) = \sum_{s=1}^{t} (p_i^b(s) - p_i^b(0) \cdot \tau_\gamma).$$

3.4. Calibration parameters and fitness function

The parameters selected for automatic calibration are those that control either the dynamics of the agents' awareness values or their number of conversations since those are sensitive parameters that are hard to estimate by the modeler using the available data (Moya et al., 2019). The calibration process assigns each of the selected model parameters to a single decision variable limiting the range of possible parameter values to a real-coded interval, $[0,1]$. Additionally, the final set of parameters that are selected for calibration is determined by the size of the model instance (i.e., the number of paid touchpoints $|\Gamma|$): three parameters for each touchpoint plus the three social parameters. Briefly, for each defined paid touchpoint $\gamma \in \Gamma$, the calibration considers its buzz increment (τ_γ), buzz decay ($d\tau_\gamma$), and awareness impact (α_γ). In addition, the initial talking probability ($p_i^b(0)$), social awareness impact (α^{WOM}), and awareness deactivation (d) are also calibrated. Thus, the overall number of parameters being calibrated is computed as $(|\Gamma|+1)\cdot 3$.

Finally, Eqs. (2) and (3) define the selected fitting functions for the historical data of the awareness (f_1) and WOM volume (f_2). Both functions compute the deviation error comparing the simulated outputs values with the target data of the objectives. This deviation is computed using the standard mean absolute percentage error function, where $\widetilde{a}$ represents the target awareness values and $\widetilde{\omega}$ represent the target WOM volume values. The simulated values are the result of averaging multiple independent Monte-Carlo simulations in order to account for the random nature of ABMs. Using the latter fitting functions, the objective function f is defined as its weighted combination using parameter $\beta \in [0,1]$: $f = \beta \cdot f_1 + (1-\beta)\cdot f_2$.

$$f_1 = \frac{100}{t_{\max} \cdot |B|} \sum_{b=1}^{|B|} \sum_{t=1}^{t_{\max}} \left| \frac{a^b(t) - \widetilde{a}^b(t)}{\widetilde{a}^b(t)} \right|, \tag{2}$$

$$f_2 = \frac{100}{t_{\max} \cdot |B|} \sum_{b=1}^{|B|} \sum_{t=1}^{t_{\max}} \left| \frac{\omega^b(t) - \widetilde{\omega}^b(t)}{\widetilde{\omega}^b(t)} \right|. \tag{3}$$

4. Coral reefs-based metaheuristics for model calibration

This section introduces the design of CRO and CRO-SL metaheuristics for calibrating ABM. Then, the coding scheme considered for the problem and the memetic design due to their hybridization with a local search procedure are described in detail.

4.1. CRO and CRO-SL algorithms

CRO (Salcedo-Sanz et al., 2014; Salcedo-Sanz, 2017; Salcedo-Sanz et al., 2017b) is an advanced evolutionary metaheuristic based on the processes occurring in a coral reef. In the analogy, a coral reef is represented by a two dimensional grid able to allocate a colony of corals (i.e., a population with variable size), where a coral represents a solution to the optimization problem. The CRO algorithm initializes some positions of the grid with random solutions, leaving the rest of the available spots empty. The second phase simulates the processes of coral reproduction and reef formation which are recreated by applying known evolutionary process in four sequential stages:

1. **External reproduction.** This stage consists of the random selection of a fraction (F_b) of the existing solutions and the generation of a pool of child solutions. These new solutions are generated by applying a crossover operator or any other exploration strategy with two existing solutions as parents. Once a solution has been selected as parent, it is not chosen for reproduction purposes during an iteration.
2. **Internal reproduction.** This reproduction is modeled as a random mutation mechanism that takes place on the remaining fraction of solutions ($1-F_b$). A percentage P_i of the solution is mutated. The subset of mutated solutions is added to the pool of children solutions.
3. **Replacement.** Once the children solutions are formed either through external or internal reproduction, they will try to set in the grid. Each solution in the pool will randomly try to set in a position of the grid and settle if the location is free. Otherwise, the new solution will replace the existing one only if its fitness is better. Each child in the pool can attempt to occupy a position at each iteration only a determined number of tries η.
4. **Elimination.** At the end of each reproduction iteration, a small number of solutions in the grid are discarded, thus releasing space in the grid for next generation. The elimination operator is applied with a very small probability (P_d) to a fraction (F_d) of the grid size with worse fitness.

Table 1
Summary of the parameters of the agent-based model for marketing scenarios.

Market parameters	
z: number of agents running in the model	$\|B\|$: number of considered brands
$\|\Gamma\|$: number of paid touchpoints	$t_{\max}$: number of steps of the simulation
$a^b(0)$: Initial awareness for brand b	d: awareness deactivation probability
WOM parameters	
$p_i^b(0)$: initial talking probability, same value for each brand b	α^{WOM}: awareness impact for social interactions
	m: parameter for social network generator
Touchpoint parameters	
χ_γ^b: GRP units invested by brand b in touchpoint γ	r_γ: reach for paid touchpoint γ
	$d\tau_\gamma$: buzz decay for paid touchpoint γ
α_γ: awareness impact for paid touchpoint γ	τ_γ: buzz increment for paid touchpoint γ

However, there are other relevant interactions in real reef ecosystems that can be incorporated into the CRO approach for improving its performance in optimization and search problems. For example, different recent studies have shown that successful recruitment in coral reefs (i.e., successful settlement and subsequent survival of larvae) strongly depends on the type of substrate on which they fall after the reproduction process (Vermeij, 2005). This specific characteristic of coral reefs was first included in the CRO algorithm in Salcedo-Sanz et al. (2017b) in order to solve different instances of the Model Type Selection Problem for energy applications, resulting in CRO-SL (CRO with substrate layers). In Salcedo-Sanz et al. (2016b), CRO-SL is enhanced with several substrate layers providing a different search procedure.

The inclusion of substrate layers in CRO can be carried out in a straightforward way: the artificial reef considered in the CRO is redefined in such a way that each cell of the square grid Ψ is now associated with a different exploration layer that indicates which structure it belongs to (search operator in this case). Each solution in the grid is then processed in a different way (with a different search operator) depending on the region (specific layer) in which it falls after the reproduction process. Fig. 2 illustrates the flowchart diagram of the CRO-SL algorithm. Any exploration operator can be integrated in the algorithm as a substrate layer as long as it follows the determined coding scheme. In order to select a well-performing combination of substrate layers for the ABM calibration problem tackled in the current contribution, a preliminary test was developed considering different search operators known to perform well when tackling the addressed problem. The considered exploration operators are detailed below.

