Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 December 2019 doi:10.20944/preprints201912.0249.v1

A practical guide for conducting calibration and decision-making optimisation with complex ecological models.

Stéphanie Mahévas[1], Victor Picheny[2], Patrick Lambert[3], Nicolas Dumoulin[4], Lauriane Rouan[5], Jean-Christophe Soulié[6], Dimo Brockhoff[7], Sigrid Lehuta[1], Rodolphe Le Riche[8], Robert Faivre[2], Hilaire Drouineau[3]

1 smahevas@ifremer.fr : Ifremer, EMH, Nantes, France.

2 MIAT, Université Fédérale de Toulouse, Inra, Castanet-Tolosan, France.

3 Irstea UR EABX, Bordeaux, France.

4 Irstea UR LISC, Clermont-Ferrand, France.

5 CIRAD, UMR AGAP, F-34398 Montpellier, France. AGAP, Univ Montpellier, CIRAD, INRA, Montpellier SupAgro, Montpellier, France.

6 CIRAD, RU Recycling & Risk, F-97743 Saint-Denis, Réunion, France. Recycling & Risk, University of Montpellier, CIRAD, Montpellier, France

7 Inria, Randomized Optimization team, Saclay, France.

8 CNRS LIMOS at Ecole des Mines de St-Etienne, France.

Abstract

Calibrating ecological models or making decisions with them is an optimisation problem with challenging methodological issues. Depending on the optimisation formulation, there may be a large variety of optimisation configurations (e.g. multiple objectives, constraints, stochastic criteria) and finding a single acceptable solution may be difficult. The challenges are exacerbated by the high computational cost and the non linear or elusive mathematical

© 2019 by the author(s). Distributed under a Creative Commons CC BY license.

properties that increased with the complexity of numerical models. From the feedbacks of practitioners, the need for a guideline for conducting optimisation of complex models has emerged. In this context, we propose a practical guide for the complex model optimisation process, covering both calibration and decision-making. The guide sets out the workflow with recommendations for each step based on existing tools and methods usually scattered throughout the literature. This guide is accompanied with an ODDO template (Overview, Design, Details of Optimisation) to standardise the published description of model-based optimisation and suggests research directions.

Keywords : Optimisation, complex models, calibration, identification, parameter estimation, decision-making, pre-processing, post-processing, tracability.

1 Introduction

Scientists are facing the challenge of modelling the complexity inherent in biological, ecological, social and socio-ecological systems to study real systems, anticipate changes and assist decision-making processes ([36] and see Annex 1 for illustrations). The continuous increase in computer power associated with easier access to simpler computer programming and increasing data availability, have favored the development of increasingly complex models. Complex models provide a more realistic description of the real system and allow to address a wider range of questions than simple models [24]. However, outcomes of complex models are generally thought to be more sensitive, less robust and less generic than those of simple models [75,4]. Moreover, simple models are easier to implement, analyze, check and communicate [22]. All these statements may outweigh the benefits provided by complex models and weaken the trust in the reliability of their outputs. It is one thing to run a model, it is quite another to grant confidence to the model outputs [48].

In this context, calibration methods are useful tools to strengthen trust in complex models [7,72]. Model calibration consists in fitting a model to observed data or established patterns by tuning the values of some model parameters, i.e. seeking sets of model input parameters that ensure that the model correctly reproduces observed or known functioning of the real system. In practice, the calibration is generally carried out by seeking the set of model parameters that minimises one (or several) objective function(s) and, as such, is based on numerical optimisation (e.g. Annex 1). Objective functions aim at quantifying the discrepancy between simulated and observed data/patterns and consequently, quantify the model fidelity to the system of interest [35,115,81,72]. Alternatively, numerical optimisation is also widely used to provide recommendations with complex models for environmental decision-making and management. Here, unknown parameters relate to alternative decision and objective functions quantify the outcome of the decision. For example “which policy can improve operational processes and resolve conflicts between stakeholders?” is a common question that models can help to answer optimising a modelled decision variable [59,105]. Decision-making using models involves searching for an optimal configuration of decision variables (a subset of the input variables with which the system can be controlled/steered) to meet given management objectives [87].

Solving an optimisation problem is often difficult, especially in the context of complex models. The first reason relates to the complexity of models by itself. For instance, numerical optimisation generally requires to run a large amount of simulations, which is not always possible because complex models often suffer from computationally intensive evaluations [119]. Moreover, the objective functions of complex models very often display specific features (e.g. among others discontinuities, non linearities, high dimension, stochasticity, multimodality) that raise specific methodological challenges [98,11]. The second reason is that a large diversity of skills is required to overcome these issues of complex optimisation, and few people master them all: thematic skills related to the model (formulating the problem and testing the solution), mathematics (formulating the objective function and creating the optimisation algorithm) and computer science (coding the algorithm and the model). In most cases, only one or two of these skills are the core business of the person conducting the optimisation. Modellers develop the models and best know their mathematical properties. Model users run the model, interpret the results and make decisions about the systems studied. Model users may have less knowledge of the models' properties (e.g., when the models are taken as black-boxes) and of optimisation methods. Numerical specialists develop the optimisation algorithms and know little about the model and its field of application. Statisticians can also provide important recommendations in the construction of the objective function and in the analysis of the results. Finally, programmers can be involved in the computer programming of the model or of the optimisation algorithm. Ideally, it would be beneficial if optimisation was undertaken in cooperation between the modeller, the model user, the numerical specialist, the programmer and the statistician. Yet there is undoubtedly too little contact between these groups of experts. This paper reflects a collective effort initiated at the MEXICO (Methods for numerical EXploratIon of COmplex models) Optimisation Workshop (MEOW [83], for more details see Annex 1) to enhance exchanges about complex ecological model optimisation among all types of experts involved in the topic. MEOW has revealed a general pragmatic but inhibiting strategy of model users. Most of them force themselves into twisting their optimisation issue to fit configurations they know and master rather than formulating the problem without a priori and selecting the most relevant (possibly new) approach.

Among other things, MEOW highlighted a need for a practical guide for ecological modellers and model users to achieve efficient and effective complex model optimisation (Annex 1). This

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 December 2019 doi:10.20944/preprints201912.0249.v1

paper aims at filling this gap by providing a guideline to complex model optimisation. The first challenge in the optimisation of complex models lies in the formulation of the problem. This step is always critical but is rarely described in the published work creating the illusion that optimisation can be reduced to the selection of the optimisation algorithm. Selecting an algorithm is the next arduous task in optimisation. As stated by the *No Free Lunch Theorem* [116], universally efficient solvers do not exist, so each problem requires an appropriate, case-specific choice of optimisation method [118,87]. This has lead to a vast literature with an abundant list of methods and a variety of implementations for each method as well as many variants or particular tunings (see e.g. [80,11]. However most papers are intended for numerical specialists. Finally, once the algorithm returns a solution, much work is still required to assess the quality of this solution and decide whether to stop the optimisation process and accept the solution found or to start the optimisation again, rectifying the process using the post-processing diagnostics [120,23]).

In this paper, we provide a practical guide dedicated to modellers and model users to help them to optimise complex systems models for both calibration and decision-making, in situation where the model is considered as a black-box. We do not claim to provide a new review of optimisation techniques and numerical algorithms but rather to describe and lead the user into a more comprehensive optimisation process. With the bias of a lack of knowledge of model properties, our objective is here to support them in conducting an informed optimisation approach through a well-defined formulation of the problem, a relevant selection of the algorithm and a rigorous analysis of the outcomes : in other words, performing a white-box optimisation process (also called « integrated approach » in [80]) while operating a black-box model. We describe the three main steps of an optimisation process (pre-processing in section 2, selecting the algorithm in section 3 and post-processing in section 4) and provide decision trees to help practitioners cutting their paths through what often looks like a jungle of questions and methods. Finally, in Section 5, we provide a practical template for summarizing and tracking an optimisation process called ODDO for Overview, Design, Details of Optimisation (by analogy with the ODD protocol of [38] for complex model description, recently updated [39].

2 Pre-processing

The main steps and decision rules (detailed below) of the optimisation pre-processing are described in Fig 1 and Fig 2. Pre-processing includes methods for defining, analysing and reshaping the problem formulation : data and parameters exploratory analysis, building an initial objective function, dimension reduction of S and reshaping the objective function. This step in the optimisation process may lead to a new formulation of the problem (see loops in Fig 2). The outcome of this analysis will facilitate the optimisation and help the model user selecting the appropriate algorithm (green boxes in Figs 1 and 2).

We can distinguish pre-processing methods that do not require any use of the model from those that require the model to be run to calculate the objective function. Contrary to post-processing (see Section 3.3), pre-processing does not require the use of any optimisation algorithm.

2.1 Problem formulation

The first step in any optimisation exercise is to achieve a formulation of the problem that reflects stakeholders purposes. This formulation is likely to be improved iteratively in course of the optimisation rounds (first results and post-processing, see following sections), but the primary aim is to state important aspects of the problem under consideration and come up with the key requirements for selecting an appropriate algorithm. We distinguish six items.

Item 1: Define the purpose of the optimisation. What is the purpose of the optimisation? Are we calibrating a model or optimising for decision-making? For model calibration, what are the features to reproduce? Are we more interested by the model outputs (for example, to enhance predictive ability of the model) or by the value of estimated parameters themselves (for example, to understand the functioning of the studied system)? These specifications will influence the building of the objective function and the agreement on the quality of solution delivered by the algorithm.

Item 2 (for calibration): List all the data available to build the most accurate and relevant objective function. Data need to be distinguished between observations from the "real world" or expert knowledge. This distinction is necessary to take appropriate precautions to formulate the objective function. Treatments could actually differ whether data are quantitative or qualitative and collected according to a scientific protocol or opportunistically. It is also recommended to divide the dataset into two sets (by analogy with a statistical approach of parameters estimation) : the first uses to perform the calibration, the second to validate the model.

