Multi-objective calibration by combination of stochastic and gradient-like parameter generation rules: the caRamel algorithm

Céline Monteil[1], Fabrice Zaoui[1], Nicolas Le Moine[2], and Frédéric Hendrickx[1]

[1]EDF R&D LNHE - Laboratoire National d'Hydraulique et Environnement, Chatou, 78400, France
[2]UMR 7619 Metis (SU/CNRS/EPHE), Sorbonne Université, 4 Place Jussieu, Paris, 75005, France

Correspondence: Céline Monteil (celine-c.monteil@edf.fr)

Abstract. Environmental modelling is complex, and models often require calibration of several parameters not directly evaluable from a physical quantity or field measurement. Multi-objective calibration has many advantages such as add constraints in a low-constrained problem or find a compromise between different objectives by defining a set of optimal parameters. The caRamel optimizer has been developed to meet the need of an automatic calibration procedure that delivers not just one but a family of parameter sets that are optimal with regard to a multi-objective target. The idea in caRamel is to rely on stochastic rules while also allowing more "local" mechanisms, such as extrapolation along vectors in the parameter space. The caRamel algorithm is a hybrid of the Multi-objective Evolutionary Annealing Simplex method (MEAS) and the Non-dominated Sorting Genetic Algorithm II (ϵ-NSGA-II). It was initially developed for calibrating hydrological models but can be used for any environmental model. CaRamel is well adapted to complex modelling. The comparison with other optimizers on hydrological case studies (i.e. NSGA-II, MEAS) confirms the quality of the algorithm. An R package `caRamel` has been designed to easily implement this multi-objective algorithm optimizer in the R environment.

1 Introduction

Environmental modelling is complex, and models often require calibration of many parameters that cannot be directly estimated from a physical quantity or a field measurement. Moreover, as models' outputs exhibit errors whose statistical structure may be difficult to characterize precisely, it is frequently necessary to use various objectives to evaluate the modelling performance. Put differently, it is often difficult to find a rigorous likelihood function or sufficient statistics to be maximized/minimized (Fisher, 1922): for example, it is well-known that errors in a simulated discharge time series are not normally distributed, and do not have constant variance and auto-correlation (Sorooshian and Dracup, 1980). In addition, Efstratiadis and Koutsoyiannis (2010) list other advantages of multi-objective calibration such as ensuring parsimony between the number of objectives against parameters to optimize, fitting distributed responses of models on multiple measurements, recognizing the uncertainties and structural errors related to model configuration and the parameter estimation procedure, and handling objectives that have contradictory performance.

Multi-objective calibration allows for finding a compromise between these different objectives by defining a set of optimal parameters. Evolutionary algorithms have become widely used to explore the Pareto-optimal front in multi-objective optimization problems too complex to be solved by descent methods with classical aggregation approaches. The advantage of these evolutionary algorithms lies not only in a dearth of alternatives for searching substantially large spaces for multiple Pareto-optimal solutions. Their inherent parallelism and capability to exploit similarities of solutions by recombination, also enables them to approximate the Pareto-optimal front in a single optimization run (Zitzler et al., 2000). Many studies in environmental modelling (Madsen, 2003; Oraei Zare et al., 2012; Ercan and Goodall, 2016; Smith et al., 2019) or land-use models (Gong et al., 2015; Newland et al., 2018) use the multi-objective approach.

The caRamel optimizer has been developed to meet the need of an automatic calibration procedure that delivers not only one, but a family of parameter sets that are optimal with regard to a multi-objective target. Most of multi-objective algorithms rely mainly on stochastic generation rules, with few deterministic aspects. The idea behind caRamel is not just to keep these stochastic "global" mechanisms (such as recombination or multivariate sampling using the covariance) but also to allow more "local" mechanisms, such as extrapolation along vectors in the parameter space that are associated with an improvement in all objective functions (a "gradient-like" qualitative approach extended to the set of objective functions).

CaRamel was initially developed and used for the calibration of hydrological models: Rothfuss et al., 2012; Magand et al., 2014; Le Moine et al., 2015; Monteil et al., 2015 (all previous to the R package release) or Rouhier et al. (2017, R version, calibration of a hydrologic model over the Loire basin, 35,707 km^2). The interesting performances of the caRamel algorithm in such studies prompted us to describe in detail the algorithm in the present paper. Considering the increasing use of R in hydrology (Slater et al., 2019), we decided to build a R package, `caRamel`, for use in any model in the R environment. The user has simply to define a vector-valued function (at least 2 objectives) for the model to calibrate as well as lower and upper bounds for the calibrated parameters.

This paper aims to describe the principles of the caRamel algorithm, through analysis of its results when used for parametrization of hydrological models. For each case study, a comparison with the two calibration algorithms that have inspired caRamel (the Non-dominated Sorting Genetic Algorithm II, NSGA-II, Reed and Devireddy, 2004, and the Multi-objective Evolutionary Annealing Simplex method, MEAS, Efstratiadis and Koutsoyiannis, 2008) is also presented.

2 Context and notations

The intent of multi-objective calibration is to find sets of parameters that provide a compromise between several potentially conflicting objectives; for instance, how to achieve a good simulation of both flood and low-flow in a hydrological model. Multi-objective calibration is also a means of adding some constraints to an under-constrained problem when many parameters have to be quantified. This can help to reduce the equifinality of parameter sets. Her and Seong (2018) showed that the introduction of an adequate number of objective functions could improve the quality of calibration without requiring additional observations. The amount of equifinality and output uncertainty overall decreased while the model performance was maintained as the number of objective functions increased sequentially until four objective functions.

To introduce our notation, Figure 1 shows a simplified calibration problem in which there is:

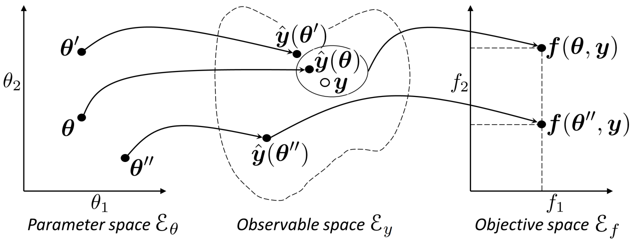

$\boldsymbol{\theta}'$
θ_2
$\boldsymbol{\theta}$
$\boldsymbol{\theta}''$
θ_1
Parameter space $\mathcal{E}_\theta$
$\hat{\boldsymbol{y}}(\boldsymbol{\theta}')$
$\hat{\boldsymbol{y}}(\boldsymbol{\theta})$
$\boldsymbol{y}$
$\hat{\boldsymbol{y}}(\boldsymbol{\theta}'')$
Observable space $\mathcal{E}_y$
$\boldsymbol{f}(\boldsymbol{\theta},\boldsymbol{y})$
f_2
$\boldsymbol{f}(\boldsymbol{\theta}'',\boldsymbol{y})$
f_1
Objective space $\mathcal{E}_f$

Figure 1. Notations to describe a model calibration: with $\boldsymbol{\theta}$ a vector from the parameter space $\mathcal{E}_\theta$, $\boldsymbol{y}$ a vector of observed values in the observable space $\mathcal{E}_y$ and $\boldsymbol{f}(\boldsymbol{\theta},\boldsymbol{y})$ an objective vector in the objective space $\mathcal{E}_f$.

- a model with $n_\theta = 2$ parameters to calibrate (θ_1 and θ_2). The model structure is thus unequivocally represented by the vector $\boldsymbol{\theta} = (\theta_1, \theta_2)$ in a $n_\theta = 2$ dimensional space, called *parameter space* $\mathcal{E}_\theta$.

- a vector $\boldsymbol{y}$ of n_y observed values that should be simulated by the model. For example, for daily times series of 1 year at 2 gauging stations, $n_y = 2 * 365 = 730$. The simulation is represented by a vector $\hat{\boldsymbol{y}}(\boldsymbol{\theta})$ in a n_y dimensional space (that cannot be illustrated graphically), called *observable space* $\mathcal{E}_y$.

- a vector of n_f objective values $\boldsymbol{f}(\boldsymbol{\theta},\boldsymbol{y})$. For the example in Fig. 1, $\boldsymbol{f} = (f_1, f_2)$ in a space with n_f dimensions, called *objective space* $\mathcal{E}_f$.

We will use the following notations: vector or matrix written in bold ($\boldsymbol{\theta}$, $\boldsymbol{y}$, $\boldsymbol{f}$, $\boldsymbol{\Sigma}$ ...), vector element and scalar written normally (θ_1, θ_2, λ, ...), space or ensemble written in cursive ($\mathcal{E}_\theta$, $\mathcal{F}$, $\mathcal{A}$, ...).

Figure 1 also illustrates the relevance of multi-objective calibration with regard to two kinds of equifinality:

1. equifinality of structure: the two points $\boldsymbol{\theta}$ and $\boldsymbol{\theta}'$ quite distant in the parameter space $\mathcal{E}_\theta$ become quite near in the observation space $\mathcal{E}_y$.

2. equifinality related to the objective: the vectors $\boldsymbol{\theta}$ et $\boldsymbol{\theta}''$ are equifinal regarding f_1, and the additional objective f_2 help to discriminate them. The use of additional objectives may then help to better constrain the calibration.

The purpose of a multi-objective algorithm is to approach the Pareto front, $\mathcal{F}$, of non-dominated solution in the objective space by an ensemble of points called the approximated Pareto front $\hat{\mathcal{F}}$. We call *archive* $\hat{\mathcal{A}}$ the ensemble of parameter sets from $\mathcal{E}_\theta$ for which simulation outputs are in $\hat{\mathcal{F}}$.