- **Uniform mutation**. The uniform random mutation is a traditional mutation operator that replaces the value of a gene for a given individual by a random value. This value is generated using an uniform distribution from an interval defined by lower and upper bounds.
- **Random walk mutation**. This operator modifies the values of a given individual by including neighborhood information into the mutation process. In the context of the present problem, the neighbors of a solution are accessed increasing or decreasing its values. This way, this operator modifies the value of each parameter of the individual solution with probability p_m by randomly moving to a neighbor. After moving to a random neighbor, the operator will keep moving to a random neighbor until the probability check fails.
- **Simulated binary crossover** (SBX). SBX (Deb and Agrawal, 1994) performs the crossover operation emulating the behavior of a single-point crossover from binary-encoding. This operator works as follows: given two parents $P_1 = (p_{11}, \ldots, p_{1n})$ and $P_2 = (p_{21}, \ldots, p_{2n})$, SBX generates two springs $O_1 = (o_{11}, \ldots, o_{1n})$ and $O_2 = (o_{21}, \ldots, o_{2n})$ as $o_{1i} = \bar{X} - \frac{1}{2} \cdot \bar{\beta} \cdot (p_{2i} - p_{1i})$ and $o_{2i} = \bar{X} + \frac{1}{2} \cdot \bar{\beta} \cdot (p_{2i} - p_{1i})$, where $\bar{X} = \frac{1}{2}(p_{1i} + p_{2i})$. $\bar{\beta}$ is a random number fetched from a probability distribution which is employed as a spread factor.

Fig. 2. Flowchart of the CRO-SL algorithm, which modifies the reproduction stages of the original CRO by selecting the operator mechanisms depending on the substrate layers.

- **BLX-α crossover**. BLX-α (Eshelman and Schaffer, 1993) takes new values from the interval defined by $[c_{\min} - I\alpha, c_{\max} + I\alpha]$, $c_{\max} = \max(v_i^1, v_i^2)$, $c_{\min} = \min(v_i^1, v_i^2)$ and $I = c_{\max} - c_{\min}$. v_i^1, v_i^2 represent the decoded values from the chromosome of the parents. In this regard, α defines the level of exploration for the operator. For example, if α is set to 0, BLX-α works as a flat crossover.

Nevertheless, it is noted that the recent surge of novel bio-inspired algorithms has been subject of controversy (Sörensen, 2015) due to the lack of scientific rigor behind some of these algorithms. CRO-SL is not one of those waste-of-time "novel" algorithms, as it can be justified beyond the novelty of the metaphor and only focusing on its

actual design and performance (Bermejo et al., 2018). The novelty of CRO-SL resides in an excellent exploration–exploitation trade-off and robustness as a consequence of the combination of multiple search patterns in its scheme. Thus, the CRO-SL general approach aims for a grid-based competitive co-evolution (Floreano and Mattiussi, 2008) in just one population where each substrate layer represents a different search process (different models, operators, parameters, constraints, repairing functions, etc.). Therefore, CRO-SL is a powerful multi-method ensemble where multiple search strategies coexist in the same population and can be classified as a low-level competitive single population approach (Del Ser et al., 2019; Wu et al., 2019). Such design provides CRO-SL with an exceptional versatility, being able to adapt different approaches to tackle a wide variety of problems while considering a single, general algorithmic scheme.

Finally, CRO-SL usually converges quickly to high quality solutions even in multi-modal search spaces, being suitable for computationally expensive optimization problems both satisfying quality and computation time constraints. However, its performance varies significantly depending on the CRO's parameters and the different substrates included in the simulated reef. This idea of CRO with substrate layers as a competitive co-evolution algorithm was successfully tested in different applications and problems such as micro-grid design (Salcedo-Sanz et al., 2016a), vibration cancellation in buildings (Salcedo-Sanz et al., 2017a), 3D medical image registration (Bermejo et al., 2018), and in the evaluation of novel non-linear search procedures (Salcedo-Sanz, 2016).

4.2. Coding scheme

As previously described in Section 3.4, the presented design assigns each of the selected model parameters to a single decision variable. Thus, during the metaheuristics execution each candidate solution consists of an array of values that represent the values of the parameters being calibrated. This design considers an integer coding scheme which encodes the individual solutions as discrete values. This coding scheme was selected because it provides a good trade-off between precision and usability, since it does not require the users to precisely adjust long-tailed real values.

This integer coding scheme samples a given continuous interval into several discrete values following Eq. (4), where the number of possible values of parameter i is computed using a minimum parameter value ($\min_i$), a maximum parameter value ($\max_i$), and a step (ϵ_i) value that regulates the granularity of the search procedure. Fig. 3 shows the representation of an individual solution using the integer coding scheme for an example model instance which considers three touchpoints using $\min = 0$, $\max = 1$, and $\epsilon = 0.001$ for each parameter in the example.

$$n_i = \left\lfloor \frac{\max_i - \min_i}{\epsilon_i} \right\rfloor + 1. \tag{4}$$

4.3. Memetic algorithms

Memetic algorithms (Moscato, 1989) combine the exploration capability of global search methods with the intensification of local search procedures, improving the quality of the candidate solutions found during the regular execution of the algorithm. The most important design issue is to define a trade-off between the local search procedure and the global search method (Ishibuchi et al., 2003). Some common approaches to memetic algorithms apply this local search refinement to every solution obtained during its run but this strategy can be time-consuming and it has proven to not always lead to the best performing memetic design (Krasnogor and Smith, 2000). Therefore, the selected approach performs local search refinement selectively with a probability p_{LS} (Krasnogor and Smith, 2000; Lozano et al., 2004). Thus, all the solutions in the population are candidates for receiving local search refinement since each solution runs a probability check at the end of a generation.

The designed memetic algorithms are the result of the hybridization of CRO and CRO-SL with a local search component. This local optimizer implements a hill climbing strategy. It refines the quality of the individuals by increasing or decreasing the encoded parameter values by a given quantity specified by a step parameter (ϵ). The result of applying this operation over an individual is called a neighbor generation. Thus, the set of all possible neighbors for a given solution is called its neighborhood. The local search component moves to the first neighbor that improves the current solution quality until no neighbor shows an improvement. This search is repeated for a given number of iterations since it is integrated with the global search of the coral reefs-based metaheuristics. The memetic algorithms proposed for CRO and CRO-SL are labeled MCRO and MCRO-SL, respectively.

5. Experimental details

This section reviews the design of the experimentation. First, the model instances considered for the experiments are presented in Section 5.1. Then, the selected metaheuristics for benchmarking the coral reefs-based methods are introduced in Section 5.2. Afterwards, Section 5.3 reviews the experimental setup and the parameter configuration of the different metaheuristics.

5.1. Calibration scenarios and datasets

The experimentation considers 12 different instances of the ABM model. These model instances were generated using an initial *baseline* instance, referred as P1(24), which models a real banking marketing scenario from one of the authors' research contracts. The rest of the instances are synthetically generated increasing the complexity of the initial instance. Each model variation introduces additional touchpoints that are built from the initial ones by modifying its investment values. In addition, each model includes a perturbation of the target historical values for both KPI, awareness and WOM. Each of the newly generated instances increases the dimensionality of the previous one, including new decision variables that enable a more complete comparison of the different metaheuristics.