Item 3: List all parameters that need to be optimised, including their bounds, and whether the values are discrete or continuous. For decision-making, the parameters are the decision variables. These parameters will be called *optimisation parameters*, denoted as *x*, and the set of possible values they can take on the *parameters space, S* (sometimes called search space in literature), so that $x \in S$. Most optimisation algorithms can deal with bounds on parameters, but only few of them can deal with discrete parameters, consequently, in many situations the user will have to find a workaround in the objective function.

Item 4: In addition to parameter bounds, set out constraints such as inequalities or equalities between parameters and between model outputs. The existence of additional constraints will guide the choice of an optimisation algorithm that can cope with such features.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 December 2019 doi:10.20944/preprints201912.0249.v1

Item 5: List the sources of uncertainty in data and processes. Process uncertainty generally corresponds to stochasticity in the model and raise specific problems for the construction of the objective function. Uncertainty in data used as observations of the real world, may lead to consider them differently in the building of the objective function (weights, ranges, trends or summary statistics instead of point values).

Item 6 : Formulate the objective function(s), f (or $(f_i)_i$), to be optimised, specifying their number. The values taken by f (or $(f_i)_i$), define what we call the objective space denoted as Ω (or $\otimes_i \Omega_i$). For calibration, the objective function quantifies the discrepancy between model outputs and observations with possible penalty terms accounting for constraints (e.g., regularization of the model outputs). For decision-making, it quantifies the effects of the decision variables on the output of interest.

2.2 Data and parameters exploratory analysis

For calibration, the model is fitted to one or several datasets. As in any modelling exercise, the first pre-processing step consists in an exploratory analysis of the available data. We will not detail this step since it is not specific to optimisation problems, however it is necessary to explore the observations for detecting common modelling pitfalls such as outliers, over-dispersion and correlations (e.g recipes in [121,122]). The user may want to get rid of these features to prevent the optimisation algorithm to be slow or to fail in reproducing them.

In the presence of discrete parameters in the objective function, it is necessary to distinguish between quantitative and qualitative (categorical) parameters. If the discrete parameter is quantitative (for example, an integer), two approaches can be conducted. Either the user treats the integer parameter as a real parameter (possibly after a log-transformation) and then shifts to the regular continuous optimisation configuration, or either he keeps the discrete values and shifts to a “mixed” optimisation configuration. For this latter approach, some optimisation algorithms have been specifically developed but are not the most prevalent and easily available (see section 3.2.3) . Although the former approach looks much simpler at first glance, there is no guarantee that the model converges to the optimum (see section 4.2). If the parameter is purely categorical (a factor with unordered levels), the problem is more complex and there is no general solution. Very often, the user will have to carry out the optimisation for each level of the categorical parameter (i.e. an enumeration) and compare the results [71]. This can become very time-consuming, especially if the model has many categorical parameters.

2.3 Building an initial objective function

This is probably the most critical point as, regardless of the techniques deployed to answer a question, the quality of the answer depends on the quality of the question. In calibration, the objective function quantifies the discrepancy between model outputs and observations from the system the model seeks to mimic. It can express the goal to reproduce observations by estimating key parameters of the system as a way to improve the understanding of the system. The objective function is therefore a measure of the distance between the observed data and the model outputs. When data are quantitative, the most common functions are the least-squares distance and more generally the likelihood function, both of which have useful and well-known statistical properties [74,47]. When the value of the parameters are one of the aim of the calibration exercise, or if uncertainty in model predictions are important (for example for risk analysis), these statistical properties can be valuable because they allow to estimate confidence intervals around parameter estimates or predictions. In decision tasks, an operational model is optimised to assess limit conditions (e.g., "what is the largest/smallest response of the system such that ..."), minimise a risk or evidence trade-offs. Note that methodologies have been proposed to assist model users in the construction of the objective functions, for example to choose the most relevant statistics for an ABC optimisation algorithm [8, 26, 85,125]. When data are qualitative (e.g. zones, behavior), tricks are needed to consider them using a quantitative index (e.g. area of the zone, classification of behaviors in numbered groups) or summarise the patterns of interest quantitatively (pattern-oriented approach, [35, 108]. Here important questions need to be raised and answered by stakeholders such as: "do we want the model to reproduce trends or absolute values?", "At which scale (spatial, temporal, hierarchical levels), does the model need to be the most accurate?".

As mentioned above (paragraph 2.1 Problem Formulation), listing the sources of uncertainty is important. When fitting a model by maximising a likelihood function, stochasticity in the model gives rise to particular calculation issues. For example, in a state-space model that incorporates both process and observation errors, likelihood calculations involve high-dimensional integrals which are time-consuming and require ad-hoc implementation strategies [21,64]. If computing the exact likelihood is too time-consuming, the user may instead try to calibrate the model on the expectancy of the likelihood, or on a specific quantile. When there is uncertainty is the data used as observations, the user may want to consider the model close to data when it is within the uncertainty range, regardless how close it is from the point value.

Two questions raised by calibration deserve special attention when building the initial objective function. First, providing a robust objective function that prevents from overfitting specific data (outliers, data with small predicted variance...) is critical in a calibration exercise. Several methods from the field of robust statistics are available to tackle this issue. For example, [28,29] propose the use of heavy-tailed distribution functions to define the likelihood function and to bound the variance to avoid overfitting. [88] recommend external estimation of sampling error in likelihood function. In addition, the preliminary data exploration (section 2.2) can help to detect and remove part of the outliers.

The second question arises when several datasets or observation scales are available for calibrating the model. The datasets are often merged into a single objective function. A standard approach consists in weighting the datasets in the objective function [30]. However, aggregation

Bmethods, weights and scales strongly influence the optimum parameter sets [109, 2007; Barbour et al., 2016; 19]. The weights can be either based on experts' knowledge [52], or on an iterative process that mix empirical information and feedback from model fitting [30]. Other statistically-based objective functions have been proposed [19,73], that cannot however remove the risk of overfitting some datasets while granting other datasets insignificant weights. Sensitivity to weights is especially important when the model is not able to fit the different datasets simultaneously, i.e. decreasing the discrepancy between the model and one dataset tends to increase the discrepancies with the other datasets. In this kind of situation, it is recommended to associate a different objective function to each dataset and move to a multi-criteria optimisation method using Pareto fronts that will explicit the trade-offs between the different objective functions. It is of particular interest in decision or design optimisation, where the formulation of the objective function can be highly delicate (see for instance Barbour et al (2016) for a review in river ecosystems management or [32]). However, with more than 4 criteria (see section 3.2.3 for more details), multi-criteria optimisation lose their incentive as the Pareto fronts contain a too large portion of the decision space and cannot be easily visualized. A possible workaround is to focus on the two or three most important criteria and to use constraints on the others, that is, to perform a multi-criteria optimisation that takes model user's preferences into account [18, 27,110].

2.4 Dimension reduction of *S*

Dimension reduction methods aim at reducing the number of parameters to be optimised and, consequently, the complexity of the optimisation problem. [105] proposed an interesting review of dimension reduction strategies which can be roughly divided into two families. The first family comprises methods for screening the parameters by quantifying the influence of the parameters on the model outputs and the objective function: parameters with a limited influence can be set to default values so that the optimisation focuses on the most important parameters.

The most important parameters can be identified for example by carrying a preliminary sensitivity analysis, which is a standard tool for exploring numerically complex models [25, 101,71]. The first step of sensitivity analysis consists in implementing a simulation design to set up simulations in a way that maximise the efficiency of the exploration of the parameter space. As an example, a Latin hypercube sampling (LHS) design [55] incorporates many suitable features of random and stratified sampling. As such, it allows an efficient variance decomposition of the objective function partitioned among the different parameters of the model [84,31]. Another advantage of this preliminary approach is that the results of the simulation design can also be used to inform on the shape of the objective function (section Reshaping the objective function) and on possible correlations among parameters.

The second family is made of methods for transforming a set of correlated parameters into a smaller set of uncorrelated parameters [70]. The identification of correlated parameters can be carried out using projection methods of model outputs (e.g. principal component analysis, principal component regression or partial least square regressions). It requires once again to run the model on a set of parameters values combinations (e.g. from an LHS design). The reduction of space can then be operated focusing on the group of correlated parameters (e.g. using the linear combination of correlated parameters) rather than on each each parameter independently.

2.5 Reshaping the objective function

Reshaping of the objective function refers to modifications of the initial objective function to achieve a more suitable one before running the optimisation algorithm. Typically, the objective function should as much as possible be isotropic with respect to parameters to facilitate the work of the optimisation algorithm [9]. Re-parametrisation is a very common technique leading to a new objective function in situations of anisotropy or when correlation between parameters is detected. Anisotropy can occur when parameter ranges are very different. Scaling techniques can address such problems and aim at redefining parameters so that their ranges are more similar [9]. [9] gave an overview of standard re-parametrisation methods that can be applied to simplify the model formulation, and a simple example of growth curve fitting can be found in [104] in a situation of strong correlation of parameters. Very simple modifications can very often much improve the situation: for example scaling datasets often reduce correlations among parameters and standardized weights among datasets. A logarithmic transformation of parameters can also be very useful if (positive) parameters have a very large upper bounds or to shift from a multiplicative function to an additive one.

Design of experiments can be used to explore the shape of the objective function. For example, computing an LHS design on *S* and simulating *f(x)* for each *x*-experiment of the design, the user can fit a metamodel (e.g. regression model like a generalised additive models GAM) with *f* as the dependent variable and parameters *x* as predictors. The plots of the *f* component smoothed on each parameter then provide valuable information on the shape of the objective function with respect to each parameter: it is possible to detect very flat zones where the optimisation algorithm can be trapped (in that case, bounds modifications can help to avoid these zones) or possible multimodality (in that case, it will be especially important in post-processing to check whether the optimisation algorithm converged to a local or a global optimum). Very often, it gives a rough idea of the location of the optimum and as such allows to provide more precise bounds to the optimisation algorithm. Moreover, the meta-model can be used to estimate correlation (concurvity for GAMs) among parameters: strong correlations among parameters will probably raise problems to the optimisation algorithm. It may either be due to insufficient data, but may also be intrinsically due to the mathematical structure of the model. In this case, the user may either re-parametrised these correlated parameters as indicated above.