3 CaRamel algorithm description

The caRamel algorithm belongs to the genetic algorithm family. The idea is to start from an ensemble of parameter sets (called "population") and to make this population evolve following some generation rules (Fig. 2). At each generation, new sets are evaluated regarding the objectives and only the more "suitable" sets are kept to build the new population. The caRamel algorithm is largely inspired by:

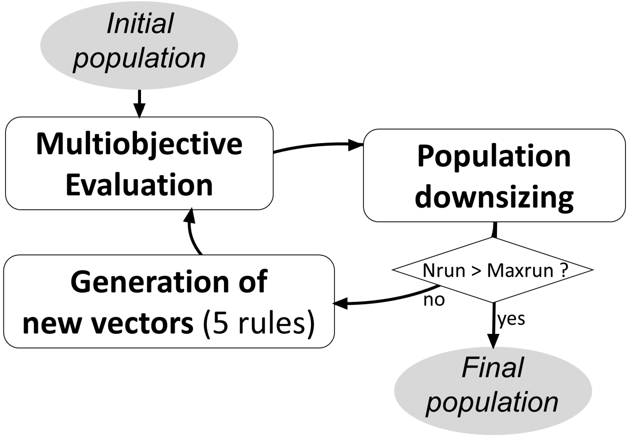


Figure 2. Flowchart of caRamel algorithm.

1. the Multi-objective Evolutionary Annealing Simplex method (MEAS, Efstratiadis and Koutsoyiannis, 2005; Efstratiadis and Koutsoyiannis, 2008), for the directional search method, based on the simplexes of the objectives space,

2. the Nondominated Sorting Genetic Algorithm II (ϵ-NSGA-II, Reed and Devireddy, 2004), for the classification of parameter vectors and the management of precision by ϵ-dominance.

This section describes the functioning of caRamel algorithm. This algorithm has been implemented in a R package `caRamel` that is described in Appendix A.

3.1 Generation rules

The caRamel algorithm has five rules for generating new solutions at each generation: (1) interpolation, (2) extrapolation, (3) independent sampling with a priori parameter variance, (4) sampling with respect to a correlation structure, and (5) recombination.

The first two rules (interpolation, extrapolation) are based on a n_θ-dimensional Delaunay triangulation in the objectives space $\mathcal{E}_f$. They assume that two neighboring points in the objectives space $\mathcal{E}_f$ have two adjacent points in the parameter space $\mathcal{E}_\theta$ as antecedents, and therefore one can try to "guess" the directions of improvement in the parameter space from the improvement directions (in a Pareto sense) in the objective space, at least near the optimal zone.

The following two rules create new parameter sets by exploring the parameter space in a non-directional and less local way: either by independent variations in each parameter, or by multivariate sampling using the covariance structure of all parameter sets located near the estimated Pareto front at the current iteration.

Finally, the recombination rule consists in creating new parameter sets using two partial subsets derived from a pair of previously evaluated parameter sets (inspired by Baluja and Caruana, 1995).

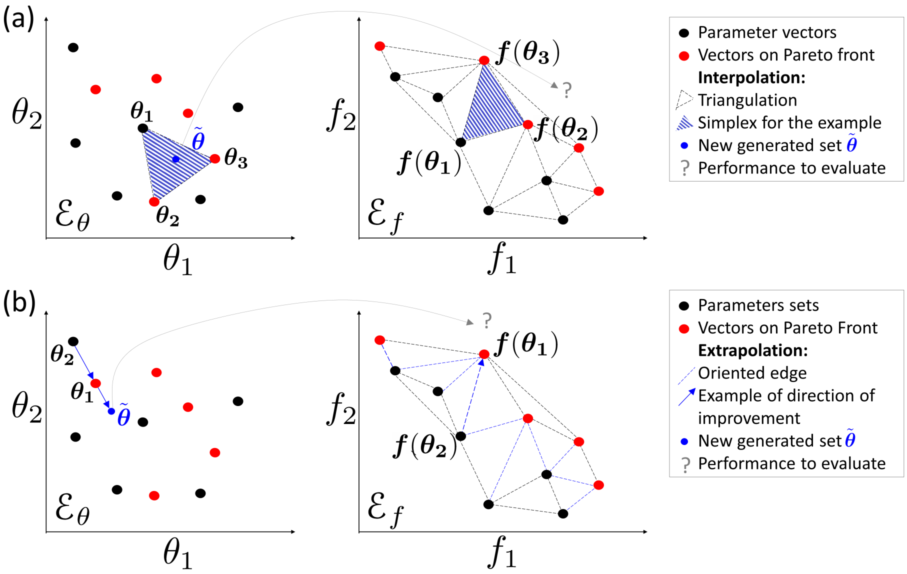

(a)
θ_2
$\boldsymbol{\theta_1}$
$\tilde{\boldsymbol{\theta}}$
$\boldsymbol{\theta_3}$
$\boldsymbol{\theta_2}$
$\mathcal{E}_\theta$
θ_1
f_2
$\boldsymbol{f}(\boldsymbol{\theta_3})$
?
$\boldsymbol{f}(\boldsymbol{\theta_2})$
$\boldsymbol{f}(\boldsymbol{\theta_1})$
$\mathcal{E}_f$
f_1
● Parameter vectors
● Vectors on Pareto front
Interpolation:
Triangulation
Simplex for the example
● New generated set $\tilde{\boldsymbol{\theta}}$
? Performance to evaluate
(b)
$\boldsymbol{\theta_2}$
$\boldsymbol{\theta_1}$
$\tilde{\boldsymbol{\theta}}$
θ_2
$\mathcal{E}_\theta$
θ_1
?
$\boldsymbol{f}(\boldsymbol{\theta_1})$
f_2
$\boldsymbol{f}(\boldsymbol{\theta_2})$
$\mathcal{E}_f$
f_1
● Parameters sets
● Vectors on Pareto Front
Extrapolation:
Oriented edge
Example of direction of improvement
● New generated set $\tilde{\boldsymbol{\theta}}$
? Performance to evaluate

Figure 3. Illustration of rules 1 and 2 based on a Delaunay triangulation in the objectives space for a maximization problem with 2 parameters (θ_1 and θ_2) and 2 objectives (f_1 and f_2): (a) Interpolation computes a new parameters vector for each simplex with a non-dominated vertex; (b) Extrapolation derives a new vector for each direction of improvement.

3.1.1 Rule 1: Interpolation

The explanation of this rule is based on Fig. 3(a). First a triangulation of the points in the objective space $\mathcal{E}_f$ is established: simplexes built with these points $\boldsymbol{f}(\boldsymbol{\theta_i})$ are a partition of the explored zone in this space (Efstratiadis and Koutsoyiannis, 2005).

Let us consider a simplex with at least one vertex on the approximated Pareto front. This simplex is the result of the function $\boldsymbol{f}$ from an ensemble of (n_f+1) points from the n_θ-dimensional parameter space $\mathcal{E}_\theta$. Under the hypothesis of continuity of $\boldsymbol{f}$, a linear combination of the form $\tilde{\boldsymbol{\theta}} = w_1\boldsymbol{\theta_1} + \ldots + w_{(n_f+1)}\boldsymbol{\theta_{(n_f+1)}}$, with the barycentric coordinates $w_i \geq 0$ and $\sum_i w_i = 1$, might give a new Pareto-optimal solution $\boldsymbol{f}(\tilde{\boldsymbol{\theta}})$ inside this zone.

First the triangulation is established, then simplex volumes are computed. The probability of generating one new point with a simplex is proportional to its volume when it has at least one point on the Pareto front (0 otherwise). If the simplex is selected, then a set of barycentric coordinates are computed by randomly generating (n_f+1) values ε_i in a uniform distribution on [0,1] (Eq. 1).

$$w_i = \frac{\varepsilon_i}{\sum_{j=1}^{(n_f+1)} \varepsilon_j} \tag{1}$$

3.1.2 Rule 2: Extrapolation

Extrapolation is based on the same hypothesis of continuity as interpolation. In this case, it is tested to find if an improvement may be obtained by extrapolating from some directions. These directions are computed from the triangulation by selecting the edges that have only one vertex on to the approximated Pareto front (the second vertex is dominated by the first). These oriented edges computed from the objective space represent directions of improvement in the parameter space (Fig. 3(b)).

The length $L = \|\boldsymbol{f}(\boldsymbol{\theta_1}) - \boldsymbol{f}(\boldsymbol{\theta_2})\|$ of each selected edge and the mean length $\overline{L}$ are computed. The probability of using an edge is proportional to its length L. In this case, the research vector in the parameter space is defined in Eq. (2) and a new parameter set is generate by $\tilde{\boldsymbol{\theta}} = \boldsymbol{\theta_1} + \lambda \boldsymbol{U}$, where λ is a scalar from an exponential distribution with average of 1.

$$\boldsymbol{U} = \frac{L}{\overline{L}} (\boldsymbol{\theta_1} - \boldsymbol{\theta_2}) \tag{2}$$

3.1.3 Rule 3: independent sampling with a priori parameter variance

The drawback of the first two rules is that the generation of new vectors is only based on a small number of existing vectors. To compensate this search by gradient and avoid convergence toward a secondary optimum, this third generation rule has two goals:

- To make the variance of parameters larger than with local rules,
- To make the variance of parameters independent from each other.