The modifications on the existing touchpoints $\gamma \in \Gamma$ consist of either increasing or reducing the original investment of each brand for each of its steps, multiplying its value by a given factor. The following reduction factors for the original values are considered: 15%, 30%, 45%, and 60%. In addition, the considered increase factors are 100%, 200%, 300%, and 400%. These factors were inspired by the investment data used during the design of the baseline model where paid touchpoints can receive different investment in different years. The election of an increase or reduction factor is made at random and stays unchanged for each step. In contrast, the modifications in the target values of awareness and WOM volume are applied adding or subtracting a given quantity to each of its time steps. This way, each modification on the target awareness values adds or subtracts 2%, 5%, 8%, or 10% to the target brand values. The resulting awareness values are truncated between 1% and 100% for avoiding unrealistic target values. With respect to target WOM volume, each modification is of either 1000, 2000, 4000, or 6000 conversations, always keeping the conversation values above 0. Like in the case of touchpoint investment, the decision of increasing or reducing the target values is made at random and maintained at each step.

All problem instances are labeled using its number of decision variables: P1(24), P2(39), P3(45), P4(54), P5(60), P6(69), P7(75), P8(84), P9(90), P10(99), P11(114), and P12(129). The parameter values of the baseline instance for each parameter not modified during calibration are the following. The main parameters are $z = 1000$, $|B| = 8$, $|\Gamma| = 7$, and $t_{\max} = 52$. The awareness of the agent population is initialized as $a^b(0) = (0.71, 0.75, 0.58, 0.25, 0.08, 0.42, 0.39, 0.34)$ and the touchpoint

α_1	α_2	α_3	τ_1	τ_2	τ_3	$d\tau_1$	$d\tau_2$	$d\tau_3$	$p^b(0)$	α^{WOM}	d
20	15	50	100	150	220	120	50	230	200	100	10

Fig. 3. Example representation of a model instance which considers three paid touchpoints using an integer coding scheme. The genes of the chromosome represent real parameters limited to $[0, 1]$.

Table 2
Summary of the simulation parameters that are not modified during the calibration process.

General market settings	Initial awareness	Touchpoints' reach
$z = 1000$	$a^1(0) = 0.71$	$r_1 = 0.92$
$\|B\| = 8$	$a^2(0) = 0.75$	$r_2 = 0.57$
$\|\Gamma\| = 7$	$a^3(0) = 0.58$	$r_3 = 0.54$
$t_{\max} = 52$	$a^4(0) = 0.25$	$r_4 = 0.035$
	$a^5(0) = 0.08$	$r_5 = 0.43$
	$a^6(0) = 0.42$	$r_6 = 0.38$
	$a^7(0) = 0.39$	$r_7 = 0.69$
	$a^8(0) = 0.34$	

reach parameters are set to $r_\gamma = (0.92, 0.57, 0.54, 0.035, 0.43, 0.38, 0.69)$. Finally, the network generation parameter in the Barabasi–Albert algorithm is set to $m = 4$. A summary with these simulation parameters is shown in Table 2.

The generated instances share this initial setup along with its corresponding reach parameter value r_γ for the new paid touchpoints, which take the value of the original touchpoint employed in its generation. This way, if a new touchpoint 12 is generated using the original touchpoint 3, they share the reach parameter value (i.e., $r_{12} = r_3$).

5.2. Metaheuristics considered for ABM automatic calibration task

Two baseline methods and three evolutionary algorithms were selected for benchmarking the coral reefs-based metaheuristics. The first baseline method follows a random search approach (RND). This procedure simply creates valid solutions by randomly generating values for each of the decision variables. This way, random solutions are generated until the stopping criteria is met and the method returns the best solution found. The second baseline method is a local search procedure implementing a HC strategy, similar to the one employed by the memetic algorithms. However, instead of starting for a given individual of the coral population, the baseline HC starts from a random individual generated using a uniform distribution. The HC search continues until the stopping criteria is met, which involves a much higher number of steps than that considered for the memetic variants.

The first evolutionary algorithm considered in the study is differential evolution (DE) (Storn and Price, 1997). This is the metaheuristic considered in several automatic calibration approaches (LaTorre et al., 2019; Trejo-Zúñiga et al., 2014; Zhong and Cai, 2015) and has been shown to provide a better performance than the basic genetic algorithms used in other contributions when calibrating ABMs (Herrmann and Savin, 2015; Zhong and Cai, 2015). DE generates new solutions by combining the existing individuals with a *donor vector* created following the next equation: $x_i(G+1) = x_{r1}(G) + F(x_{r2}(G) - x_{r3}(G))$, where x_i is the generated donor vector and $r1$, $r2$, and $r3$ are different solutions at generation G. For each generation, a donor vector x_i is generated for every individual i and its values are combined with those of the original individual by means of an uniform crossover according to a CR parameter. For every gene of the newly created individual, the algorithm takes the value of the donor vector with probability CR, otherwise it takes the original value of individual i. If the fitness value of the resulting individual is better or equal than that of individual i, it will replace i in the population. Additionally, the selected design considers the variant DE/best/1/bin (Storn and Price, 1997). In this variant, "best" refers to the fact that the best individual in the population is always selected as the individual $x_{r1}(G)$ in the previous operator to obtain the mutated offspring $x_i(G+1)$. Meanwhile, the values "1" and "bin" respectively stand for the use of a single vector difference in that operator, as shown in the former equation, and of the uniform crossover operator, as also described.

The second evolutionary algorithm considered in the study is success-history based adaptive differential evolution with linear population size reduction (L-SHADE) (Tanabe and Fukunaga, 2014). L-SHADE is an extension of SHADE, a history-based variant of differential evolution (Tanabe and Fukunaga, 2013). SHADE and other adaptive variants of DE self-adapt the values of the crossover rate (CR) and the mutation rate (F) during the optimization process, which are the main control parameters. In the case of SHADE, the *successful* values of CR and F are stored into a historical memory. A parameter value is *successful* if the solution generated using it improves the previous individual. In addition, L-SHADE extends SHADE by including a mechanism for reducing the population after each generation. This population reduction is computed linearly with respect of the number of fitness evaluations, which improves the convergence of the algorithm as the search process advances.

Finally, the third evolutionary algorithm is a restart CMA-ES with increasing population size (IPOP-CMA-ES) (Auger and Hansen, 2005). CMA-ES is an evolution strategy that relies on a covariance matrix for sampling new search points. At each iteration of the optimization process, λ new solutions are independently sampled according to a multi-variate normal distribution. Then, the best μ solutions (or search points, as referred by CMA-ES literature) are weighted and summed for obtaining a new mean value for the distribution. CMA-ES employs the fitness information of previous iterations (called the evolution path) in combination with the latest solutions sampled for updating the covariance matrix. IPOP-CMA-ES extends the typical CMA-ES design by including a restart strategy that increases the current population size by a given factor each time the algorithm restarts because a stopping criteria is met. These stopping criteria typically involve a stagnation of the search process (Auger and Hansen, 2005).

In addition, memetic variants for L-SHADE and IPOP-CMA-ES were also included, which are referred as ML-SHADE and MIPOP (for short). These memetic algorithms include the same local optimizer employed for MCRO and MCRO-SL and the local search refinement is applied with a probability p_{LS} to any solution of the population. To sum up, 11 metaheuristics are compared for the automatic ABM calibration problem: the coral reefs-based metaheuristics, two baseline methods, three evolutionary algorithms, and four memetic variants.