3 Choosing the optimisation algorithm

Given that the numerical optimisation is at the core of the problem, it is not surprising that many different optimisation algorithms have been proposed in the literature. The abundance of algorithm types and implementations makes it difficult for the model user to choose an appropriate optimisation algorithm for a specific problem. To assist with this, numerical benchmarking is a crucial step to understand and to recommend algorithms in practice.

Here, we do not want to review all aspects of the wide research field of optimisation but instead give a short overview of the practically relevant class of *black-box* optimisation algorithms and give recommendations about which problem characteristics are important in order to make reasonable choices about the available algorithms. As a concrete example, a few recommendations for the most basic (and often observed) class of single-objective, unconstrained black-box optimisation problems in continuous domain are detailed as well as pointers to some efficient methods in more complex settings.

3.1 A brief review of black-box model optimisation algorithms

Black-box optimisation algorithms make only very few assumptions on the problem's underlying objective function and in particular do not assume that gradients are available. In general, black-box optimisation algorithms proceed by sampling the objective function sequentially, starting from an arbitrary point (or a set of points, $x \in S$) and explore the parameter space S to reach better values of the objective function $f(x)$. Difficulties in locating the best points typically arise if the search space is large, if the objective function(s) is(are) multimodal (with several local minima), discontinuous, if there are large plateaus, if the optimal region is narrow or curved and if the model underlying the optimisation criteria is numerically costly. Efficient optimisers that guarantee a global optimum for any problem do not exist [116] and many different algorithms have been proposed for a wide range of configurations (see e.g. two recent reviews of [98] and [1]).

There are several possible ways of classifying black-box model solvers. For example, [98] distinguish between direct and model-based approaches, between local and global searches and between stochastic and deterministic algorithms. We have chosen here to focus on the way the algorithms generate the information they need to guide the sequential optimisation of the black-box model. The reason is that in practice it seems less important whether for example an algorithm is stochastic or not. The main property we care about (besides an easy-to-use interface) is whether the algorithm is delivering good solutions in short time. In particular, we distinguish algorithms here only based on (i) how they propose new solutions at each of their iterations and (ii) how, internally, they approximate the considered objective function(s).

First, local searches aim at finding local optima and typically propose new solutions in a (small) part of the design space close to already sampled solutions, while global searches constantly compromise between exploring the whole design space (to avoid getting trapped around local optima) and local intensification (to locate the optimum more precisely). Local searches may converge to a local optimum that is determined by the initial values, while global search algorithms attempt to reduce this risk. Note that local search algorithms also have the capacity to perform a wide exploration, in particular in early iterations, when the "targeted" local region is still broad. Furthermore, local search algorithms can be made globally convergent by restarts [49,46,79]. This results in a smooth distribution of algorithms, from "more local" to "more global" (Fig 3).

Second, we distinguish between approaches by looking at whether or not an approximation to the objective function (or surrogate model, also known as metamodel) they use, since the usage of surrogate models is a basic tool for handling complex models. This results in dividing optimisation algorithms into four families (Fig 3) that will be detailed below: 1) “Local model” (with approximation of the objective function), 2) “Local sampling”, 3) “Global model” (with approximation of the objective function), 4) “Global sampling”. This classification still applies in the presence of constraints, stochasticity in the optimisation criteria, multiple objectives and parameters with discrete values.

Algorithms with Local Models : Algorithms with local models typically rely on information gathered around the current algorithm iteration and summed up in a locally valid approximation of the model. This information can only consist in so-called zeroth order information, i.e., (evaluated) objective and constraints values, and/or in additional information in terms of the first (“gradient”) and/or second-order (“Hessian”) partial derivatives of the criteria in case the variables are once/twice differentiable. Using these local surrogates of objective and constraint functions, the algorithms calculate steps to try to find a better solution. Newton algorithms, quasi-Newton algorithms such as the L-BFGS-B algorithm [76], and Trust-region approaches such as the derivative-free NEWUOA (Powell, 2006) belong to this category. Although not originally designed for it, those among these algorithms that require derivative information (mainly the quasi-Newton methods) can be applied to black-box model optimisation using numerical approximations of the derivatives (e.g. from finite differences) or automatic differentiation (though the objective has to be strictly differentiable with respect to the parameters in this case). Typically, local model optimisers are robust, that is always converge (in particular, trust-region methods have proven convergence towards a local optimum on well-behaved functions) and are able to tackle problems with numerous parameters. However, they only locate local optima unless coupled with multistart schemes, see for instance recommendations in [43]. They are usually suitable for restricted simulation budgets, but estimating the derivatives may become expensive as the number of parameters increases. Nevertheless, the derivative information (gradient, Hessian) can be useful for assessing the quality of the solution (see Section 3.3.3).

Locally Sampling Algorithms : The second family of algorithms relies on a local sampling scheme to determine which step should be taken from the current iterate. At each iteration one or several sets of new parameter values are chosen according to a sampling scheme that incorporates information about the location of the last or recent iterations (usually the best solution found so far or the mean of the current distribution) and dispersion around it. The scheme can be deterministic, based on a geometric approach (using a pattern search or simplex, for instance the Nelder-Mead algorithm [90], Mesh Adaptive Direct Search MADS [2], or random (using metaheuristics, for example evolutionary strategies, CMA-ES [42], Particle Swarm Optimisation, PSO [62], or Simulated Annealing, SA [111]). Such approaches can be applied to a wide range of functions as they do not assume that the optimisation criteria have regularity properties, and they can accommodate large numbers of parameters. But the stochastic members of this group (SA, PSO and CMA-ES) typically require large computational budgets to be competitive. Local sampling algorithms (often) have the ability to escape local optima, especially during the early iterations when the sampling scheme is broad, which has led some to consider them as global search algorithms, leading to confusion with global models and global sampling algorithms. However, in practice, they may converge to a local optimum in a single run. Again, restarting a local algorithm from a random initial point (or a set of random initial points) lead to mimic a global algorithm.

Algorithms with Global Models : The third family of optimisers relies on global models of the function. This group includes metamodel-based approaches (the so-called Bayesian optimisations see [10]; EGO of [58,12]) and Lipschitz optimisation (e.g., DIRECT, [57]). The overall idea is to build some knowledge of the form of the objective function throughout the search space (e.g. kriging), in order to control the algorithm trade-off between exploration (of unknown regions) and intensification (in promising ones). Such approaches typically seek the global optimum and asymptotically explore the entire parameter space. While the use of surrogate models allows optimisation with restricted simulation budgets, most algorithms are limited in terms of the number of parameters that they can handle.

Globally Sampling Algorithms : The fourth family of global sampling algorithms are similar in form to local sampling algorithms but differ in the sampling scheme with a trade-off between exploration and intensification and maintaining a global search through the iterations. Estimation of Distribution Algorithms (EDA, [69]), Spatial Branch and Bound (SBB, [51]) and Approximate Bayesian Computation (ABC, mostly for calibration problems, [15]) belong to this family. Such approaches may locate the global optimum but require very large computational budgets and few optimisation parameters.

3.2 Important problem properties and recommendations

3.2.1 Problem properties that play an important role

While the above history is almost generic to any black-box problems, concrete algorithms are still tailored toward certain types of black-box problems and it is crucial to first understand the problem properties.

The problem characteristics that predominantly influence the choice of the algorithm are the number of variables to be optimised and the available budget (in the number of function evaluations). In particular, some algorithms require large budgets to be able to be competitive, while others are unable to handle many variables. Other concrete problem properties that play an important role in the choice of an algorithm are the following. Does the problem contain a single or multiple objective function(s)? Do constraints need to be fulfilled? Is the search space continuous, discrete, or a combination of both? Is the problem noisy, that is, for example, is the same solution evaluated differently each time the objective function is called on it? Is the problem uni- or multi-modal, i.e. has a single global optimum or multiple (local) optima? Some of those questions might be only answered after a first optimisation run (see section 4), but the more details about the problem at hand are available, the better recommendations can be made about which algorithm to start from in practice. In general, it is a good idea to base any the decision for which algorithm to use on published experimental data with test functions rather than on guesses or random decisions. We therefore recommend to consider numerical experiments and data from sources like the Comparing Continuous Optimizers platform COCO ([44], http://coco.gforge.inria.fr/doku.php?id=algorithms-bbob-biobj) or the Black-Box Optimization Competition (BBComp, https://bbcomp.ini.rub.de/) showing proximities with user optimisation configuration.

3.2.2 Concrete Recommendations for Single-objective Unconstrained Black-box Problems in Continuous Domain

There are many benchmark studies and literature reviews that can help model users select algorithms but they generally focus on comparing the average performances of closely related algorithms over a large set of problems, which may not correspond to the specific needs associated to a given complex model. Here, we focus on the common class of single-objective unconstrained black-box problems in continuous domain for which the Black-Box Optimisation Benchmarking test suite implemented in the Comparing Continuous Optimizer Platform COCO [43,45] has become the state of the art and an excellent source of practically relevant algorithm recommendations.

Fig 4 is an attempt to summarize algorithm recommendations for this class of problems. It shows the maximum number of parameters an algorithm type can typically handle as well as the minimum number of model runs required to achieve the optimisation. The lower half of the graph contains what can be considered as unsolvable problems: when the limit on the number of simulations is too small for the number of parameters, most algorithms simply will not have

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 December 2019 doi:10.20944/preprints201912.0249.v1

sufficient "time" to propose a solution. The position of the algorithms in Fig 4 represents a typical use by numerical experts of the methods: if the number of parameters is low enough, it is worthwhile attempting global methods, while for large number of parameters local methods are more appropriate. Furthermore, algorithms that construct meta-models do not allow large numbers of simulations.

More concrete recommendations can be made from extensive benchmarking experiments such as the Black-box Optimisation Benchmarking test suite if additional problem properties are available. Digging into the available data from the COCO platform (http://coco.gforge.inria.fr/doku.php?id=algorithms-bbob), we can for example give the following recommendations.