When considering a vector $\boldsymbol{\theta}$ from the archive $\hat{\mathcal{A}}$, the third rule is to generate n_θ new vectors ($\tilde{\boldsymbol{\theta}}_{\boldsymbol{k}}$ with k from 1 to n_θ) by making each element of $\boldsymbol{\theta}$ (Eq. (3) vary individually with σ_i^2 the a priori variance of the i-th parameter, and ε_i a value from a normal distribution of average 0 and variance 1).

$$\forall i \in [1:n_\theta]_{i \neq k} \quad \tilde{\theta}_{ki} = \theta_i \quad ; \quad \text{if} \;\; i = k \quad \tilde{\theta}_{ki} = \theta_{ki} + \sigma_i \varepsilon_i \tag{3}$$

The algorithm selects the n_θ vectors that maximize individually each element of the objective vector and an additional vector that represents a "central" point of the Pareto front. To select this vector, the minimum of each vector $\theta \in \hat{\mathcal{A}}$ is computed and the vector that maximizes this value is chosen.

One generation of this rule is then generating $(n_f + 1) \times n_\theta$ new vectors. For this reason, this rule is applied every K generation, with K to be defined by the user. By default, K is computed so that each rule generates in average the same number of vectors.

3.1.4 Rule 4: sampling with respect to a correlation structure

The variance-covariance matrix $\boldsymbol{\Sigma}$ is computed by Eq. (4) where $\mathbb{E}[X]$ is the expectancy of a random variable X, $\boldsymbol{\theta}$ is a vector from the archive $\mathcal{A}$, $\boldsymbol{\mu} = \mathbb{E}_{\boldsymbol{\theta} \in \mathcal{A}}[\boldsymbol{\theta}]$ is the barycenter of $\mathcal{A}$, and ${}^t\boldsymbol{M}$ is the transpose of the matrix $\boldsymbol{M}$.

$$\boldsymbol{\Sigma} = \mathbb{E}_{\boldsymbol{\theta} \in \mathcal{A}} \left[(\boldsymbol{\theta} - \boldsymbol{\mu}) \, {}^t(\boldsymbol{\theta} - \boldsymbol{\mu}) \right] \tag{4}$$

This matrix reflects the correlation structure between the parameter sets. For instance in the case of a hydrological model, parameters are frequently not independent of each other. This rule intends to obtain an estimate $\hat{\boldsymbol{\Sigma}}$ of $\boldsymbol{\Sigma}$ and $\hat{\boldsymbol{\mu}}$ of $\boldsymbol{\mu}$ in order to generate new parameter vectors that respect this correlation structure and so limit the risk of generating "non-functional" parameter sets.

There are many possibilities in selecting the vector for evaluating the covariance matrix:

1. Having a library of "historical" vectors for the calibrated model. The drawback is that this library has to be previously established and it does not take into account progression of the running calibration.

2. Selecting vectors from the archive $\hat{\mathcal{A}}$ that give points on the approximated Pareto front at the running generation. The new vectors are frequently improving the front, but as the variance is low, they do not allow getting out of a local optimum.

3. Selecting all vectors of the running population. It helps to keep a diversity but has a high computational cost as few new vectors will make the front to progress.

Finally, the algorithm uses a mix between item 2 and 3: all simplexes from the first rule triangulation that have at least a vertex in the approximated Pareto front are selected. Reference vectors for computation of the variance-covariance matrix are defined by the ensemble $\mathcal{G}$ from the objective space whose images by $\boldsymbol{f}$ are all the vertices of these simplexes. The estimates $\hat{\boldsymbol{\Sigma}}$ and $\hat{\boldsymbol{\mu}}$ are computed in Eq. (5-6).

$$\hat{\boldsymbol{\mu}} = \mathbb{E}_{\boldsymbol{\theta} \in \mathcal{G}} \left[\boldsymbol{\theta} \right] \tag{5}$$

$$\hat{\boldsymbol{\Sigma}} = \mathbb{E}_{\boldsymbol{\theta} \in \mathcal{G}} \left[(\boldsymbol{\theta} - \hat{\boldsymbol{\mu}})\ {}^{t}(\boldsymbol{\theta} - \hat{\boldsymbol{\mu}}) \right] \tag{6}$$

This operation increases significantly the number of selected points for the averages computation. However, the risk is still of having too low a variance. To reduce this risk, the variance of all the parameters is increased by the same factor (empirically doubled): $\hat{\hat{\boldsymbol{\Sigma}}} = 2\hat{\boldsymbol{\Sigma}}$.

The new vectors are obtained from a classical procedure for multivariate generation:

1. computation of the upper triangular matrix $\boldsymbol{T}$ with ${}^{t}\boldsymbol{T}\boldsymbol{T} = \hat{\hat{\boldsymbol{\Sigma}}}$, by Cholesky decomposition;

2. generation of vectors $\tilde{\boldsymbol{\theta}} = \hat{\boldsymbol{\mu}} + {}^{t}\boldsymbol{T} \cdot \boldsymbol{\varepsilon}$, where $\boldsymbol{\varepsilon}$ is a vector with n_θ independent and normally distributed components with average 0 and variance 1.

This fourth rule enables us to randomly explore some area of space $\mathcal{E}_\theta$ while implicitly reducing its dimension through the correlations between parameters. It reduces the number of evaluations needed of the objective function.

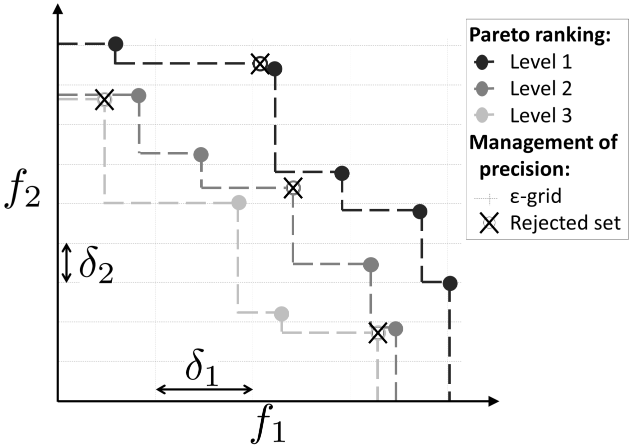


Figure 4. Method for population downsizing for a maximization problem with 2 objectives: Pareto ranking (Level 1 is the current approximated Pareto front) and partition of the objective space according to the chosen δ_i precision (only one vector by hypercube is kept).

3.1.5 Rule 5: Recombination

As for rule 4, recombination considers that the parameters from a model are not independent. In a hydrological model, they can frequently be grouped in functional blocks (for instance rapid runoff, base flow, snow dynamics, transfer...). A new parameter vector is simply generated by combining blocs of parameters from vectors of the archive $\hat{\mathcal{A}}$. The parameter blocks are specific to the calibrated model and are defined by the user.

3.2 Population downsizing

At the end of each generation, population is kept under a maximum size (N_{max} sets). This limitation is set for memory reasons (no need to keep poor parameter sets) and for computational time as the triangulation computation is done at each generation.

The population downsizing is adapted from ϵ-NSGA-II (Reed and Devireddy, 2004) and is performed in 3 steps (Fig. 4):

1. Pareto ranking: the parameter vectors are sorted according to ranking order of the Pareto level to which they belong. Points from level 1 are non-dominated, points from level 2 are dominated only by points from level 1, and so on ...

2. Downsizing according to the chosen precision: the objective space is partitioned by a n_f-dimensional grid with the precision δ_i for each of the n_f objective values. All the points in the same hypercube are considered as equifinal with regard to accuracy, and only one point is kept. The selected point is the one which belongs to the lowest Pareto level. When many points are the lowest level, the selected point is taken at random from among them.

3. Keeping the population size under N_{max}: if the number of sets is still above N_{max}, keeping only the N_{max} sets of the smaller level.

4 Optimization evaluation framework

The aim is to compare the performance of the caRamel algorithm to other optimizers on various case studies. Two optimizers have been selected for the comparison: NSGA-II (Deb et al., 2002) and MEAS (Efstratiadis and Koutsoyiannis, 2008). The comparison focuses on different aspects: optimization evolution evaluated by specific metrics, optimization results in the objective space, parameters space, and observable space. This section presents the optimizer configuration, the evaluation metrics and the four case studies.

4.1 Optimizer configurations

CaRamel is used in its general form, with a generation of five new parameters sets for each rule by iteration, involving an average of 25 parameter sets by generation.

NSGA-II (Deb et al., 2002) is called by using the function nsga2 from the R package `mco` "Multiple Criteria Optimization", Mersmann et al., 2014). The arguments are the function to minimize, the input and output dimensions, the parameter bounds, the number of generations, the size of the population and the values for crossover, mutation probability and distribution index. Some previous calibration experiments have been conducted to determine the best parameters configuration. NSGA-II has been used with crossover probability set to 0.5 and mutation probability to 0.3.

The MEAS algorithm (Efstratiadis and Koutsoyiannis, 2005) combines a performance evaluation procedure based on a Pareto approach and concept of feasibility, an evolving pattern based on the downhill simplex method, and a simulated annealing strategy, to control randomness during evolution. The algorithm evolution is sensitive to the value of mutation probability which has been adapted to each case study according to its complexity (5% for Kursawe, 50% for the other case studies).

For each optimizer, the end of one optimization is set to a maximum number of model evaluations depending on the case studies. As the algorithms use random functions, 40 optimizations of each test case have been run for each optimizer to obtain representative results. In order to focus on the evolution of the optimization, the initial population is the same for each optimizer (40 initial populations for each case study).