5.3. Experimental setup

The selected metaheuristics were implemented in Java using the ECJ framework (Luke, 1998). Since this framework does not include the coral reefs-based algorithms, the CRO implementation is integrated by modifying some ECJ components. Readers can access the code for the algorithms and the experimentation in GitHub: https://github.com/nachomsenias/abmcalibration. Each metaheuristic runs 20 independent times using different random seeds. Every algorithm execution considers 10,000 evaluations as stopping criteria. During each evaluation of a candidate solution, the objective function f is computed using the output of each independent run with the provided historical data considering $\beta = 0.5$ (i.e. the same weight is considered for both KPIs). Due to the highly time-consuming task of simulating multiple times for every parameter configuration, the final fitness of the individual is calculated as the average value of f for 15 Monte-Carlo simulations of the ABM using the parameters encoded in the individual.

With respect to the coding scheme, since the parameters of this calibration problem are real values limited to $[0, 1]$, the integer-coded approach transforms the parameter values using $\min_i = 0$, $\max_i = 1$, and $\epsilon_i = 0.001$, for every parameter i, resulting in 1,001 possible values. The parameter setup for the metaheuristics was specified via a preliminary experimentation and the final values are as follows:

- HC movements increase or decrease the integer value of the genes by one unit. Each HC run takes 200 iterations and applies 50 movements during each iteration.
- DE considers a population size of 100 individuals. The parameter value for the crossover rate is set to CR = 0.9 and the value of the mutation rate is set to F = 0.5.
- L-SHADE uses an initial population size of 100 individuals and a external archive size of 200. During the reproduction phase, the p value for the current-to-pbest/1/bin strategy is set to $p = 0.1$. Finally, the size of the historical memory is set to the dimensionality of the problem.
- IPOP-CMA-ES considers a population size of $\lambda = 15$ with $\mu = 6$ and an increasing factor of 2. In addition, the learning rates are set to $c_\sigma = 0.568$, $c_c = 0.6962$, and $c_{\text{cov}} = 0.4897$. Since the calibration problem transforms the search space from the interval $[0, 1]$ into $[0, 1000]$, a relatively high $\sigma^{(0)}$ value of $\sigma^{(0)} = 56.747$ is selected. Finally, the dampening for the step size update is set to $d_\sigma = 4.2939$.
- CRO uses a reef size of 50 individuals. Regarding the BLX-α crossover, the probability is set to $p_c = 0.2$ using $\alpha = 0.25$ and tournament selection of size 3. The mutation probability of the random mutation is set to $p_m = 0.1$. The rest of CRO parameter values are $k = 3$, $\rho_0 = 0.6$, $F_a = 0.05$, $F_b = 0.9$, $F_d = 0.08$, and $P_d = 0.15$.
- CRO-SL defines a reef populated by 50 individuals. Its CRO-based values are set to $k = 3$, $\rho_0 = 0.6$, $F_a = 0.05$, $F_b = 0.9$, $F_d = 0.05$, and $P_d = 0.15$. The substrate layers of CRO-SL integrate uniform random mutation, random walk mutation, SBX, and BLX-α, and its parameter values are $p_m = 0.2$, $p_c = 1$, and $\alpha = 0.51$.
- Regarding the memetic variants, the refinement probability is set to $p_{\text{LS}} = 0.0625$ since it is the recommended value for similar problems (Lozano et al., 2004). Each time an individual is refined using the local search procedure, it will use 50 evaluations using the local search procedure. The rest of the parameters of the memetic algorithms are shared with its corresponding global search algorithm (i.e., MCRO has the same setup as CRO).

6. Analysis of results

In order to conduct a clear and comprehensive analysis of the different metaheuristics' performance, the analysis of the calibration results considers different stages. After the preliminary analysis, the improvement of including memetic approaches to the original metaheuristics is studied. Finally, the best performing methods are selected and studied at the end of this section.

Table 3 presents the performance of the original metaheuristics (i.e., without considering the memetic component). This information is complemented with a statistical test considering the ranking of the metaheuristics and with several post-hoc procedures for highlighting significant differences in their performance. Specifically, Friedman's nonparametric test (Friedman, 1940), Bonferroni–Dunn's test (Dunn, 1961), and Holm's test (Holm, 1979) are used. The average ranking and the resulting p−values of Bonferroni–Dunn's test and Holm's test are shown in Table 4. With respect to the Friedman's test, the result of applying the test is $\chi^2_F = 64.35$ and its corresponding p−value is $5.8 \cdot 10^{-12}$. As the p−value is lower than the desired level of significance ($\alpha = 0.01$), the test concludes that there are significant differences in the algorithms' performance.

It can be observed in these results that CRO-SL outperforms the other algorithms since it ranks first achieving the lowest mean rank (1.5) and finds the configuration with lowest fitness value for almost every model instance. After CRO-SL, CRO and IPOP-CMA-ES rank second and third with close ranking values (2.25 and 2.42, respectively). L-SHADE and DE rank fourth and fifth but perform significantly worse than the control method CRO-SL, since the p−values from both Bonferroni–Dunn's and Holm's tests (0.003 and 0.001 in the case of L-SHADE, 0.001 and 0.001 in the case of DE) are lower than the considered significance level ($\alpha = 0.05$). Finally, the baseline methods HC and RND also perform significantly worse than the control method and rank last for every model instance.

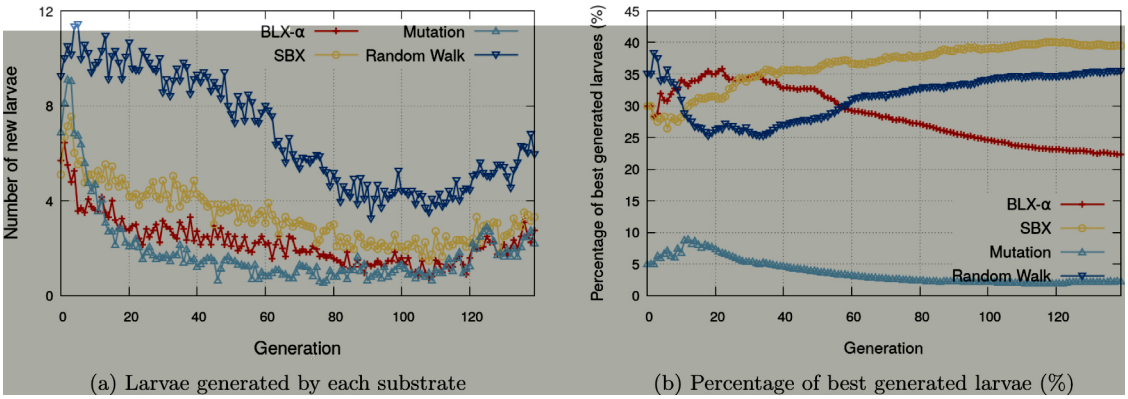
The performance of CRO-SL can be analyzed in terms of the behavior of the different search procedures when solving a single problem instance. Thus, the P12(129) instance is selected as it is the instance with the highest dimensionality. Fig. 4 shows the number of settling larvae (i.e., new solutions in the population) obtained by each substrate

Table 3
Fitness function values obtained by the metaheuristics for every ABM instance. The values are shown using average, standard deviation, and minimum result of the 20 different runs.