Nelder-Mead works well in low dimension (2 or 3 variables). With very small budget (<50 times dimension function evaluations), use surrogate model-based algorithms such as SMAC-BBOB or other Bayesian optimisation algorithms. In larger dimension but with still small budget (<500 times dimension function evaluations), good algorithms are Powell's NEWUOA or SLSQP [65]. With larger dimension (5+ variables) and larger budget (>500 times dimension function evaluations), stochastic algorithms and in particular CMA-ES variants [41] perform best if no further information is available and the latter are even more preferable if the problem is non-separable and/or ill-conditioned. Portfolio algorithms, that run several algorithms in parallel, such as HCMA [77] might combine the best of all worlds if the available budget is large.

For a discussion about which algorithms perform well in the case of large-scale problems with hundred of variables, we refer to [112].

3.2.3 Cases with noise, multiple objectives, constraints and / or mixed variables

In general, one can find adaptations of most algorithms described above to specifically handle noise, multiple objectives, constraints and discrete variables. Note first that those characteristics may substantially increase the difficulty of the problem, hence the number of model runs needed to solve it using a given algorithm. However, this increase in the number of model evaluations is expected to be less than an order of magnitude, so Fig 4 may still apply to those cases. Although providing extensive recommendations for all the cases (and combinations of them) would be overly complex here, we point below situations for which the adaptation of some families of algorithms are more natural (or simply more successful). Note equally that algorithm recommendations shall be optimally made based on experimental data of numerical benchmarking experiments. Unfortunately, not much data is publicly available for all the considered cases as extensive research in these areas is still ongoing. Hence, one has to produce data for a specific problem of interest first or we have to fall back on our experiences with algorithms when data is not available.

In a multi-objective context, first benchmarking results are available, for example in the context of the COCO platform ([45], http://coco.gforge.inria.fr/doku.php?id=algorithms-bbob-biobj) or the Black-Box Optimization Competition (BBComp, https://bbcomp.ini.rub.de/) on which we build our recommendations here. Evolutionary strategies often seem the best alternative, provided that the budget is sufficient, since sampling sets of points at a time (instead of a single one) adapts well to the objective of uncovering a (subset of the) Pareto set. Standard choices may be the celebrated NSGA-II [17] for its simplicity or more recent alternatives, such as the SMS-EMOA [5] and its variants or the unbounded population MO-CMA-ES [66]. Recently, global model-based algorithms, such as SMS-EGO [94] have aroused substantial attention for their good performance with small budgets and also other deterministic global search methods such the DIviding RECtangles method [117] can be considered for small budgets. Again, portfolio algorithms such as HMO-CMA-ES [78] can be performant for both small and large budgets. Of note, most algorithms are designed to solve problems with two or three objectives, and perform poorly with more (the so-called “many-objective problems”, see [56]).

When constraints need to be satisfied, most of the already mentioned methods can accommodate them by simply adding a penalty term to the objective function or limiting the search space (in the case of box or linear constraints). General techniques to handle constraints with evolutionary methods can be found in [14]. Difficulty arises when constraints are complex to satisfy, either because of their number or because they are also outputs of the black-box, or if the feasible space (subset for which all the constraints are satisfied) is small. In that case, local model-based algorithms are good candidates as they more naturally account for constraints, for instance COBYLA, which relies on linear models [94], or SPG [6], that uses projected gradients. Note that problems become naturally the more complex the more constraints are added and it is thus advisable to remove or combine (some of) them if the first optimisation run is not satisfactory.

For noisy problems, generic recommendations are even more challenging as the type and strength of the noise can play a role. Learning form experimental data from the COCO platform,it turns out that, overall noise types and strengths, algorithms like MCS [53], SNOBFIT [54] and NEWUOA [95] are well suited for small budgets like in the noiseless case with the latter being particularly suited for moderate but not for severe noise. For large budgets of 1000 times problem dimension, stochastic methods, in particular CMA-ES variants are the best choice.

In many practical situations, in addition to the continuous variables we have been dealing with so far, there are discrete variables. The discrete variables can be ordinal, in which case there is a possible ordering of them, or categorical when there is no such ordering. The problem is said to have mixed variables when both discrete and continuous variables are present. Whenever discrete variables occur, the resulting optimisation problem belongs to the large domain of operations research where the most efficient algorithms are specialized for specific subclasses of problems. An important family of algorithms is that of Mixed Integer NonLinear Programming methods (MINLP,[123]) for which the discrete variables are ordinal and can be cast into integers. MINLP can be handled with ad hoc deterministic algorithms where the set of all possible combinations of discrete variables takes the form of a tree whose branches are restrictions on the possible values of the discrete variables (typically inequalities on them) that become singletons at the leaves. The leaves are made of continuous optimisation problems since all discrete variables have been chosen. MINLP algorithms use properties of the problem (convexity, possibility to "relax" integers to continuous numbers, …) to "cut" branches to avoid enumerating all discrete variables combinations. However, when the number of discrete variables combinations increases and when there is no simplifying mathematical property such as convexity or linearity, MINLP algorithms are expensive in terms of calls to the objective function. Therefore, in the most general problems with mixed variables, one typically relies on globally sampling heuristics, such as variants of the CMA-ES [44], SA and PSO methods cited above. An intensely studied example is that of evolutionary algorithms [60] where discrete and continuous variables are processed together through probabilistic moves

4 Post-processing

Post-processing aims at deciding if an acceptable solution has been found or if the optimisation process should be started again. The notion of acceptable is rather subjective and case/user-dependant, but we propose the three following steps to perform this post-processing stage: 1) checking the algorithm convergence, 2) assessing the quality of the solution, 3) given outcomes of 1) and 2) deciding if user stops the optimisation process and accepts the delivered solution or if user continues the process of optimisation returning to pre-processing stage with an alternative formulation, or to the selection of algorithm stage (selecting a new algorithm or settings new controlling parameter, sometimes called metaparameters). Post-processing involves operations performed on the outputs of the optimisation algorithm. We start the section detailing standard outputs of algorithms that are relevant for investigating the convergence and the quality of the optimum. As for the previous section, we provide general recommendations for the single-objective and unconstrained configuration and we dedicate a specific subsection to expand the discussion to multi-objective optimisation. As for pre-processing section, we summarised the decision rules and recommended actions in three figures, from Figs 5 to 7.

4.1 Outputs of algorithms for post-processing

Optimisation algorithms explore the parameters space to find an optimal value of the objective function. The most straightforward output is therefore the so-called trace of the algorithm that is made of the collection of visited points in the parameters search space S, (T_S) , and the corresponding objective function values in the objective space Ω, (T_Ω). Depending on the algorithm family (Section 3.2.1, Fig 3), several additional outputs can be post-processed. Algorithms with a “local model” usually provide, in the case of continuous parameters, an approximation of the derivatives and often the second-order derivatives (the Hessian matrix) of the optimisation criteria at the best point found. These are useful for assessing if the final solution is at least a local optimum (see 4.2), if there is an infinite number of solutions around it (a flat feasible valley characterised by null slope and directional curvature - see 4.3.2), and to build confidence intervals around the optimum. Sampling-based algorithms provide not only an estimated optimum but also a set of solutions around it which can serve to approximate the optimisation criteria locally. This set of good solutions may be transformed into a covariance matrix of the estimated parameters for later confidence measurements or, in the case of CMA-ES, the covariance matrix precedes the set of points. While local search algorithms provide accurate insight into the shape of the objective function around the region of convergence, global search algorithms give a more widespread but locally less precise information. The more exploratory the global optimisation algorithm is, the more the trace provides information for a global analysis of the objective function. As a general rule, we recommend to investigate the entire internal state variables of an algorithm and their evolution over time if this is accessible to the user. Internal state variables can take different forms: for example, the simplex of the Nelder Mead algorithm [90], the tuple of current search point, stepsize, evolution paths; covariance matrix in the CMA-ES [42]; or in the case of discrete optimisation the vector of sample probabilities in PBIL [3].

Most, if not all algorithms, have various stopping criteria and specify which criterion was met at the end of the iteration process (Fig 5). Optimisation algorithms generally have three kinds of stopping criteria: (1) some are based on simulation budget (time of computation, number of function evaluations, number of iterations....), (2) others are based on the exploration of the objective space (number of iterations without any progress, relative difference among the population of solutions…), (3) and a last type correspond to criteria on the exploration of the parameter space (derivatives, second-order derivatives, difference among solutions…). Checking which criterion was met will provide elements on the convergence of the algorithm (see section 4.2).

In the literature reviewing optimisation algorithms, algorithm behavior and algorithm performance are two widely used concepts to compare algorithms and understand why certain algorithms perform better than others [80]. Algorithm behavior aims at investigating the balance operated between between intensification (convergence to the nearest solution) and exploration (covering the parameter space to find a global solution) while algorithm performance focuses on how well an algorithm has performed on a particular problem and on comparing several algorithms. Several plots or synthetic quantitative metrics computed on the trace and on the distribution of solutions on the parameter space and the objective space have been proposed to perform diagnostics of behavior: distribution of solutions throughout the parameter space *S* like spread of solutions, predominant direction, frequency statistics of solutions, position of solutions on the boundary of feasible regions. On the other hand, effectiveness, efficiency, and for stochastic algorithm, reliability and robustness are the most widespread quantitative measures to assess the algorithm performance [80]. Both criteria allow to capture the important features of an algorithm's run-time search in order to gain insight into the actual influence of an algorithm's search methodology and its controlling parameters on its search behavior [3]. We explore how these metrics can be used in post-processing analysis.

4.2 Checking the algorithm convergence

Several factors can indicate the non-convergence of the algorithm and therefore that the derived solution is not exploitable yet. The most obvious is when the algorithm stops because the simulation budget was exhausted (criterion (1) in 4.1), i.e. generally before criteria on objective or parameters are reached. The two other configurations of algorithm stopping (criterion (2) on objective or criterion (3) on parameters) are obviously more suitable. However a good behavior of the algorithm in the parameter space can hide some instability in the objective space and the other way around. That it is why the user should check whether the criteria on the objective space are far from being respected or not if the algorithm stopped validating criteria on the parameter space, (and conversely checking criteria in *S* if it stops because of criteria in *Ω*). For example, if the objective function is not improving anymore across generations but derivatives of parameters remain high, this may indicate a strong correlation among parameters and the potential need for starting again the optimisation process by reducing the dimension of *S* (see section 3.4).