4.2 Optimization metrics

To evaluate the optimizer performances, we chose metrics from the literature. Evaluating optimization techniques experimentally always involves the notion of performance. In the case of multi-objective optimization, the definition of quality is substantially more complex than for single-objective optimization problems, because the optimization goal itself consists of multiple objectives (Zitzler et al., 2000). Riquelme et al. (2015) categorize the metrics to evaluate three main aspects:

- The accuracy, which is the closeness of the solutions to the theoretical Pareto front (if known) or relative closeness;
- The diversity, which can be described with two aspects: the spread of the set (range of values covered by the solutions) and the distribution (relative distance among solutions in the set);
- The cardinality, which qualifies the number of Pareto-optimal solutions in the set.

To quantify these aspects, we selected three different metrics that are evaluated in the objective space:

- Hypervolume (HV), which is a volume-based index that takes into account accuracy, diversity and cardinality (Zitzler and Thiele, 1999), Hypervolume computes the volume between the vectors of the estimated Pareto front $\hat{\mathcal{F}}$ and a reference point;
- Generational Distance (GD), which is a distance based accuracy performance index (Van Veldhuizen (1999), Eq. 7);

  $$GD = \frac{(\sum_{i=1}^{n} d_i^2)^{1/2}}{n} \tag{7}$$

  where n is the number of vectors in the approximated Pareto front $\hat{\mathcal{F}}$ and d_i is the Euclidean distance between each vector and the nearest member of the reference front.
- Generalized Spread (GS), which evaluates the diversity of the set (Zhou et al., 2006; Jiang et al., 2014).

The evaluation of metrics GS and GD requires us to establish a reference front. For each case study, this reference front is built by evaluating the Pareto front on all the final optimization results of all optimizers.

4.3 Case studies

Four case studies have been designed to have an increasing complexity. (1) is an analytical example with a Kursawe test function (Kursawe, 1991); (2) is a case study on a fictive catchment with a GR4J open source hydrological model (Coron et al., 2017, 2019); (3) is a case study on a pluvial catchment with a MORDOR-TS semi-distributed model (Rouhier et al., 2017); (4) is a case study on a snowy catchment, also with a MORDOR-TS model.

4.3.1 Kursawe test function

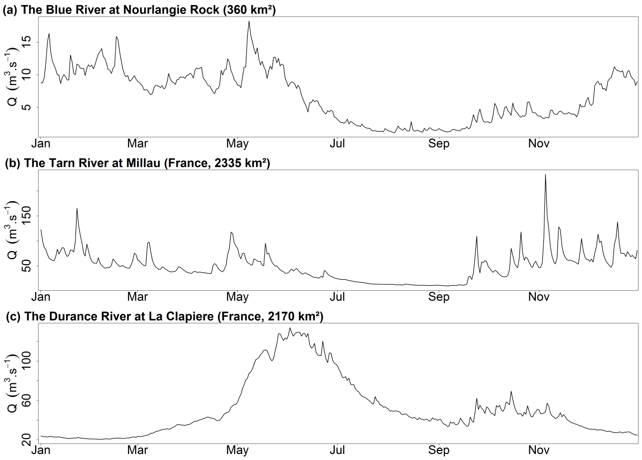
The objective of a test function is to evaluate some characteristics of optimization algorithms. The final Pareto front has a specific shape (non-convex) that the optimizer has to accurately reproduce. The Kursawe test function has three parameters (x_1, x_2, x_3) and two objectives (Obj_1, Obj_2) to minimize (Kursawe (1991), Eq. 8).

$$\begin{cases} Obj_1 = -10 \cdot (e^{-0.2\sqrt{x_1^2+x_2^2}} - e^{-0.2\sqrt{x_2^2+x_3^2}}) \\ Obj_2 = |x_1|^{0.8} + 5 \cdot sin(x_1^3) + |x_2|^{0.8} + 5 \cdot sin(x_2^3) + |x_3|^{0.8} + 5 \cdot sin(x_3^3) \end{cases} \tag{8}$$

The optimizations are run on 50,000 model evaluations. The R script to run the Kursawe function optimization with caRamel is available in Appendix B, or as a vignette in the `caRamel` package.

4.3.2 Calibration of GR4J model on a fictive catchment

The hydrological model GR4J is a widely used global rainfall-runoff model (Perrin et al., 2003) that has been implemented in an open-source R package `airGR` (Coron et al., 2017, 2019). This package contains a data sample from a fictive catchment


(a) The Blue River at Nourlangie Rock (360 km²)

(b) The Tarn River at Millau (France, 2335 km²)

(c) The Durance River at La Clapiere (France, 2170 km²)

Figure 5. Daily discharge regimes at the 3 studied catchments.

called "Blue River at Nourlangie Rock" (360 km^2, code L0123001), which has a pluvial regime (Fig. 5a). The advantage of using this case study is in having an open-source script with open-data.

GR4J has four parameters to calibrate: the production store capacity $X1$; the inter-catchment exchange coefficient $X2$; the routing store capacity $X3$; and the unit hydrograph time constant $X4$.

The calibration is done on the daily time series for the period 1990-1999. The Kling-Gupta Efficiency (KGE, Gupta et al., 2009) is frequently used in hydrology. KGE can be split into three components that reflects the correlation between the simulated and observed values (KGE_r), the bias in standard deviation (KGE_α), and the bias in volume (KGE_β). The calibration is done on these 3 components (Eq.9).

$$\begin{cases} KGE_r = 1 - \sqrt{(1-r)^2} \\ KGE_\alpha = 1 - \sqrt{(1-\alpha)^2}, \quad \text{with } \alpha = \sigma_s/\sigma_o \\ KGE_\beta = 1 - \sqrt{(1-\beta)^2}, \quad \text{with } \beta = \mu_s/\mu_o \end{cases} \tag{9}$$

where r is the linear correlation coefficient between simulated and observed time series, σ_s and σ_o represent their standard deviations, and μ_s and μ_o their mean values.

For each component, the optimal value is 1 and the optimization consists in a maximization. At the end of the optimization only the sets with $KGE_\beta > 0$ are considered, as a KGE_β with negative value indicates poor quality for hydrological results. This leads us to exclude a few sets for calibration with NSGA-II and caRamel but not for MEAS.

The R script to run an optimization of GR4J model with caRamel is available in Appendix C.

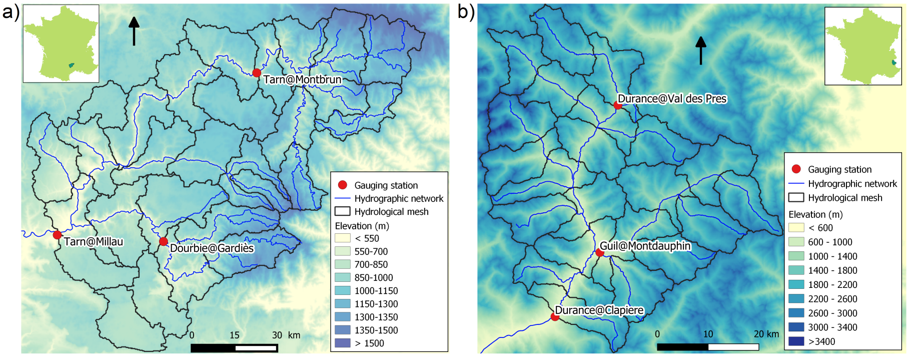


Figure 6. Maps of the studied catchments: a) Tarn at Millau (2 335 km^2) ; b) Durance at La Clapière (2 170 km^2)

4.3.3 Calibration of MORDOR-TS model on two contrasted catchments

The spatially distributed rainfall-runoff MORDOR-TS model (Rouhier et al., 2017) is a spatialized version of the conceptual MORDOR-SD model (Garavaglia et al., 2017) widely used for operational applications at Électricité de France (EDF, the French electric utility company). The catchment is divided into elementary sub-catchments connected according to the hydrographic network which constitutes a hydrological mesh.

This model was implemented at a daily time step for two French catchments with contrasted climates. The Tarn catchment at Millau (Fig. 6a) covers an area of 2,335 km^2, with middle altitude (350 to 1,600 m). The regime is pluvial, with almost no influence of snow. The Durance at the La Clapière catchment (2,170 km^2, Fig. 6b) is located in the French Alps, with elevations ranging from 800 m to about 4000 m. Its hydrological regime is strongly influenced by the snow with a maximum during the melting season in June (Fig. 5c).

The hydrological meshes have been built with an average cell area of 100 km^2, meaning 28 cells for the Tarn catchment and 22 cells for the Durance catchment.

MORDOR-TS has 22 free parameters in its comprehensive formulation. For the Tarn case study, a simplified formulation is adopted with 12 free parameters to calibrate in order to describe the functioning of conceptual reservoirs, evapotranspiration correction and wave celerity (Table 1). For the Durance catchment, parametrization of the snow module of MORDOR-TS is more complex and 16 parameters are to be calibrated for the hydrological model. The parameter distribution is uniform for the two case studies, which means that the same set of parameters applies to all cells. Calibration is conducted over 10 years (01/01/1991–31/12/2000) based on three objectives that have to be maximized. The theoretical optimum is the point (1, 1, 1) in the objectives space.

For the Tarn catchment, the calibration is based on the Nash-Sutcliffe efficiencies NS (Nash and Sutcliffe, 1970) at three gauging stations: the catchment outlet (Tarn at Millau), and two interior points (Tarn at Montbrun and Dourbie at Gardiès). For the Durance catchment, the Kling-Gupta efficiency KGE (Gupta et al., 2009) is computed at three gauging stations: the catchment outlet (Durance at La Clapière), and two interior points (Durance at Val-des-Prés and Guil at Montdauphin).