		RND	HC	DE	LSHADE	IPOP	CRO	CRO-SL
P1(24)	Avg.	34.03	37.46	31.48	29.57	**25.79**	27.87	26.98
	Std. dev.	0.84	1.24	2.12	1.65	0.27	2.86	2
	Best	32.18	34.45	26.82	25.69	25.51	25.19	24.96
P2(39)	Avg.	53.26	51.9	42.87	42.74	43.02	40.76	**38.13**
	Std. dev.	1.63	5.84	0.57	1.02	0.55	1.19	0.24
	Best	49.8	45.05	42.11	40.86	42.35	38.36	37.85
P3(45)	Avg.	31.56	34.57	28.87	27.59	**23.99**	26.83	25.17
	Std. dev.	0.55	1.76	1.05	0.64	0.94	2.44	1.16
	Best	30.48	31.57	24.93	26.18	23.42	23.79	23.65
P4(54)	Avg.	42.93	46.04	37.59	37.48	**31.85**	31.95	32.55
	Std. dev.	0.66	1.46	2.17	1.51	0.42	1.87	3.34
	Best	41.55	42.5	33.68	34.41	31.41	30.86	29.19
P5(60)	Avg.	37.54	39.58	32.54	32.17	**26.66**	27.53	26.72
	Std. dev.	0.56	1.17	1.76	1.13	0.25	1.38	1.95
	Best	36.47	37.11	28.33	29.01	26.35	25.58	25.05
P6(69)	Avg.	55.39	59.58	47.54	43.52	42.41	40.82	**39.32**
	Std. dev.	1.01	1.48	0.82	0.88	0.93	1.51	0.92
	Best	53.47	57.1	45.35	42.32	41.22	39.37	38.34
P7(75)	Avg.	46.33	48.29	39.09	40.63	35.41	**35.09**	35.14
	Std. dev.	0.66	1.03	1.22	1.81	0.95	1.09	2.22
	Best	45.08	45.51	34.88	35.86	34.5	34.35	32.59
P8(84)	Avg.	55.16	58.1	37.63	43.41	32.82	32.59	**28.7**
	Std. dev.	0.94	1.89	0.22	2.5	1.61	2.1	1.87
	Best	53.02	53.79	37.02	38.76	31.66	29.36	25.91
P9(90)	Avg.	49.14	51.35	31.04	35.14	27.91	27.2	**26.54**
	Std. dev.	0.6	0.94	1.05	2.76	1.62	1.99	2.47
	Best	47.84	49.87	27.93	30.7	26.05	25.38	24.23
P10(99)	Avg.	72.37	80.35	54.72	52.79	48.05	46.02	**40.81**
	Std. dev.	1.64	3.9	0.93	1.06	1.75	4.05	1.18
	Best	69.54	71.84	51.33	50.28	46.09	41.36	39.01
P11(114)	Avg.	48.61	50.93	33.98	37.63	31.04	30.28	**26.47**
	Std. dev.	0.47	1.03	1.02	2.99	1.94	2.22	1.85
	Best	47.8	49	29.95	34.94	27.71	28.15	24.55
P12(129)	Avg.	41.96	43.81	34.07	34.88	28.24	30.94	**26.92**
	Std. dev.	0.56	0.79	1.08	1.25	0.65	1.38	0.57
	Best	40.55	42.51	31.96	32.55	27.46	27.68	25.87

Table 4
Average ranking of the metaheuristics and their resulting p−values for Bonferroni's and Holm's test using CRO-SL as the control method.

	Rank	Bonferroni p	Holm p
CRO-SL	1.5	–	–
CRO	2.25	1	0.52
IPOP-CMA-ES	2.42	1	0.52
L-SHADE	4.33	0.003	0.001
DE	4.5	0.001	0.001
RND	6.08	$1.4 \cdot 10^{-7}$	$1.2 \cdot 10^{-7}$
HC	6.92	$2.6 \cdot 10^{-10}$	$2.6 \cdot 10^{-10}$



(a) Larvae generated by each substrate (b) Percentage of best generated larvae (%)

Fig. 4. (a) Number of new larvae settling in the reef per substrate at each generation. (b) Percentage of best larvae obtained by a substrate at each generation. These values are computed as the average of the multiple Monte-Carlo runs of CRO-SL for P12(129) instance.

Table 5
Fitness function values obtained by the memetic algorithms for every model instance. The values are shown using average, standard deviation, and best result of the 20 different runs.

		LSHADE	MLSHADE	IPOP	MIPOP	CRO	MCRO	CRO-SL	MCRO-SL
P1(24)	Avg.	29.57	32.53	25.79	25.98	27.87	**25.73**	26.98	26.75
	Std. dev.	1.65	1.65	0.27	0.68	2.86	1.34	2	2.19
	Best	25.69	28.37	25.51	25.53	25.19	25.07	24.96	25.13
P2(39)	Avg.	42.74	46.39	43.02	42.57	40.76	39.42	**38.13**	40.09
	Std. dev.	1.02	2.23	0.55	0.62	1.19	0.73	0.24	1.64
	Best	40.86	42.42	42.35	41.95	38.36	38.45	37.85	38.34
P3(45)	Avg.	27.59	30.33	23.99	**23.81**	26.83	26.1	25.17	27.23
	Std. dev.	0.64	1.04	0.94	0.39	2.44	3.21	1.16	1.04
	Best	26.18	27.05	23.42	23.47	23.79	22.42	23.65	24.91
P4(54)	Avg.	37.48	40.22	31.85	31.79	31.95	**30.62**	32.55	34.09
	Std. dev.	1.51	2.39	0.42	0.5	1.87	0.51	3.34	3.22
	Best	34.41	34.93	31.41	31.19	30.86	29.52	29.19	30.02
P5(60)	Avg.	32.17	34.73	26.66	27.09	27.53	**25.87**	26.72	27.54
	Std. dev.	1.13	2.23	0.25	1.08	1.38	0.74	1.95	2.33
	Best	29.01	29.4	26.35	25.97	25.58	25.06	25.05	25.24
P6(69)	Avg.	43.52	52.18	42.41	42.76	40.82	**39.2**	39.32	41.49
	Std. dev.	0.88	1.94	0.93	0.8	1.51	0.26	0.92	2.15
	Best	42.32	47.09	41.22	42.05	39.37	38.83	38.34	38.85
P7(75)	Avg.	40.63	44.82	35.41	35.71	35.09	**33.6**	35.14	36.24
	Std. dev.	1.81	2.34	0.95	0.59	1.09	0.47	2.22	2.45
	Best	35.86	39.54	34.5	35.02	34.35	32.82	32.59	33.29
P8(84)	Avg.	43.41	46.99	32.82	32.41	32.59	**27.61**	28.7	34.49
	Std. dev.	2.5	3.65	1.61	2.03	2.1	0.71	1.87	2.97
	Best	38.76	38.83	31.66	29.34	29.36	26.46	25.91	28.1
P9(90)	Avg.	35.14	42.01	27.91	27.69	27.2	**24.69**	26.54	27.03
	Std. dev.	2.76	3.56	1.62	0.76	1.99	0.38	2.47	2.72
	Best	30.7	32.9	26.05	26.81	25.38	24.24	24.23	24.3
P10(99)	Avg.	52.79	59.1	48.05	48.08	46.02	42.55	**40.81**	45.51
	Std. dev.	1.06	4.67	1.75	1.09	4.05	1.92	1.18	2.01
	Best	50.28	55.39	46.09	46.34	41.36	39.91	39.01	42.75
P11(114)	Avg.	37.63	39.42	31.04	30.59	30.28	**25.97**	26.47	30.08
	Std. dev.	2.99	5.31	1.94	1.71	2.22	0.89	1.85	3.55
	Best	34.94	34.51	27.71	28.53	28.15	24.83	24.55	25.45
P12(129)	Avg.	34.88	37.03	28.24	28.03	30.94	27.22	**26.92**	28.47
	Std. dev.	1.25	2.83	0.65	0.51	1.38	0.7	0.57	1.18
	Best	32.55	31.08	27.46	27.42	27.68	26.26	25.87	26.43