Beyond stopping criteria, the user should also analysed the traces of the algorithm and internal parameters of the algorithm. When converging, the objective function is expected to reach an asymptote (e.g. in the continuous variable case, a linear convergence, i.e. a constant slope on the convergence graph displaying log(f - $f_{\text{best-so-far}}$) vs. the number of function evaluations). For population-based metaheuristics, the same type of convergence should be observed on average. This exploration of the traces can be pragmatically performed using graphical exploratory plots of the objective function f against each parameter x and each couple of (x_j,x_k). This kind of analysis provides insights on how far the solution is from the probable optimum and so if it is necessary to extend the iteration process by modifying stopping criteria. For optimisers that provide an estimate of the Hessian matrix, the graphical exploration can also be combined with an analysis of the properties of this matrix that is expected to be positive definite (or at least positive semi-definite) at an optimum (local or global) [33].

What to do if the algorithm has not converged? Before increasing the number of iterations or relaxing some stopping criteria, explore the traces (T_S, T_Ω) over the last iterations can help to disentangle whether the non-convergence is due to an inappropriate formulation and an additional pre-processing step is required, or if it is due to a problem with the algorithm which requires new controlling parameters or even a change of algorithm. If the single-objective function is a combination of multiple components f_i (e.g. linear combination, see section 2.3), this graphical exploration can be expanded to each components f_i. Typical problems that can be detected by analysing those traces are described in a latter section (section 4.3.1) and potential solutions are proposed for each type of problem.

Some quantitative metrics in S and in Ω can be computed to synthesize and provide diagnostics of algorithm behavior. For instance, the dominant direction (the parameter leading to the highest gain in the objective function when changing its value) and the distribution of ending-traces solutions can be derived from the estimated slopes on each exploratory plot. If diagnostics metrics show that the algorithm is close from the convergence (region for x and decreasing f), then relaxing the stopping criteria and/or modifying some controlling parameters to increase the intensification could help and to speed up the convergence. Global model-based and global sampling-based approaches can often provide an estimate of the potential gain from additional iterations, for example the Expected Improvement of the EGO algorithm [57].

Though in practice, the use of optimisation algorithms is often off-the-shelf in a first step (without tuning algorithm parameters in the first place), in the post-processing step, a user might consider the (automated) tuning of the algorithm's internal parameters as an optimisation problem itself to finally obtain better results on the problem instance(s) of interest in a subsequent optimisation run [50]. On the other hand, if diagnostic metrics show oscillations between several regions of S across iterations, a (time-consuming) strategy would be to explore each one, one at a time, using a more local and efficient algorithm and select the best solutions among all. This might also indicate a problem of correlation of parameters or a parameter that have a very limited influence on the objective function that could necessitate to return to the pre-processing step to reduce the dimension of S or reshape the objective function (see section 2.4 and 2.3).

4.3 Quality of the solution

After checking that the algorithm has indeed converged to an optimum (exploring the last iterations of the algorithm), the user should assess the quality of this optimum (exploring the full behavior of the algorithm). Is this the global optimum or only a local optimum (Fig 6)? is this unique (Fig 7)? Is this realistic in the context of modelling or decision-making ?

4.3.1 Local versus global optimum

Checking that convergence is at a global optimum (i.e. we have found the best combination of parameters values of the model) remains a critical issue. Depending on the algorithm family, there are more or less chances to miss the problem optimal solution. Global search algorithms (global model and global sampling) are designed to provide a global optimum. However, even if they approach the overall optimum asymptotically, the associated cost may not be affordable. On the other hand, local search algorithms need fewer model evaluations but at the risk of only locating a local optimum. Checking the properties of the solution is therefore necessary whatever the selected optimisation algorithm (Fig 6).

The analysis of the whole traces (T_S, T_Ω) through graphical plots or synthetic quantitative measures as described in the section 4.2 are the most straightforward tools to assess the risk of convergence to a local optimum. For example, if the algorithm converges too fast and explores a very restricted region of the parameter space, this may indicate that the initial point of the algorithm is too close from an optimum and/or that the tuning of the algorithm is not appropriate (too small exploration and too strong intensification). In this case, it is worthwhile changing initial points (especially for local algorithms) and/or modify the settings of the controlling parameters of the optimisation algorithm to increase the exploration (see section 3.2)).

Another typical situation corresponds to cases in which the optimiser oscillates between different zones of the parameters space across generations. This generally suggests that the function is multi-modal. In such a situation, it is worthwhile carrying out the optimisation multiple times with different initial points to check that the optimisation algorithm consistently converges to the same solution. As proposed in the previous section, the user can also try to use local search with very limited exploration tuning and appropriate bounds to explore more intensively those differents zones of the parameter space. More generally, repeating local optimisations started from distant initial points is a widespread strategy for increasing the trust in the estimated optimum. One result of this exploration that is unfortunately seldomly assessed by model users is the distribution of the convergence points (e.g. [114]). The spread of this distribution (or its entropy) is an indication of the "risk" of missing the optimum.

The multi-dimensional nature of complex optimisation can impair the ability to analyse all the exploratory plots or synthetic measures of algorithm behavior. Approximating the objective function using a metamodel fitted on the whole traces (T_S, T_Ω) is an alternative way for addressing the issue of global versus local optima. We mentioned in the pre-processing step that a metamodel can be built after carrying out an exploratory numerical exploration of the objective function to derive information on its shape (e.g the presence of plateaus, multimodalities, local optima, convex and concave regions but also the presence of barriers). Such a metamodelling approach can also be applied after the optimisation using the traces provided by the algorithm. Moreover, comparing the metamodel based on the traces of the algorithm and the metamodel based on the preliminary exploration of the objective function can help to disentangle whether an optimisation issue originates directly from the characteristics of the functions or from an inappropriate exploration of the search space by the algorithm. However, to our knowledge, there are still few (or no) references on learning from metamodels fitted to optimisation traces.

4.3.2 Parameters Identifiability

A model is defined to be identifiable if only a single set of parameters values gives a particular model output [107]. Mathematically, an identifiable model is an injective map. Otherwise, there are redundant parameters. Problems of identifiability may arise from either an improper mathematical formulation of the model (intrinsic redundancy of parameters) or, for calibration, from insufficient data to discriminate between solutions (extrinsic redundancy of parameters) [34,123]. The identifiability issue arises for both calibration and decision-making. In decision-making, it implies that several decisions can lead to similar results. In calibration, it implies that it is difficult to decide between several sets of parameter values and may impair the prediction ability of the model. [34] give an overview of methods for detecting problems of identifiability in the context of mark-recapture models. Some of them, such as profile likelihood or formal derivatives are not easy to apply for black-box complex models. The former requires the objective function to be a likelihood or a pseudo-likelihood and the latter can be approximated with a metamodel of objective function. When using a model-based algorithm, the rank of the Hessian matrix gives the number of identifiable parameters [113]. For calibration, [61] and [103] proposed specific approaches for identifying intrinsic identifiability problems. They investigate estimability of model parameters increasing step by step model complexity and beginning with a reference model (simple and known to be identifiable). These methods are useful for disentangling identifiability issues arising from the data such as a lack of data or poor data quality, from those linked to the model and the formulation of the optimisation problem. With Bayesian algorithms, an identifiability problem can generally be detected by the presence of multiple optima in the posterior distributions, by a slow convergence, and by plotting correlation scatterplots between parameters. [124] recommended to explore the posterior distribution using tests based on cross-validation and pivot discrepancy measures. If the trace of the algorithm shows a long period of oscillations between solutions, this may show correlations between parameters that might be explicit in the optimisation problem formulation. In the case of “local model” algorithms applied to continuous parameters, the gradient vector and the eigen-analysis of the Hessian matrix tell whether the objective function is very sharp or flat around the optimum [33]. More generally as it also applies to non-continuous parameters, it is informative to carry out a numerical exploration such as a local sensitivity analysis around the solution to explore the local form of the objective function [63,86]: very flat directions often indicate correlations among two or more parameters.

Different solution are possible if some parameters are found to be redundant (red boxes in Fig 7). In some situation, for example if the model is calibrated for a purely prediction use and the values of parameters are of minor importance, the correlation of parameters is not necessarily problematic. Otherwise, it can be useful to detect whether the redundancy is extrinsic redundancy (lack of data) or intrinsic (formulation of the model). A convenient solution to disentangle the source of parameter redundancy is to carry out a simulation/estimation exercise (also called twin experiments) : the user simulates an artificial and very large dataset using the model using guessed value for parameters and then calibrate the model on this dataset [34]. In the absence of intrinsic redundancy, the optimisation algorithm should converge to the guessed parameter value. The problem is then due to the data and additional data are required. However, it should be noticed that this kind of exercise can be time-consuming for very complex models. In the presence of problematic intrinsic redundancy, additional pre-treatments such as a reduction of the dimension of a reshaping of the objective function are required. Finally, the redundancy also a result in itself that can be communicated to decision-makers with a take-home message like several solutions are possible regarding the selected criteria.

4.3.3 Residual analysis

In a calibration exercise, analysing the residuals (the difference between observed data and model prediction) is an essential task [13]. We will not detail this step since it is not specific to complex models [89]. However, it is important to check that the statistical assumptions used to sample the search space (often independent and identically distributed sample) and to approximate the objective function (often unbiased model outputs) are met (Fig 7). Identifying patterns in the residuals, such as a correlation between residuals and a covariate, can help to detect if some features of the model do not well-capture a functioning of the studied system (though this is often difficult where there is a large number of parameters). Residual correlations may also reveal a necessary change in the building of the objective function where residuals are re-weighted to be decorrelated.