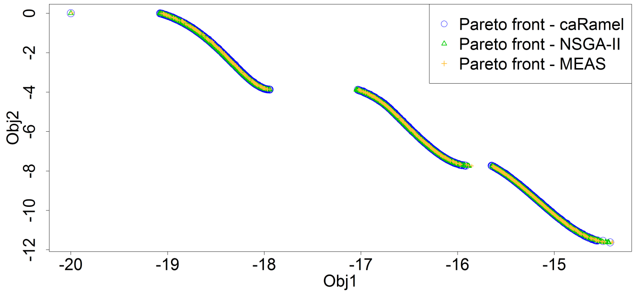


Figure 7. Pareto front after 50,000 model evaluations with caRamel (1,183 points), NSGA-II (1,780 points) or MEAS (687 points) for the Kursawe test function.

5 Results of calibration evaluations

Four aspects are considered for the results of the case studies: the shape of the final Pareto fronts; the dynamics of the optimizations; the distribution of the calibrated parameters, and their consequences on simulated stream flow for the hydrological case studies. To illustrate the results on the simulated stream flow, a "best-compromise set" has been selected regarding to the distance to the point (1,1,1) in the objective space for each hydrological case studies.

5.1 Final Pareto front

First of all, it is important to accurately reproduce the shape of the Pareto front for the Kursawe test function, and this is the case for all the optimizers (Fig. 7).

For the 3 hydrological case studies, the Pareto fronts look quite similar for caRamel and NSGA-II and more narrow with MEAS (Fig. 8) The number of sets on the Pareto front change depending on the case and there is no rank for the optimizers. For the Blue River study, there are 1,172 sets with caRamel, 878 sets with NSGA-II, and 268 points with MEAS. Then there are 1,457, 789 and 1,882 sets for the Tarn study, and 708, 408, and 525 sets for the Durance study with caRamel, NSGA-II and MEAS respectively.

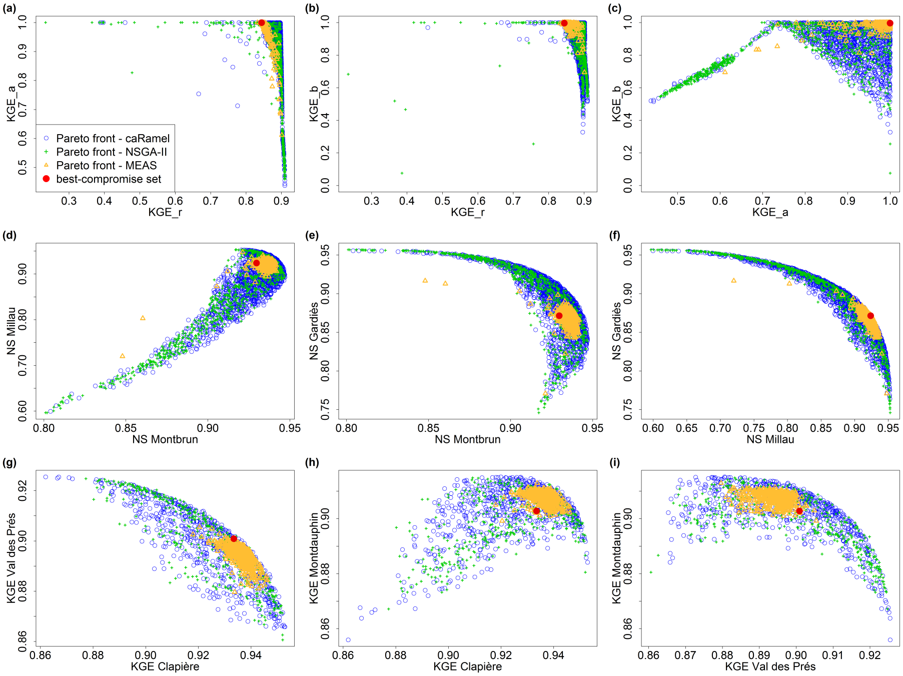


Figure 8. Pareto fronts over 40 optimizations with optimizers caRamel, NSGA-II and MEAS for each hydrological case study: Blue River with GR4J (a-c), Tarn with MORDOR-TS (d-f) and Durance with MORDOR-TS (g-i). The red point represents a "best compromise" set that is used to illustrate model results.

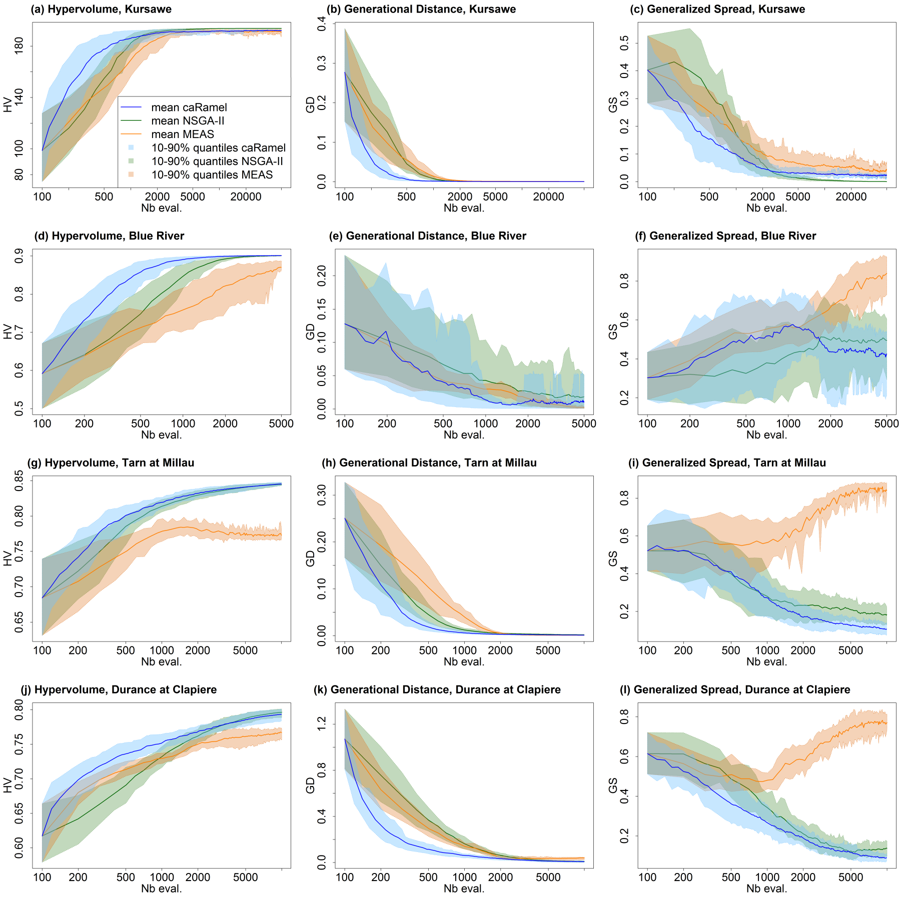


Figure 9. Metrics evolution over 40 optimizations with caRamel, NSGA-II and MEAS: mean evolution and 10-90% quantiles of the metrics regarding the number of model evaluations: (a-c) metrics for Kursawe test function; (d-f) metrics for GR4J calibration of at Nourlangie Rock; (g-i) metrics for MORDOR-TS calibration of Tarn at Millau; (j-l) metrics for MORDOR-TS calibration of Durance at La Clapière.

5.2 Dynamics of the optimizations

Figure 9 summarizes the dynamics of the optimizations for the four case studies.

CaRamel is converging more quickly for accuracy (metrics HV and GD in most of the cases). CaRamel dynamics is closer to NSGA-II dynamics than to MEAS as they have almost the same final values for the three metrics.

GS dynamics is different for the Kursawe test case than for the hydrological case studies. For the Kursawe test case, the optimal final front has a spread, so all optimizers give the same results. For the hydrological cases, the optimal solution is a point (1,1,1) and so the Pareto front may get smaller with the optimization. NSGA-II and caRamel are generating more diversity than MEAS (GS final values).

The envelopes over 40 optimizations are comparable for the three optimizers, meaning reproducibility is similar.

5.3 Parameter distribution

Figure 10 displays the distribution of parameters from the tree case studies.

In the parameter space, the optimizers provide very similar results that explore the equifinality of the model, meaning that different parameter sets give similar performances (Fig. 10). Some parameters (such as kr or lkn) may have optimized values on the whole range defined by the bounds, while other parameters are better constrained ($X1$, cel). These constitute a family of sets that are optimal with regard to the chosen objectives.

The difference in the diversity of the final sets is also visible in the parameter distributions. Distributions are quite similar for caRamel and NSGA-II but much narrower for MEAS.

5.4 Impact on model results

Consequences on the simulated streamflow are displayed on Fig. 11. The envelopes with NSGA-II and caRamel are quite similar, whereas the envelope with MEAS is narrower. It confirms that caRamel and NSGA-II are generating more diversity on their Pareto front. The red line represents the simulated streamflow with the "best-compromise" set and fits quite well with the observed one. Multi-objective calibration allows having a range of variation of calibrated discharges around the best-compromise simulation.

Figure 11c) represents a flood event on the Tarn River at Millau. The observed streamflow points are in the envelope of simulation. The best-compromised simulation does not accurately reproduce the flood peak. The figure also displays the simulated discharges obtained by optimizing parameters on the 3 gauging stations separately, and the simulation with the set that optimizes NS at Millau fits better with the observed points.