at each generation and the percentage of times each substrate produces the best solution of the generation. These plots show that SBX and random walk are the best performing substrates since they end up generating 40% and 35% of the best quality solutions, respectively. Random walk also stands out as the substrate generating the higher number of solutions which replace other solutions in the reef due to their quality. It can also be observed that every substrate is productive during the first generations when there is space in the reef available for new solutions. A higher number of solutions during the first generations are produced by the mutation substrates, in a period when exploration of the search space is crucial. BLX-α decays after obtaining some of the best solutions during the first generations and later on the operator SBX stands out by delivering good quality solutions. Thus, both exploration and exploitation are maintained by the different operators and balanced until the convergence of the algorithm.

Table 5 shows the results of the memetic variants together with the results of the original metaheuristics so their values can be compared. At this stage of the analysis, the results of the baseline methods are not shown because they are significantly outperformed by the other algorithms. Similarly, the improvement of a DE-based memetic algorithm

Table 6
Average ranking of the metaheuristics including the memetic algorithms. The resulting p−values corresponds to Bonferroni's and Holm's tests using MCRO as the control method.

	Rank	Bonferroni p	Holm p
MCRO	1.5	–	–
CRO-SL	2.5	1	0.287
MIPOP	4	0.054	0.015
CRO	4.25	0.024	0.013
IPOP	4.25	0.024	0.013
MCRO-SL	4.58	0.007	0.005
L-SHADE	6.92	$5.9 \cdot 10^{-8}$	$5 \cdot 10^{-8}$
ML-SHADE	8	$3.3 \cdot 10^{-11}$	$3.3 \cdot 10^{-11}$

is not studied since it was the worst ranked evolutionary algorithm during the preliminary analysis. Besides, a more advanced DE variant, L-SHADE, is considered. Table 6 presents the corresponding average ranking and the resulting p−values of Bonferroni–Dunn's and Holm's tests using MCRO as the control method. In this case, the Friedman's test results in $\chi_F^2 = 62.94$ and a p−value of $3.8 \cdot 10^{-11}$. Once again the test concludes that there are significant differences between the algorithms' performance.

The results of the memetic variants reveal that not every hybridized algorithm improves the performance of its original counterpart since MCRO and MIPOP improve the original algorithms but ML-SHADE and MCRO-SL do not. MCRO ranks first and achieves the lowest average ranking (1.5), improving CRO in every instance and ranking first in eight instances. MIPOP obtains a small improvement compared to the original IPOP-CMA-ES performance for most instances but ranks third with an average ranking of 4, ranking first in P3.

CRO-SL is ranked second in average (2.5) and ranks first for the remaining three model instances. In addition, CRO-SL still finds the best solution for most model instances. In contrast, MCRO-SL ranks sixth (4.58) and is significantly outperformed by MCRO. MCRO-SL improves the original CRO-SL in the first instance but its performance decreases for the rest. In order to assess the performance loss of MCRO-SL, it was checked if the stopping criteria can restrict the memetic performance by rerunning the experimentation fixing the stopping criteria in 20,000 evaluations for CRO-SL and MCRO-SL. However, the results were similar to those originally delivered for 10,000 evaluations showing again that MCRO-SL cannot improve CRO-SL. Therefore, the performance reduction cannot be explained by the stopping criteria but for the role of the local search component. Since CRO-SL shows a good balance between exploration and exploitation (note that it obtains the most of the best solutions), the addition of the local search component can negatively modify this balance by reducing its diversity and causing the algorithm to converge prematurely.

With the inclusion of the memetic variants, CRO obtains the same rank as IPOP-CMA-ES and both algorithms are tied with an average rank of 4.25 in the fourth/fifth position. However, these algorithms are significantly outperformed by MCRO since the p−values of both Bonferroni and Holm tests (0.024 and 0.013, respectively) are below the considered significance threshold. L-SHADE and its memetic counterpart ML-SHADE are the worst ranked algorithms. ML-SHADE does not obtain any performance increment from its hybridization with local search and ranks last for every instance.

The impact of hybridization on the performance of these algorithms can be visually corroborated by displaying their fitness values for the instance with the highest dimensionality, P12(129). Fig. 5 shows the multiple runs of the algorithms for this instance using points: the X axis refers to the fitness values and the Y axis marks the different iterations. This plot shows how MCRO obtains the highest improvement with respect to its non-memetic counterpart. MIPOP slightly improves IPOP in many of the runs but with a reduced margin and not in every run. In addition, the loss of performance of MCRO-SL and MLSHADE is also noticeable.

Table 7
Resulting p−values corresponding to Wilcoxon ranksum test when comparing MCRO, CRO-SL, and MIPOP using pairwise comparisons.

MCRO	vs.	CRO-SL	MIPOP
p−values	–	0.018	$4 \cdot 10^{-4}$
CRO-SL	vs.	MCRO	MIPOP
p−values	–	0.984	0.016

An additional statistical test is applied to the best ranking algorithms (MCRO, CRO-SL, and MIPOP) using pairwise comparisons. Hence, the analysis performs the Wilcoxon's ranksum test (null hypothesis $R_i = R_j$, alternate hypothesis $R_i < R_j$, where R_i and R_j represent the ranking of two different algorithms) (García et al., 2008). The test is applied to every pair of the best performing metaheuristics, considering a level of significance $\alpha = 0.05$. The resulting p−values are shown in Table 7. The results of Wilcoxon's test conclude that MCRO performs significantly better than CRO-SL and MIPOP since the resulting p−values are lower than the defined significance level. In addition, CRO-SL outperforms MIPOP with statistical significance.

Finally, there are some limitations of the presented approach that need to be addressed. One of these limitations can be identified by visually analyzing the output fitting of the calibrated solutions. Fig. 6 shows the fitting for both awareness and WOM volume of the best calibrated solutions of MCRO and CRO-SL (i.e., those with the lowest fitness) for P12(129). In order to improve clarity, the KPI values refer to only two brands (6 and 8) since their behavior is a good indicator of the remaining brands. The displayed values represent the average of the multiple Monte-Carlo simulations. These values show that even when having solutions with good fitness values, the calibrated solutions are only able to capture the trend of the target values for both brands. This effect is a consequence of the objective fitting function f, which relies in the numeric deviation error computed by the loss measure but ignores the pattern characteristics of the historical values. In addition, these synthetic instances could be harder to calibrate than others based on real data as they were generated using randomness.