If the objective function is based on several datasets, the residual analysis is also useful to check whether one (or several) datasets was not overfitted (consistently very small and unbiased residuals) with respect to the others (large and/or biased residuals). In this case, the weights of the objective function were probably not appropriate (see section objective function) and a sensitivity analysis with respect to the weights can help to find more suitable weights. If the number of datasets is limited, multi-criteria optimisation methods can be useful to visualize the trade-offs between the different datasets.

4.4 Multi-criteria optimisation

While single criterion optimisation has a single optimal solution (even if the model user might be interested in a set of good solutions, either to avoid local optima or to explore different solutions when the model is not identifiable), multi-objective optimisation generally admits a set of solutions. In this context, post-processing encompasses checking Pareto optimality (i.e. if none of the objective functions can be improved in value without degrading some of the other objective values) and screening solutions within the calculated trade-off set [16]. By post-processing the resulting solution set, looking at its members together in space of parameters and criteria, knowledge can be gained about the ill-posedness of the problem (many solutions almost equivalent with respect to certain objectives) and about the sensitivity of given objectives within the set of Pareto optimal points [23]. Analysing the solutions is relatively easy if there are only two or three objectives, as this can be done visually, for instance with the help of Empirical Attainment Functions [40], but becomes more difficult with more objectives and requires specific tools [56].

5 Synthesis

Fig 8 sums up the recommended workflow for conducting complex model optimisation. The process is based on the three prerequisites given in the box at the top of Fig 8: a model, a question to be addressed during the optimisation, and a set of observation data to be mimicked or a decision/design to be optimised. The progress of the optimisation is mainly sequential (represented by the large arrow in the background of Fig 8) with three steps, pre-processing, optimisation and post-processing. However most diagnoses performed at each step can lead to adaptation or a restart of the optimisation process (as highlighted by the numerous loops in Fig 8). The process starts with the formulation of the objective function and constraints. This preliminary step includes the definition of the parameters to be optimised and the exploration of the observation data. When there are too many parameters, a sensitivity analysis may be performed to reduce their number. Data should be explored before starting the optimisation procedure in order to assess which data are usable (detection of outliers, overdispersion problems, correlation between parameters). Devoting thoughts to the meaning and the mathematical properties of the optimisation criteria making the problem formulation, before implementing the algorithm, will often improve the overall process performance. This can be seen as an adjustment of the problem to make it more suitable to the available algorithms, the issues addressed and the data available. An optimisation algorithm is then chosen and then implemented. Post-processing generally focuses on the convergence criteria and on parameters identifiability to assess whether the optimisation results are satisfactory. For calibration, the relevance of parameters values (derived from the optimisation) may be analysed by simulation using a specific set of data that was not used for the calibration, or simply tested by comparison with theoretical knowledge in the domain. For decision-making, the solutions, or at least some of them, must make sense to the decision maker. If the results are unsatisfactory, the objective functions, constraints, and the optimisation algorithm (including the controlling parameter values) are examined critically. A new iteration is then needed. Finally, sensitivity analyses and risk assessment are performed to evaluate if the provided solution(s) is the optimal one(s) in the research space , checking the convergence criteria but also analysing the algorithm's convergence traces. The numerous new evaluations of the objective functions during the optimisation run should be capitalised to improve the assessment of the objective function shape, especially for a new iteration in the

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 December 2019 doi:10.20944/preprints201912.0249.v1

optimisation process (loop to “reshaping the objective function” in Fig 8). This information will be also useful for updating the settings of the algorithm controlling parameters (loop to “Selecting the optimisation algorithm and setting controlling parameters” in Fig 8).

Finally, we propose a template called ODDO for Overview, Description, Design of Optimisation by analogy to the ODD protocol of [38], that can be filled during the optimisation process (Table 1 & Fig 8). This template is a practical form that i) renders the process history in a linear fashion, ii) so that the choices made can be understood in light of the failures and difficulties encountered, iii) and provides enough details for model users to easily reproduce the optimisation, using for instance comment cells. [38,39] have shown the importance of such protocols to provide transparent description of models, to make replication of the process easier and hence less easily dismissed as unscientific.

6 Discussion

Model optimisation is often carried out pragmatically by model users, in many cases by trial-error, without following explicit guidelines, sometimes leading to unreliable solutions [92]. [9] advised model users to “follow the herd” as it is generally wise to use the most popular tools in your research community. Theses authors considered that this strategy reduces the risk of failure of the optimisation process. Practises of MEOW modellers revealed that this behavior was widely shared among users with the behavior of mimicking his/her nearest neighbour. However it is also often accompanied by a feeling of frustration because this “herd” strategy constrains model users to reshape their original optimisation issue in the popular framework. We hope that the provided guidelines could help users to overcome their fears in conducting their optimisation accounting for their specific constraints and objectives.

Another issue often raised by users (and also highlighted at MEOW) was the limited time devoted to the optimisation stage. Because optimisation of complex models is computationally expensive [105], this task is often constrained by time and by the simulation budget, but often botched especially as this occurs at the end of the already long process of model building with high expectation in model outputs. Model users should take time to accurately formulate the optimisation problem, to test new methods, to try out several controlling parameters of algorithms even if it is difficult to quantify the expected gain [80,50]. The lack of automatic pre-processing tools means that model users must wait until they have unsatisfactory optimisation outputs to have a deeper analysis of the problem formulation. More work before running the optimisation would avoid some disappointment at the end of the process.

The implementation of optimisation algorithms turned out to be another issue. The absence of a platform-independent and open-source software bundling most state-of-the-art algorithms with standardised model interfacing methods, too often forces model users to reprogram the selected algorithm within the model framework, making it difficult to test several algorithms [9]. Furthermore, the support documentation for algorithms is often incomplete and very technical especially for setting controlling parameters. Default values provided by the developers are generally used. However, even if these are well-suited for a wide range of applications, exploring alternative settings could be considered as a remedy when the algorithm does not provide satisfactory solution. Model users expressed the need for benchmarks to help in selecting algorithms and metaparameters values. A first step in this direction for problems with continuous variables has been made by the COCO platform ([45], https://github.com/numbbo/coco). However, not covered by COCO is the availability of a free, multi-operating systems toolbox adopted by the numerical optimisation community so as to provide up-to-date optimisation algorithms implementations within a common formalism.

Finally, post-processing could help model users transform failure into success. We strongly encourage authors to document failures of the optimisation process. Optimisation success stories guide model users toward the most popular algorithms [9]. Just like successes, stories of well understood failure will be helpful in selecting the most appropriate pair formulation /

algorithm and prevent model users, for example, from setting metaparameters to “bad” values. We can only regret that the publishing process discourages reports of analyses of optimisation failures.

7 General recommendations

Finally, we would like to provide two sets of recommendations. Firstly, we recommend a number of lines of development for the numerical analysis community.

- Improve existing benchmarks by providing clear, user-friendly selection criteria to assist users in choosing algorithms [87]. The benchmarks should not only be based on closed-form, simple, mathematical functions but also on prototypes of complex ecological models. This will require the definition of a standardized collection of pseudo-complex ecological models (e.g. inspired from some in Annex 1), like the “Stupid Models” defined for benchmarking agent-based modelling platforms [97], that are easily portable to different frameworks and for which the numerical behaviour is documented well enough to avoid failures during the optimisation process. In addition, the benchmarked algorithms should come with a clear documentation of the effects of their metaparameters.
- Develop one or many toolboxes for post-processing optimisation results (solution point diagnosis, global optimisation analysis, and convergence analysis).
- Move toward more adaptive algorithms that could take into account more knowledge gained at each iteration of the algorithm (e.g. adapting algorithms parameters using post-processing diagnostics during the optimisation phase) .
- Move towards a better integration of the various tools to facilitate the use of methods without having to reformat data or redefine problems [9].

Secondly, we make a number of recommendations for model users who are setting out to optimise complex models.

- Do not concentrate on the choice of algorithm to the detriment of the pre- and post-processing.
- Exploit the full optimisation trace, which should be made more easily accessible by algorithm developers, and not just the final results.
- Acknowledge the utility of an *iterative* optimisation process to improve your results, even if it can be seen as time and effort consuming in a first time..
- Provide more details about how the optimisation was carried out in subsequent publications, filling the ODDO template (Table 1), to enhance the trust in the resulting complex ecological models on the one hand, and, on the other hand, to enrich the understanding of optimisation process.

Acknowledgements

We would like to thank the MIA (Applied Mathematics and Informatics) department of Inra, the RNSC (French National Network for Complex Systems), Ifremer, Irstea and Cirad for their financial support to the French methodological research network in environmental science and ecology MEXICO (Méthodes pour l'EXploration Informatique de modèles COmplexes). This work emerged from fruitful discussions during the MEOW workshop organised by MEXICO. We deeply appreciate sharing experience of optimisation with MEOW participants (Thierry Faure, Eric Gozé, Franck Jabot, Romain Lardy, Hervé Monod, David Ojeda, Thibaud Rougier, Darietou Sambakhe, Marie Savina-Rolland, Morgane Travers-Trolet, Ronan Trepos, Audric Vigier). Finally, we would also like to thank Verena Trenkel and Vincent Badts for helpful comments on the ODDO template, and Olivier Berthelé for improving the design of post-processing Figures.