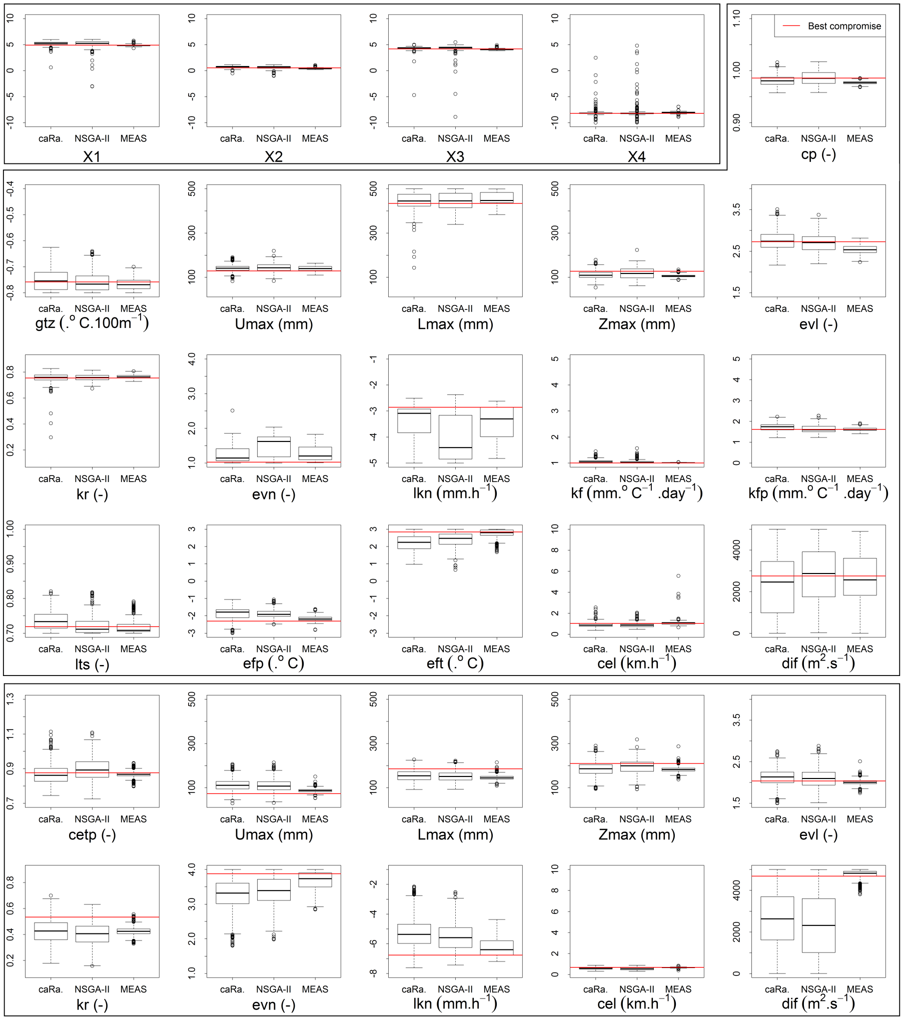


Figure 10. Calibrated parameters distribution for the sets on the Pareto front (y limits are the calibration bounds) with caRamel, NSGA-II, and MEAS for the three hydrological case studies: Blue River (first bloc of four parameters), Durance river (second bloc of 16 parameters), and Tarn River (third bloc of 10 parameters). Parameter values from the "best-compromise set" are displayed in red.

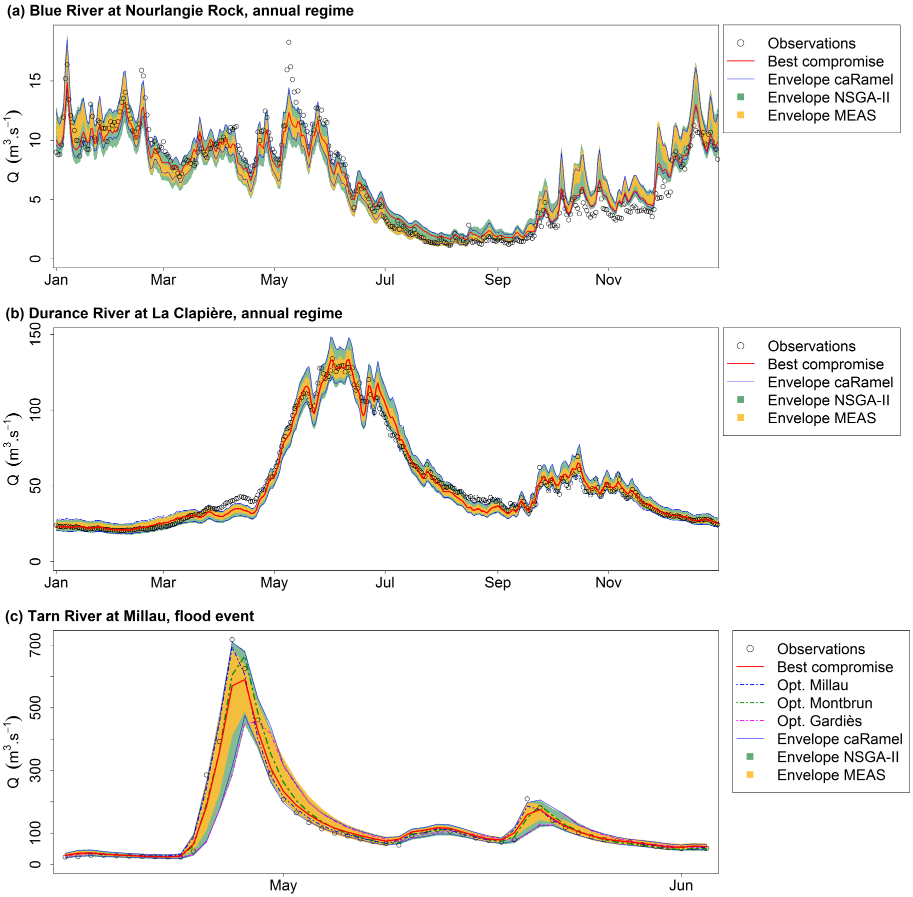


Figure 11. Observed and simulated stream flow for the 3 case studies. "Observations": observed streamflow, "Best compromise": best-compromise simulated streamflow, "Envelope": simulated streamflow envelope using all parameter sets on the Pareto front (over 40 optimizations) with caRamel, NSGA-II, and MEAS. (a) Daily runoff regime of Blue River at Nourlangie Rock (1990–1999); (b) Daily runoff regime of Durance at La Clapière (1991-2000); (c) Flood event of Tarn River at Millau (14/04/1993 - 03/06/1993)

6 Conclusions

CaRamel is an optimization algorithm for multi-objective calibration, its result is a family of parameter sets that are Pareto-optimal with regard to the different objectives. The algorithm is a hybrid of the MEAS algorithm (Efstratiadis and Koutsoyiannis, 2005) by using the directional search method based on the simplexes of the objective space and the ϵ-NGSA-II algorithm with archiving management of the parameter vectors classified by ϵ-dominance (Reed and Devireddy, 2004).The combination of stochastic and gradient-like parameter generation rules helps convergence of optimization while preserving the diversity of the population in both objective and parameter spaces. Four examples of case studies of increasing complexity have been used to compare caRamel with NSGA-II and MEAS. Results are quite similar between optimizers and show that optimization converges more quickly with caRamel.

Multi-objective optimization may require thousands of evaluations, which can be a limitation for the calibration of time consuming models. To cope with this issue, parallel computation is implemented in the R package `caRamel`. Moreover, as convergence can be sensitive to the randomly chosen initial population, it is recommended to run several optimizations to ensure reproducibility. Finally, the sensitivity to caRamel internal parameters has not been presented in this manuscript. Recommendation based on benchmark studies is to give the same weight to each of the five generation rules. In the default version of the package, each rule generates five parameter sets.

A better consideration of equality or inequality constraints, such as relationship between two parameters, could be an improvement. Another perspective would be the ability of caRamel to deal with discrete parameters.

Code and data availability. The data analysis was performed with the open-source environment R (https://www.r-project.org). The algorithm is provided as an R package `caRamel`, available from GitHub at https://github.com/fzao/caRamel, or from CRAN: https://cran.r-project.org/package=caRamel. The case study of Blue River at Nourlangie Rocks has been run by using `airGR` package for the GR4J hydrological model and for the data set, available at https://cran.r-project.org/package=airGR.

Appendix A: The `caRamel` R package

The `caRamel` package has been designed as an optimization tool for any environmental model, provided that it is possible to evaluate the objective functions in R. The main function, caRamel, is called with this syntax: caRamel(nobj, nvar, minmax, bounds, func, popsize, archsize, maxrun, prec). Arguments are detailed in Table 2. The main argument of caRamel is the objective function that has to be defined by the user. This enables flexibility as the user gives all the necessary information: the number and the definition of all the objectives, the minimization or maximization goal for each objective function, the number of parameters to calibrate and their bounds, and other numerical parameters such as the maximum number of simulations allowed. Additional optional arguments give the following possibilities:

- Creation of blocks/subsets of parameters that should be jointly recombined (for example parameters of a same module);

- Choice of parallel or sequential computation;
- Continuation of optimization starting from an existing population;
- Saving of the population after each generation or only the final one;
- Indicating the number of parameters sets generated by generation.