7. Final remarks and future work

In this paper, a new approach to automatic ABM calibration using coral reefs-based metaheuristics with an integer-coded scheme was introduced. These methods, referred as CRO and CRO-SL, exhibit competitive performance due to the emulation of the processes involving the life cycle of corals, which empower the resulting heuristic search algorithms with a strong trade-off between diversification and intensification. The performance of the coral reefs-based methods was assessed by developing an exhaustive comparison against relevant evolutionary algorithms and by checking the significance of the results with statistical tests. This analysis also considered several memetic algorithms resulting from the hybridization of the selected metaheuristics with a local search procedure. This study was carried out by applying the selected methods to the calibration of an ABM for marketing scenarios considering brand awareness and WOM volume.

The analysis concluded that both MCRO and CRO-SL performed significantly better than the other methods. Both metaheuristics consistently achieved competitive results across every model instance, including those with high dimensionality. L-SHADE and IPOP-CMA-ES achieve competitive results to the coral reefs-based metaheuristics for most model instances. However, they end up outranked in their average ranking, which may be related with the selected integer coding scheme that may have reduced their performance. With respect to the memetic algorithms, the action of the local search procedure has a different behavior depending on the original metaheuristic, since only half of the memetic algorithms improved their performance when compared with their corresponding non-memetic counterparts. MCRO-SL reduces its performance when the dimensionality of the instances increases,

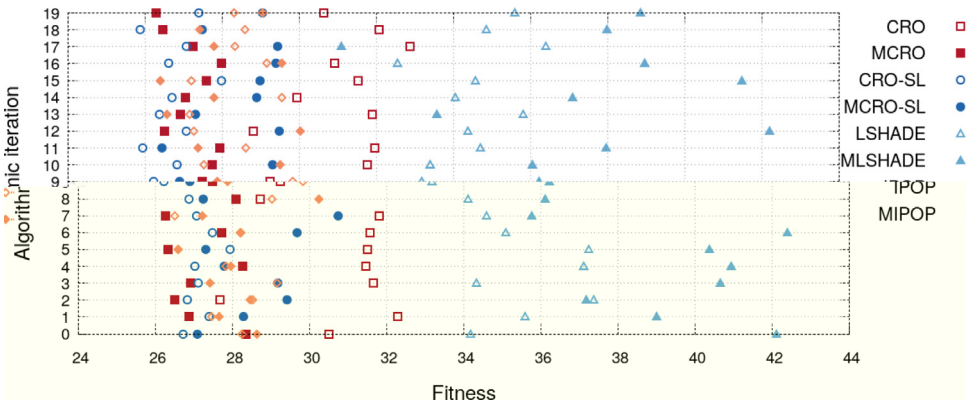
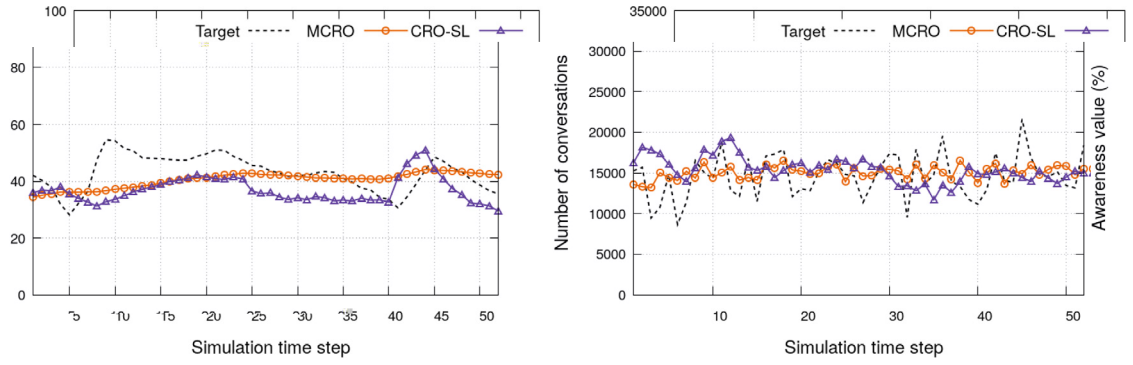


Fig. 5. Fitness results of the different runs of the eight best performing algorithms for the problem instance P12(129). The values in the Y axis refer to the different runs and the values in the X axis shows the fitness error for the problem instance.



(a) Awareness output for brand 8 (b) WOM volume output for brand 6

Fig. 6. Overtime awareness fitting for brand 8 (a) and overtime number of conversations (WOM volume) fitting for brand 6 (b) for P12(129). These values are obtained using the average overtime values of the multiple Monte-Carlo simulations. The lines with circles represent the best solution obtained by MCRO and the lines with purple triangles represent the best solution obtained by CRO-SL. The dashed lines represent the target historical values.

which suggests that the local search procedure negatively affects the good balance of CRO-SL between exploration and exploitation.

Future work will be focused on analyzing the problem of calibrating ABM from a multiobjective perspective since some ABM are calibrated considering multiple KPIs. This could also involve tackling the calibration of ABM whose search space have special properties (e.g., multimodal search space). In addition, future work can address the limitations of the current calibration approach (exposed at the end of Section 6) by including qualitative pattern features (Yücel and Barlas, 2011).

CRediT authorship contribution statement

Ignacio Moya: Software, Formal analysis, Writing - original draft. **Enrique Bermejo:** Software, Methodology, Formal analysis. **Manuel Chica:** Conceptualization, Validation, Writing - review & editing, Supervision. **Óscar Cordón:** Resources, Writing - review & editing, Supervision, Project administration, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work is supported by the Spanish Agencia Estatal de Investigación, the Andalusian Government, Spain, the University of Granada, Spain, and European Regional Development Funds (ERDF) under grants EXASOCO (PGC2018-101216-B-I00), SIMARK (P18-TP-4475), and AIMAR (A-TIC-284-UGR18). M. Chica is also supported through the Ramón y Cajal program, Spain (RYC-2016-19800). Authors thank the Centro de Servicios de Informática y Redes de Comunicaciones (CSIRC), University of Granada, for providing the computing resources.

References

Auger, A., Hansen, N., 2005. A restart CMA evolution strategy with increasing population size. In: 2005 IEEE Congress on Evolutionary Computation. pp. 1769–1776.

Back, T., Fogel, D.B., Michalewicz, Z., 1997. Handbook of Evolutionary Computation. IOP Publishing Ltd, Bristol (UK).

Barabási, A.L., Albert, R., 1999. Emergence of scaling in random networks. Science 286, 509–512.

Bermejo, E., Chica, M., Damas, S., Salcedo-Sanz, S., Cordón, O., 2018. Coral reef optimization with substrate layers for medical image registration. Swarm Evol. Comput. 42, 138–159.