References

1. Amaran, S., Sahinidis, N. V., Sharda, B., & Bury, S. J.. Simulation optimization: a review of algorithms and applications. Annals of Operations Research, 2016, 240(1), 351-380.
2. Audet, C., and Dennis Jr, J. E.. Mesh adaptive direct search algorithms for constrained optimization. SIAM Journal on optimization, 2006, 17(1), 188-217.
3. Baluja, S.. Population-based incremental learning. a method for integrating genetic search based function optimization and competitive learning. Technical report, 1994,No. CMU-CS-94-163. Carnegie-Mellon University, Pittsburgh PA, USA, Dept Of Computer Science.
4. Banos, A., Sansders, L. Modéliser et simuler les systèmes spatiaux en géographie. in : Varenne F., Silberstein M. (dir), Modéliser et Simuler - Epistémologies et pratiques des modèles et des simulations, Paris, Matériologiques, 833-863. 2013.
5. Beume, N., Naujoks, B. and Emmerich, M. SMS-EMOA: Multiobjective selection based on dominated hypervolume. *European Journal of Operational Research* 2007, 181, no. 3 (2007): 1653-1669.
6. Birgin, E. G., Martínez, J. M., & Raydan, M. Algorithm 813: SPG—software for convex-constrained optimization. *ACM Transactions on Mathematical Software (TOMS)*, 2001, *27*(3), 340-349.
7. Blower, S.M. and Dowlatabadi, H. Sensitivity and Uncertainty Analysis of Complex Models of Disease Transmission. International Statistical Review, 1994, 62, 229-243.
8. Blum, Michael GB, Maria Antonieta Nunes, Dennis Prangle, Scott A. Sisson, et others. A comparative review of dimension reduction methods in approximate Bayesian computation . *Statistical Science* 2013,28, n^{o} 2 : 189–208.
9. Bolker, Benjamin M., Beth Gardner, Mark Maunder, Casper W. Berg, Mollie Brooks, Liza Comita, Elizabeth Crone, et al. Strategies for Fitting Nonlinear Ecological Models

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 December 2019 doi:10.20944/preprints201912.0249.v1

in R, AD Model Builder, and BUGS . *Methods in Ecology and Evolution 2013*, 4, n^{o} 6: 501–512. doi:10.1111/2041-210X.12044.

10. Booker, A. J., J. E. Dennis Jr, P. D. Frank, D. B. Serafini, V. Torczon, et M. W. Trosset. A. Rigorous Framework for Optimization of Expensive Functions by Surrogates . *Structural Optimization* 1999, 17, n^{o} 1 (février 1999): 1–13. doi:10.1007/BF01197708.
11. Boussaïd, I., Lepagnot, J. and Siarry, P. A survey on optimization metaheuristics. *Inf. Sci*, 2013. 237, 82-117
12. Brochu, E., Cora, V. M. and De Freitas, N.. A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. *arXiv 2010, preprint arXiv:1012.2599*.
13. Brynjarsdóttir, J. and O'Hagan, A. Learning about physical parameters: The importance of model discrepancy.Inverse Problems 2014, 30, 114007 (24pp)
14. Coello, Carlos A. Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art." *Computer methods in applied mechanics and engineering* 2002, 191, no. 11-12 : 1245-1287.
15. Csilléry, K., Blum, M. G., Gaggiotti, O. E., & François, O. Approximate Bayesian computation (ABC) in practice. *Trends in ecology & evolution*, 2010, *25*(7), 410-418.
16. Deb, K. (2001). *Multi-objective optimization using evolutionary algorithms* (Vol. 16). John Wiley & Sons.
17. Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. A. M. T. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE transactions on evolutionary computation*, 2002, *6*(2), 182-197.
18. Deb, K., and Sundar, J. Reference point based multi-objective optimization using evolutionary algorithms . In *Proceedings of the 8th annual conference on Genetic and*

evolutionary computation, 2006, 635–642. ACM, 2006. http://dl.acm.org/citation.cfm?id=1144112.

19. Deriso, R B, M N Maunder, et J R Skalski. Variance estimation in integrated assessment models and its importance for hypothesis testing . *Canadian Journal of Fisheries and Aquatic Sciences* 2007, 64, n^{o} 2 : 187–197.
20. Drouineau, H, S Mahévas, M Bertignac, et D Duplisea. A length-structured spatially explicit model for estimating hake growth and migration rates. *ICES Journal of Marine Science* 67, n^{o} 8 (2010): 1697–1709.
21. de Valpine, P. Review of methods for fitting time-series models with process and observation error and likelihood calculations for nonlinear, non-gaussian state space models. *Bulletin of* Marine Science 2002, 70, no 2 : 455–471.
22. Edmonds, B. and Moss, S.J. From Kiss to KIDS - an 'antisimplistic' modelling approach;in : P. Davidsson et al. (eds.): Multi Agent Based Simulation; Springer, Lecture Notes in Artificial Intelligence, 2004,3415: p.130-144.
23. Efstratiadis, A., and Koutsoyiannis, D. One decade of multi-objective calibration approaches in hydrological modelling: a review . Hydrological Sciences Journal–Journal des Sciences Hydrologiques 2010, 55, no 1: 58–78.
24. Evans, M.R., Grimm, V., Johst, K., Knuuttila, T., de Langhe, R., Lessells, C.M., Merz, M., O'Malley, M.A., Orzack, S.H., Weisberg, M., Wilkinson, D.J., Wolkenhauer,O, & Benton, T.G. Do simple models lead to generality in ecology? Trends in Ecology and Evolution 2013, 28:578-583.
25. Faivre, R, B Iooss, S Mahévas, D Makowski, et H Monod. *Analyse de sensibilité et exploration de modèles*. Quae, 2013.
26. Fearnhead, Paul, et Dennis Prangle. Constructing summary statistics for approximate Bayesian computation: semi-automatic approximate Bayesian computation. *Journal of*

the Royal Statistical Society: Series B (Statistical Methodology) 2012, 74, n^{o} 3: 419–474.

27. Fleming, Peter J., Robin C. Purshouse, et Robert J. Lygoe. Many-objective optimization: An engineering design perspective . In *Evolutionary multi-criterion optimization*, 14–32. Springer, 2005. http://link.springer.com/chapter/10.1007/978-3-540-31880-4_2.
28. Fournier, D A, J R Sibert, J Majkowski, et H Hampton. MULTIFAN a likelihood-based method for estimating growth parameters and age composition from multiple length frequency data sets illustrated using data for southern bluefin tuna (*Thunnus* maccoyii). *Canadian Journal of Fisheries and Aquatic Sciences* 1990, 47, n^{o} 2: 301–317.
29. Fournier, David A., Hans J. Skaug, Johnoel Ancheta, James Ianelli, Arni Magnusson, Mark N. Maunder, Anders Nielsen, et John Sibert. AD Model Builder: using automatic differentiation for statistical inference of highly parameterized complex nonlinear models. *Optimization Methods and Software* 2012, 27, n^{o} 2: 233–249. doi:10.1080/10556788.2011.597854.
30. Francis, R.I.C. Chris. Data weighting in statistical fisheries stock assessment models . *Canadian Journal of Fisheries and Aquatic Sciences* 2011, 68, n^{o} 6: 1124–1138. doi:10.1139/f2011-025.
31. Gasche, L., Mahévas, S., et Marchal, P. Supporting fisheries management by means of complex models: can we point out isles of robustness in a sea of uncertainty ? Plos One 2013, 8(10)
32. Giagkiozis, I. and P. J. Fleming. Pareto Front Estimation for Decision Making. *Evolutionary Computation* 2014, 22, n^{o} 4 651–678.

33. Gill, P. E., Murray, W. and Wright,M. H. *Practical optimization*. Academic press, 1981.
34. Gimenez, O, A Viallefont, E A Catchpole, R Choquet, et B J T Morgan. Methods for investigating parameter redundancy. *Animal Biodiversity and Conservation* 2004, 27, n^{o} 1 : 561–572.
35. Grimm, V., Frank, K., Jeltsch, F., Brandl, R., Uchmań ski, J.,Wissel, C., Pattern-oriented modelling in population ecology. Sci. Total. Environ. 1996, 183, 151 – 166.
36. Grimm, V., Revilla, E., Berger, U., Jeltsch, F., Mooij, W.M., Railsback, S.F., Thulke, H.-H., Weiner, J., Wiegand, T. and DeAngelis, D. L. Pattern-Oriented Modeling of Agent-Based Complex Systems: Lessons from Ecology. *Science* 2005, 310, n^{o} 5750: 987.
37. Grimm, V. and Railsback, S.F. Individual-Base Modeling and Ecology. Princeton University Press: Princeton. 2005.
38. Grimm, V., Berger, U., Bastiansen, F., Eliassen, S., Ginot, V., Giske, J., Goss-Custard,J., Grand, T., Heinz, S., Huse, G., Huth, A., Jepsen, J.U., Jørgensen, C., Mooij, W.M.,Müller, B., Pe'er, G., Piou, C., Railsback, S.F., Robbins, A.M., Robbins, M.M., Ross-manith, E., Rüger, N., Strand, E., Souissi, S., Stillman, R.A., Vabø, R., Visser, U.,DeAngelis, D.L.. A standard protocol for describing individual-based andagent-based models. Ecol. Model. 2006, 198, 115–126.
39. Grimm V, Berger U, DeAngelis D L, Polhill J G, Giske J and Railsback S F. The ODD protocol: A review and first update. *Ecological Modelling* 2010, 221 (23), 2760-2768.
40. Grunert da Fonseca, V., Fonseca, C. M., and Hall, A.O. Inferential performance assessment of stochastic optimisers and the attainment function. In International Conference on Evolutionary Multi-Criterion Optimization - EMO 2001, pages 213-225. Springer, Berlin, Heidelberg.

41. Hansen, N and Ostermeier, A. Completely Derandomized Self-Adaptation in Evolution Strategies. Evolutionary Computation, 2001, 9(2), pp. 159-195

42. Hansen, N, Müller, S. D. and Koumoutsakos,P. . Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES). Evolutionary computation 2003, 11, no 1: 1–18.

43. Hansen, N, Auger, A., Ros, R., Finck, S. and Pošík, P. Comparing results of 31 algorithms from the black-box optimization benchmarking BBOB-2009. In Proceedings of the 12th annual conference companion on Genetic and evolutionary computation, 2010, 1689-1696. ACM.

44. Hansen, N. A CMA-ES for mixed-integer nonlinear optimization. INRIA research report 2011, RR-7751, 2011

45. Hansen, N., Auger, A., Mersmann, O., Tusar, T., & Brockhoff, D. COCO: A platform for comparing continuous optimizers in a black-box setting. *arXiv preprint* 2016, *arXiv:1603.08785*.