As a result, the function returns a list of six elements:

- success: a logical, "TRUE" if the optimization process ran with no errors,
- parameters: matrix of parameter sets from the Pareto front (dimension [number of sets in the front, number of calibrated parameters]),
- objectives: matrix of associated objective values (dimension [number of sets in the front, number of objectives]),
- save_crit: matrix that describes the evolution of the optimization process: for each generation, the first column is the number of model evaluations, and the following ones are the optimum of each objective taken separately (dimension [number of generations, (number of objectives +1)]),
- total_pop: total population (dimension [number of parameters sets, (number of calibrated parameters + number of objectives)]).
- gpp: the calling period for the third generation rule (independent sampling with a priori parameters variance). It is computed by the algorithm if the user does not fix it.

The R package contains an R vignette that gives as examples benchmark functions with 2 objectives and 1 or 3 parameters Schaffer (Schaffer, 1984) or Kursawe (Kursawe, 1991).

Appendix B: Example of R script for Kursawe test function optimization

```
# Kursawe function definition
kursawe <- function(i) {
Obj1 <- -10 * exp(-0.2 * sqrt(x[i,1] ˆ 2 + x[i,2] ˆ 2)) - 10 * exp(-0.2 * sqrt(x[i,2] ˆ 2 + x[i,3] ˆ 2))
Obj2 <- abs(x[i,1]) ˆ 0.8 + 5 * sin(x[i,1] ˆ 3) + abs(x[i,2]) ˆ 0.8 + 5 * sin(x[i,2] ˆ 3) + abs(x[i,3]) ˆ 0.8 + 5 * sin(x[i,3] ˆ 3)
return(c(Obj1, Obj2))
}
# Parameters definition and caRamel run
nobj <- 2 ; nvar <- 3 ; bounds <- matrix( c(rep(-5, nvar),rep(5, nvar)), ncol = 2 ) # range [-5, 5]
results <- caRamel (nobj = nobj , nvar = nvar , minmax = c(FALSE, FALSE) , bounds = bounds, func = kursawe, popsize =
100 , archsize = 100, maxrun = 5000, prec = rep(1.e-3,nobj) )
```

Appendix C: Example of R script for GR4J optimization

```
library(airGR)
library(caRamel)
# loading catchment data #
data(L0123001)
# preparation of the InputsModel object
InputsModel «- CreateInputsModel(FUN_MOD = RunModel_GR4J, DatesR = BasinObs$DatesR, Precip = BasinObs$P, PotEvap = BasinObs$E)
# run period selection
Ind_Run <- seq(which(format(BasinObs$DatesR, format = "%Y-%m-%d")=="1990-01-01"),
which(format(BasinObs$DatesR, format = "%Y-%m-%d")=="1999-12-31"))
# preparation of the RunOptions object
RunOptions «- CreateRunOptions(FUN_MOD = RunModel_GR4J,InputsModel = InputsModel, IndPeriod_Run = Ind_Run)
# Observation object
Obs «- BasinObs$Qmm[Ind_Run]

# Definition of functions for the optimizer #
# Function for model evaluation #
EvalGR <- function(i)
# Transformation of the parameter set to real space
RawParamOptim <- airGR::TransfoParam_GR4J(ParamIn = x[i,],Direction = "TR")
# Simulation given a parameter set
OutputsModel <- airGR::RunModel_GR4J(InputsModel = InputsModel,RunOptions = RunOptions,Param = RawParamOptim)
# Evaluation of the 3 components of KGE
Sim <- OutputsModel$Qsim
ix <- is.na(Obs + Sim)
B <- sum(Sim[!ix])/sum(Obs[!ix])
alpha <- sd(Sim[!ix],na.rm = TRUE)/sd(Obs[!ix],na.rm = TRUE)
rho <- cor(Obs[!ix],Sim[!ix])
KGE_3 <- c(rho , alpha, B)
return(1-sqrt((1-KGE_3)2̂))
```

```
# Function for cluster initialization
InitGR <- function(cl,numcores)
parLapply( cl, 1:numcores, function(xx)require('airGR'))
clusterExport(cl=cl, varlist=c("InputsModel", "RunOptions", "Obs"))



# Optimization #
# definition of the bounds of parameters (between -9.99 and 9.99)
nobj <- 3
bounds <- matrix(c(rep(-9.99, 4),rep(9.99, 4)), ncol = 2)
# Run
results <- caRamel(nobj = nobj, nvar = 4, minmax = rep(TRUE,nobj), bounds = bounds, func = EvalGR, funcinit = InitGR,
popsize = 100, archsize = 100, maxrun = 5000, objnames = c("KGE_r","KGE_a","KGE_b"), prec = rep(1.e-4,nobj))
```

Author contributions. NLM developed the algorithm in the Scilab platform. FH, FZ and CM adapted the algorithm as R package and performed various tests cases. CM prepared the manuscript with contributions from all co-authors.

Competing interests. The authors declare that they have no conflict of interest.

Acknowledgements. The authors want to thank the editor and the reviewers of this paper for their constructive suggestions. Special thanks to Andreas Efstratiadis who gave us MEAS source code and to Guillaume Thirel for a first script with `airGR`.

References

Baluja, S., and Caruana, R.: Removing the genetics from the standard genetic algorithm. In Machine Learning Proceedings 1995 (pp. 38-46). Morgan Kaufmann, 1995.

Coron, L., Thirel, G., Delaigue, O., Perrin, C. and Andréassian, V. The Suite of Lumped GR Hydrological Models in an R package. Environmental Modelling and Software, 94, 166-171. DOI: 10.1016/j.envsoft.2017.05.002, 2017.

Coron, L., Delaigue, O., Thirel, G., Perrin, C. and Michel, C. airGR: Suite of GR Hydrological Models for Precipitation-Runoff Modelling. R package version 1.3.2.23. https://CRAN.R-project.org/package=airGR, 2019.

Deb, K., Pratap, A., Agarwal, S. Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput., 6(2), 182-197, 2002.

Efstratiadis, A. and Koutsoyiannis, D.: The multi-objective evolutionary annealing-simplex method and its application in calibration hydrological models, in EGU General Assembly 2005, Geophysical Research Abstracts, Vol. 7, Vienna, 04593, European Geophysical Union. doi:10.13140/RG.2.2.32963.81446, 2005.

Efstratiadis, A., and Koutsoyiannis, D.: Fitting hydrological models on multiple responses using the multiobjective evolutionary annealing simplex approach. In: Practical hydroinformatics: Computational intelligence and technological developments in water applications, edited by R.J. Abrahart, L. M. See, and D. P. Solomatine, 259-273, doi:10.1007/978-3-540-79881-1_19, Springer, 2008.

Efstratiadis, A., and Koutsoyiannis, D.: One decade of multiobjective calibration approaches in hydrological modelling: a review. Hydrological Sciences Journal, 55(1), 58-78, doi:10.1080/02626660903526292, 2010.

Ercan M. B., and Goodall, J. L.: Design and implementation of a general software library for using NSGA-II with SWAT for multi-objective model calibration. Environmental Modelling & Software 84, 112-120. http://dx.doi.org/10.1016/j.envsoft.2016.06.017, 2016.

Fisher, R.A.: On the mathematical foundations of theoretical statistics, Philosophical Transactions of the Royal Society A. 222: 309–368. doi:10.1098/rsta.1922.0009, 1922.

Garavaglia, F., Le Lay, M., Gottardi, F., Garçon, R., Gailhard, J., Paquet, E., and Mathevet, T.: Impact of model structure on flow simulation and hydrological realism: from a lumped to a semi-distributed approach, Hydrol. Earth Syst. Sci., 21, 3937-3952, https://doi.org/10.5194/hess-21-3937-2017, 2017

Gong, W., Duan, Q., Li, J., Wang, C., Di, Z., Dai, Y., Ye, A., and Miao, C.: Multi-objective parameter optimization of common land model using adaptive surrogate modeling, Hydrol. Earth Syst. Sci., 19, 2409-2425, https://doi.org/10.5194/hess-19-2409-2015, 2015.

Gupta, H. V., Kling, H., Yilmaz, K., and Martinez, G. F.: Decomposition of the Mean Squared Error & NSE Performance Criteria: Implications for Improving Hydrological Modelling, Journal of Hydrology 377 80-91. doi:10.1016/j.jhydrol.2009.08.003, 2009.

Her, Y., and Seong, C.: Responses of hydrological model equifinality, uncertainty, and performance to multi-objective parameter calibration, Journal of Hydroinformatics 20 (4): 864–885, https://doi.org/10.2166/hydro.2018.108, 2018.

Jiang, S., Ong, Y. S., Zhang, J., and Feng, L.: Consistencies and Contradictions of Performance Metrics in Multiobjective Optimization, IEEE Transactions on Cybernetics, 2014.

Kursawe, F.: A variant of evolution strategies for vector optimization, in PPSN I, Vol 496 Lect Notes in Comput Sc. Springer-Verlag, pp. 193–197, 1991.

Le Moine, N.: Description d'un algorithme génétique multi-objectif pour la calibration d'un modèle pluie-débit (in French). Post-Doctoral Status Rep. 2, UPMC/EDF, 13 pp., https://www.metis.upmc.fr/~lemoine/docs/CaRaMEL.pdf, 2009.

Le Moine, N., Hendrickx, F., Gailhard, J., Garçon, R., and Gottardi, F.: Hydrologically Aided Interpolation of Daily Precipitation and Temperature Fields in a Mesoscale Alpine Catchment. Journal of Hydrometeorology, 16 (6), 2595–2618, doi: 10.1175/JHM-D-14-0162.1, 2015.