Bhattacharya, M., 2013. Evolutionary approaches to expensive optimisation. Int. J. Adv. Res. Artif. Intell. 2.

Biedrzycki, R., 2017. A version of IPOP-CMA-ES algorithm with midpoint for CEC 2017 single objective bound constrained problems. In: 2017 IEEE Congress on Evolutionary Computation (CEC). pp. 1489–1494.

Biswas, A., Biswas, B., 2017. Analyzing evolutionary optimization and community detection algorithms using regression line dominance. Inform. Sci. 396, 185–201.

Branke, J., Asafuddoula, M., Bhattacharjee, K.S., Ray, T., 2017. Efficient use of partially converged simulations in evolutionary optimization. IEEE Trans. Evol. Comput. 21, 52–64.

ten Broeke, G., van Voorn, G., Ligtenberg, A., 2016. Which sensitivity analysis method should i use for my agent-based model?. J. Artif. Soc. Soc. Simul. 19, 5.

Calvez, B., Hutzler, G., 2006. Automatic tuning of agent-based models using genetic algorithms. In: Sichman, J.S., Antunes, L. (Eds.), Multi-Agent-Based Simulation VI. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 41–57.

Camacho-Gómez, C., Marsa-Maestre, I., Gimenez-Guzman, J.M., Salcedo-Sanz, S., 2019. A coral reefs optimization algorithm with substrate layer for robust wi-fi channel assignment. Soft Comput. 23, 12621–12640.

Canessa, E., Chaigneau, S., 2015. Calibrating agent-based models using a genetic algorithm. Stud. Inform. Control 24, 79–90.

Chica, M., Barranquero, J., Kajdanowicz, T., Cordón, O., Damas, S., 2017. Multimodal optimization: an effective framework for model calibration. Inform. Sci. 375, 79–97.

Chica, M., Juan, A., Bayliss, C., Cordon, O., Kelton, D., 2020. Why simheuristics? benefits, limitations, and best practices when combining metaheuristics with simulation. SORT 22, 1–35.

Chica, M., Rand, W., 2017. Building agent-based decision support systems for word-of-mouth programs: A freemium application. J. Mark. Res. 54, 752–767.

Coates, G., Li, C., Ahilan, S., Wright, N., Alharbi, M., 2019. Agent-based modeling and simulation to assess flood preparedness and recovery of manufacturing small and medium-sized enterprises. Eng. Appl. Artif. Intell. 78, 195–217.

Dai, C., Yao, M., Xie, Z., Chen, C., Liu, J., 2009. Parameter optimization for growth model of greenhouse crop using genetic algorithms. Appl. Soft Comput. 9, 13–19.

Deb, K., Agrawal, R.B., 1994. Simulated Binary Crossover for Continuous Search Space. Technical Report, Departement of Mechanical Enginering, Indian Institute of Technology, Kanpur, India.

Del Ser, J., Osaba, E., Molina, D., Yang, X.S., Salcedo-Sanz, S., Camacho, D., Das, S., Suganthan, P.N., Coello Coello, F., 2019. Bio-inspired computation: Where we stand and what's next. Swarm Evol. Comput. 48, 220–250.

Dunn, O.J., 1961. Multiple comparisons among means. J. Amer. Statist. Assoc. 56, 52–64.

Epstein, J.M., 2006. Generative Social Science: Studies in Agent-Based Computational Modeling. Princeton University Press.

Eshelman, L.J., Schaffer, J.D., 1993. Real-coded genetic algorithms and interval-schemata. In: Whitley, L.D. (Ed.), Foundations of Genetic Algorithms. vol. 2, Elsevier, pp. 187–202.

Fabretti, A., 2013. On the problem of calibrating an agent based model for financial markets. J. Econ. Interact. Coord. 8, 277–293.

Farmer, J.D., Foley, D., 2009. The economy needs agent-based modelling. Nature 460, 685–686.

Farris, P.W., Bendle, N.T., Pfeifer, P.E., Reibstein, D.J., 2010. Marketing Metrics: The Definitive Guide To Measuring Marketing Performance, second ed. Wharton School Publishing.

Floreano, D., Mattiussi, C., 2008. Bio-Inspired Artificial Intelligence: Theories, Methods, and Technologies. The MIT Press.

Friedman, M., 1940. A comparison of alternative tests of significance for the problem of m rankings. Ann. Math. Stat. 11, 86–92.

García, S., Molina, D., Lozano, M., Herrera, F., 2008. A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: a case study on the cec'2005 special session on real parameter optimization. J. Heuristics 15, 617–644.

Garcia-Hernandez, L., Garcia-Hernandez, J., Salas-Morera, L., Carmona-Muoz, C., Alghamdi, N., de Oliveira, J.V., Salcedo-Sanz, S., 2020a. Addressing unequal area facility layout problems with the coral problems with 15,617–644.

Talbi, E.G., 2009. Metaheuristics: From Design to Implementation. John Wiley & Sons.
Tanabe, R., Fukunaga, A., 2013. Success-history based parameter adaptation for differential evolution. In: 2013 IEEE Congress on Evolutionary Computation. pp. 71–78.
Tanabe, R., Fukunaga, A.S., 2014. Improving the search performance of shade using linear population size reduction. In: 2014 IEEE Congress on Evolutionary Computation (CEC). pp. 1658–1665.
Thiele, J.C., Kurth, W., Grimm, V., 2014. Facilitating parameter estimation and sensitivity analysis of agent-based models: A cookbook using netlogo and 'r'. J. Artif. Soc. Soc. Simul. 17 (11).
Trejo-Zúñiga, E.C., López Cruz, I.L., García, A.R., 2014. Parameter estimation for crop growth model using evolutionary and bio-inspired algorithms. Appl. Soft Comput. 23, 474–482.
Vermeij, M., 2005. Substrate composition and adult distribution determine recruitment patterns in a caribbean brooding coral. Mar. Ecol. Prog. Ser. 295, 123–133.
Waldrop, M.M., 2018. Free agents. Science 360, 144–147.
Watts, D.J., Strogatz, S.H., 1998. Collective dynamics of 'small-world' networks. Nature 393, 440–442.
Wilensky, U., Rand, W., 2015. Introduction to Agent-Based Modeling: Modeling Natural, Social, and Engineered Complex Systems with NetLogo. MIT Press.
Wu, F., Huberman, B.A., 2007. Novelty and collective attention. Proc. Natl. Acad. Sci. 104, 17599–17601.
Wu, G., Mallipeddi, R., Suganthan, P.N., 2019. Ensemble strategies for population-based optimization algorithms – a survey. Swarm Evol. Comput. 44, 695–711.
Yang, J., Leskovec, J., 2010. Modeling information diffusion in implicit networks. In: 2010 IEEE International Conference on Data Mining. IEEE, pp. 599–608.
Yücel, G., Barlas, Y., 2011. Automated parameter specification in dynamic feedback models based on behavior pattern features. Syst. Dyn. Rev. 27, 195–215.
Zhong, J., Cai, W., 2015. Differential evolution with sensitivity analysis and the Powell's method for crowd model calibration. J. Comput. Sci. 9, 26–32, Computational Science at the Gates of Nature.