Madsen, H.: Parameter estimation in distributed hydrological catchment modelling using automatic calibration with multiple objectives, Advances in Water Resources, Volume 26, Issue 2, p 205-216, ISSN 0309-1708, https://doi.org/10.1016/S0309-1708(02)00092-1, 2003.

Magand, C., Ducharne, A., Le Moine, N., and Gascoin, S.: Introducing hysteresis in snow depletion curves to improve the water budget of a land surface model in an Alpine catchment. J. Hydrometeor., 15, 631–649, doi:10.1175/JHM-D-13-091.1, 2014.

Mersmann O., H. Trautmann, D. Steuer, B. Bischl, K. Deb. mco: Multiple Criteria Optimization Algorithms and Related Functions, website: https://CRAN.R-project.org/package=mco, 2014.

Monteil, C., Hendrickx, F., Samie, R. and Sauquet, E.: Modeling a complex system of multipurpose reservoirs under prospective scenarios (hydrology, water uses, water management): the case of the Durance River basin (South Eastern France, 12 800 km2). Geophysical Research Abstracts Vol. 17, EGU2015-4121-1, 2015 EGU General Assembly, https://www.researchgate.net/publication/323399497_Modeling_a_complex_system_of_multipurpose_reservoirs_under_prospective_scenarios_hydrology_water_uses_water_management_the_case_of_the_Durance_River_basin_South_Eastern_France_12_800_km2, 2015.

Nash,J. E., and Sutcliffe, J. V.: River flow forecasting through conceptual models part I — A discussion of principles, Journal of Hydrology 10, 282-290, https://doi.org/10.1016/0022-1694(70)90255-6, 1970.

Newland, C. P., Maier, H. R., Zecchin, A. C., Newman, J. P., van Delden, H.: Multi-objective optimisation framework for calibration of Cellular Automata land-use models. Environmental Modelling & Software 100, 175-200. https://doi.org/10.1016/j.envsoft.2017.11.012, 2018.

Oraei Zare, S., Saghafian, B., and Shamsai, A.: Multi-objective optimization for combined quality–quantity urban runoff control, Hydrol. Earth Syst. Sci., 16, 4531-4542, https://doi.org/10.5194/hess-16-4531-2012, 2012.

Perrin, C., Michel, C., Andréassian, V.: Improvement of a parsimonious model for streamflow simulation. J. Hydrol. 279, 275e289. http://dx.doi.org/10.1016/S0022-1694(03)00225-7, 2003.

Reed, P. and Devireddy, D.: Groundwater monitoring design: a case study combining epsilon-dominance archiving and automatic parametrization for the NSGA-II, in Coello-Coello C, editor. Applications of multi-objective evolutionary algorithms, Advances in natural computation series, vol. 1, pp. 79-100, World Scientific, New York. doi:10.1142/9789812567796_0004, 2004.

Riquelme, N., Von Lucken, C., and Baran, B.: Performance metrics in multi-objective optimization, XLI Latin American Computing Conference (CLEI), https://doi.org/10.1109/CLEI.2015.7360024, 2015.

Rothfuss, Y., Braud, I., Le Moine, N., Biron, P., Durand, J.-L., Vauclin, M., and Bariac, T.: Factors controlling the isotopic partitioning between soil evaporation and plant transpiration: Assessment using a multi-objective calibration of SiSPAT-Isotope under controlled conditions. Journal of Hydrology, 442–443, 75–88, doi:10.1016/j.jhydrol.2012.03.041, 2012.

Rouhier, L., Le Lay, M., Garavaglia, F., Le Moine, N., Hendrickx, F., Monteil, C., and Ribstein, P.: Impact of mesoscale spatial variability of climatic inputs and parameters on the hydrological response. Journal of Hydrology 553, 13-25. http://dx.doi.org/10.1016/j.jhydrol.2017.07.037, 2017.

Schaffer, J. D.: Some experiments in machine learning using vector evaluated genetic algorithms (artificial intelligence, optimization, adaptation, pattern recognition), PhD, Vanderbilt University, 1984.

Slater, L. J., Thirel, G., Harrigan, S., Delaigue, O., Hurley, A., Khouakhi, A., Prosdocimi, I., Vitolo, C., and Smith, K.: Using R in hydrology: a review of recent developments and future directions, Hydrol. Earth Syst. Sci., 23, 2939-2963, https://doi.org/10.5194/hess-23-2939-2019, 2019.

Smith, K. A., Barker, L. J., Tanguy, M., Parry, S., Harrigan, S., Legg, T. P., Prudhomme, C., and Hannaford, J.: A Multi-Objective Ensemble Approach to Hydrological Modelling in the UK: An Application to Historic Drought Reconstruction, Hydrol. Earth Syst. Sci., 23, 3247–3268, https://doi.org/10.5194/hess-23-3247-2019, 2019.

Sorooshian, S., and Dracup, J. A.: Stochastic parameter estimation procedures for conceptual rainfall-runoff models: Correlated and heteroscedastic error case. Water Resources Research, 16(2), 430-442, doi:10.1029/WR016i002p00430, 1980.

Van Veldhuizen, D. A.: Multiobjective Evolutionary Algorithms: Classifications, Analyses, and New Innovations, Faculty of the Graduate School of Engineering of the Air Force Institute of Technology, Air University, Dissertation AFIT/DS/ENG/99-01, 1999.

Zitzler, E. and Thiele, L.: Multiobjective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach, IEEE Transactions on Evolutionary Computation, vol. 3, no. 4, p.p. 257-271, 1999.

Zitzler, E., Deb, K., and Thiele, L.: Comparison of Multiobjective Evolutionary Algorithms: Empirical Results, Evolutionary Computation, vol. 8, no. 2, p.p. 173-195, 2000.

Zhou, A., Jin, Y., Zhang, Q., Sendhoff, B., and Tsang, E.: Combining model-based and genetics-based offspring generation for multi-objective optimization using a convergence criterion. In Proc. IEEE Cong. Evol. Comput., pages 892–899, 2006.

Table 1. Parameters to calibrate for MORDOR-TS and bounds of variation

Parameter	**Units**	**Prior range**	**Description**
cetp	–	[0.7, 1.3]	PET multiplicative correction factor
cp	–	[0.9, 1.1]	Precipitation multiplicative correction factor
gtz	$°\mathrm{C}.100\mathrm{m}^{-1}$	[-0.8, -0.4]	Air temperature gradient
umax	mm	[30, 500]	Maximum capacity of the root zone
lmax	mm	[30, 500]	Maximum capacity of the hillslope zone
zmax	mm	[30, 500]	Maximum capacity of the capillarity storage
evl	–	[1.5, 4]	Outflow exponent of storage L
kr	–	[0.1, 0.9]	Runoff coefficient
evn	–	[1, 4]	Outflow exponent of storage N
lkn	$\mathrm{mm.h}^{-1}$	[-8, -1]	Outflow coefficient of storage N
kf	$\mathrm{mm}.°\mathrm{C}^{-1}.\mathrm{day}^{-1}$	[1, 5]	Constant part of melting coefficient
kfp	$\mathrm{mm}.°\mathrm{C}^{-1}.\mathrm{day}^{-1}$	[0, 5]	Variable part of melting coefficient
lts	–	[0.7, 1]	Smoothing parameter of snow pack temperature
eft	$°\mathrm{C}$	[-3, 3]	Additive correction of melting temperature
efp	$°\mathrm{C}$	[-3, 3]	Additive correction of rain/snow partition temperature
cel	$\mathrm{km.h}^{-1}$	[0.1, 10]	Wave celerity
dif	$\mathrm{m}^2.\mathrm{s}^{-1}$	[10, 5000]	Wave diffusion

Table 2. Arguments of the caRamel() function. Optional arguments are printed in grey.

Name	Type	Description
nobj	integer, length = 1	number of objectives to optimize (at least 2)
nvar	integer, length = 1	number of parameters to calibrate
minmax	logical, length = nobj	vector of logicals that indicates for each objective whether it should be maximized (TRUE) or minimized (FALSE)
bounds	matrix, nrow = nvar, ncol = 2	lower and upper bounds for the variables
func	character, length = 1	the function to optimize (defined by the user), with VecObj = func(i) where i is the tested set index in the population matrix (x), and VecObj is the vector of objectives for this set.
popsize	integer, length = 1	population size for the genetic algorithm
archsize	integer, length = 1	size of the Pareto front
maxrun	integer, length = 1	maximum number of model runs
prec	double, length = nobj	desired precision for the objectives (used for downsizing population)
repart_gene	integer, length = 4	number of new parameter sets for each rule and per generation
gpp	integer, length = 1	calling period for the rule (3)
blocks	list of vector integer	functional groups for parameters
pop	matrix, nrow = nset, ncol = nvar or nvar+nobj	initial population (used to restart an optimization)
objnames	character, length = nobj	names of the objectives
listsave	list of character	names of the listing files (NULL by default: no output)
write_gen	integer, length = 1	if = 1, save files ’pmt’ and ’obj’ at each generation (= 0 by default)
carallel	logical, length = 1	run parallel computations (TRUE by default)
numcores	integer, length = 1	number of cores for the parallel computations (all cores by default)
funcinit	character, length = 1	the function (defined by the user) applied on each node of cluster for initialization when parallel computation (for example load of packages or copy of data). Arguments must be cl and numcores.
graph	logical, length = 1	plot graphical output at each generation (TRUE by default)