46. Harik, G. R. and Fernando G. Lobo: A Parameter-less Genetic Algorithm . In *Proceedings of the 1st Annual Conference on Genetic and Evolutionary Computation-1999, Volume 1*, pp. 258-265. Morgan Kaufmann Publishers Inc.

47. Hastie, T., Tibshirani, R., Friedman, J. H.. The Elements of Statistical Learning (second ed.). Springer-Verlag.2009, ISBN 978-0-387-84858-7.

48. Hemez, F. M. and Y. Ben-Haim, Info-gap robustness for the correlation of tests and simulations of a nonlinear transient, *Mechanical Systems and Signal Processing*, 2004, vol. 18, #6, pp.1443-1467.

49. Hoos, H. and Stützle, T. Evaluating Las Vegas Algorithms: Pitfalls and Remedies . In *Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence,* 1998, pp. 238-245. Morgan Kaufmann Publishers Inc.

50. Hoos, H. Automated algorithm configuration and parameter tuning. In *Autonomous search*, 2011, pages 37-71. Springer, Berlin, Heidelberg.

51. Horst, R., & Tuy, H. *Global optimization: Deterministic approaches*. Springer Science & Business Media. 2013.

52. Hulson, Peter-John F., Sara E. Miller, Terrance J. Quinn, Gary D. Marty, Steven D. Moffitt, et Frederick Funk. Data Conflicts in Fishery Models: Incorporating Hydroacoustic Data into the Prince William Sound Pacific Herring Assessment Model. *ICES Journal of Marine Science:* 2008, 65, no 1: 25–43. doi:10.1093/icesjms/fsm162.

53. Huyer, W., and A. Neumaier. Global optimization by multilevel coordinate search. Journal of Global Optimization 1999, 14, no. 4: 331-355.

54. Huyer, W., and Neumaier. A., SNOBFIT--stable noisy optimization by branch and fit." ACM Transactions on Mathematical Software (TOMS) 2008, 35, no. 2: 9.

55. Iman, R.L. and Conover, W. J. Small sample sensitivity analysis techniques for computer models, with an application to risk assessment. Communication in Statistics. 1980, A9 (17). 1749-1842pp.

56. Ishibuchi, H., Tsukamoto, N. and Nojima, Y. Evolutionary many-objective optimization: A short review. In Evolutionary Computation, CEC 2008. IEEE World Congress on Computational Intelligence, 2008, pages 2419–2426.

57. Jones, D. R., Perttunen, C. D., & Stuckman, B. E.. Lipschitzian optimization without the Lipschitz constant. *Journal of Optimization Theory and Applications*, 1993 .*79*(1), 157-181.

58. Jones, D. R. A taxonomy of global optimization methods based on response surfaces. *Journal of global optimization* 2001, 21, n^{o} 4: 345–383.

59. Kasprzyk, J. R., Nataraj, S., Reed, P. M. and Lempert, R. J. Many objective robust decision making for complex environmental systems undergoing change .

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 December 2019 doi:10.20944/preprints201912.0249.v1

Environmental Modelling & Software 2013, 42: 55-71

60. Keijzer, M., Merelo, J.J. , Romero, G., Schoenauer, M. Evolving objects: A general purpose evolutionary computation library , Artificial Evolution 2002, 2310, 829-888.

61. Kendall, W. L., and Nichols, J. D. Estimating State-Transition Probabilities for Unobservable States Using Capture-Recapture/Resighting Data. *Ecology*. 2002, 83, n^{o} 12: 3276–3284. doi:10.2307/3072078.

62. Kennedy, J. Particle swarm optimization. In *Encyclopedia of machine learning* 2011, (pp. 760-766). Springer US.

63. Kleijnen, J. P. C. and Sargent, R. G. A methodology for fitting and validating metamodels in simulation1. *European Journal of Operational Research* 2000, 120, n^{o} 1: 14–29.

64. Knape, J. and de Valpine. P. Fitting complex population models by combining particle filters with Markov chain Monte Carlo. *Ecology* 2011, 93, n^{o} 2: 256–263.

65. Kraft, Di. Package for sequential quadratic programming. Forschungsbericht Deutsche Forschungs- und Versuchsanstalt fur Luft- und Raumfahrt, Oberpfaffenhofen, Germany, 1988.

66. Krause, O, Glasmachers, T., Hansen, N. and Igel, C. Unbounded population MO-CMA-ES for the bi-objective BBOB test suite. In *Proceedings of the 2016 on Genetic and Evolutionary Computation Conference Companion*, 2016, pp. 1177-1184. ACM.

67. Kumar,U., Ma. Rebecca Laza, Jean-Christophe Soulié, Richard Pasco, Kharla V.S. Mendez, et Michael Dingkuhn. Compensatory phenotypic plasticity in irrigated rice: Sequential formation of yield components and simulation with SAMARA model. *Field Crops* Research, 2016, 193: 164–177.

68. Lardy, R., Mazzega, P., Sibertin-Blanc, C., Auda, Y., Sanchez-Perez, J.S., Sauvage, S.,

Therond, O.. Calibration of simulation platforms including highly interweaved processes: the MAELIA multi-agent platform. In Daniel P. Ames, Nigel W.T. Quinn and Andrea E. Rizzoli (Eds.). 7th Int. Congress on Env. Modelling and Software 2014, San Diego, CA, USA, 8 pp

69. Larrañaga, P., & Lozano, J. A. (Eds.). Estimation of distribution algorithms: A new tool for evolutionary computation (Vol. 2). Springer Science & Business Media. 2001.

70. Lebart, L., Morineau, A. and Piron, M. Statistique Exploratoire Mutidimensionnelle. Édité par Dunod. 2nd Edition.1997.

71. Lehuta, S., Mahévas, S., Petitgas, P., et LeFloch, P. A simulation-based approach to assess sensitivity and robustness of fisheries management indicators for the pelagic fishery in the Bay of Biscay. Can. J. Fish. Aquat. Sci. 2013, 70, 1741–1756

72. Lehuta Sigrid, Girardin Raphael, Mahevas Stephanie, Travers-Trolet Morgane, Vermard Youen. Reconciling complex system models and fisheries advice: Practical examples and leads . Aquatic Living Resources 2016, 29(2), 208 (1-20).

73. Lenormand, M., Jabot, F. and Deffuant, G. Adaptive Approximate Bayesian Computation for Complex Models. *Computational Statistics* 2013, 28, n^{o} 6 : 2777–2796.

74. Le Riche, R., and Guyon, F. Least Squares Parameter Estimation and the Levenberg-Marquardt Algorithm : Deterministic Analysis, Sensitivities and Numerical Experiments , Technical Report 1999, no. 041/99, Laboratoire de Mécanique de Rouen, INSA de Rouen.

75. Levins, R. The strategy of model building in population biology. American Scientist, 1966, 54(4), 421-431

76. Liu, D. C., and Nocedal, J. On the limited memory BFGS method for large scale optimization. Mathematical programming, 1989, 45(1), 503-528.

77. Loshchilov, I., Schoenauer, M. and Sebag, M. Intensive Surrogate Model Exploitation in Self-Adaptive Surrogate-Assisted Cma-Es (Saacm-Es). In Proceeding of the Fifteenth Annual Conference on Genetic and Evolutionary Computation Conference - GECCO '13, 439. Amsterdam, The Netherlands: ACM Press. 2013.

78. Loshchilov, I. and Glasmachers, T. Anytime Bi-Objective Optimization with a Hybrid Multi-Objective CMA-ES (HMO-CMA-ES). In Proceedings of the 2016 on Genetic and Evolutionary Computation Conference Companion, 2016, pp. 1169-1176. ACM.

79. Luersen, M. A., and Le Riche, R. Globalized Nelder–Mead method for engineering optimization. *Computers & structures*, 2004, *82*(23), 2251-2260.

80. Maier, H. R., Zoran Kapelan, J. Kasprzyk, J. Kollat, L. S. Matott, M. C. Cunha, G. C. Dandy, et al. Evolutionary algorithms and other metaheuristics in water resources: current status, research challenges and future directions . *Environmental Modelling & Software* 2014, 62: 271–299.

81. Mahévas, S., et Iooss, B. Grilles de sélection d'une méthode d'analyse de sensibilité globale. In *Analyse de sensibilité et exploration de modèles - Applications aux modèles environnementaux*. Quae. Ed. Robert Faivre, Bertrand Iooss,David Makowski, Stéphanie Mahévas, Hervé Monod. 2013.

82. Mahévas, S., Pelletier D., ISIS-Fish, a generic and spatially-explicit simulation tool for evaluating the impact of management measures on fisheries dynamics. Ecological. Modelling. 2004,Vol 171/1-2 pp 65-84.

83. Mahévas, S. , Picheny, V., Lambert, P., Dumoulin, N., Rouan, L., Soulié, J.-C. Document de restitution de l'atelier Mexico, MExico Optimisation Workshop 2014, MEOW, 4-6 novembre 2014, La Rochelle. , http://reseau-mexico.fr/sites/reseau-mexico.fr/files/BookSynth%C3%A8seJanv2015.pdf

84. Manache, G. and Melching, C.S. Sensitivity analysis of a water-quality model using

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 December 2019 doi:10.20944/preprints201912.0249.v1

Latin hypercube sampling. J. Water Resour. Plann. Manag., 2004, 130 (3) pp. 232-242

85. Marin, J.-M., Pillai,N. S., Robert, C. P., Rousseau, J. Relevant statistics for Bayesian model choice. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 2014, 76, n^{o} 5 : 833–859.

86. Marrel, A., Iooss, B., Da Veiga, S., Ribatet, M. Global Sensitivity Analysis of Stochastic Computer Models with Joint Metamodels. *Statistics and Computing* 2012, 22, n^{o} 3: 833–847.

87. Matott, L. S., Tolson, B. A. and Asadzadeh, M. A Benchmarking Framework for Simulation-Based Optimization of Environmental Models . *Environmental Modelling & Software* 2012, 35 : 19-30.

88. Maunder, M. N. and Piner, K. R. (2017). Dealing with data conflicts in statistical inference of population assessment models that integrate information from multiple diverse data sets . *Data conflict and weighting, likelihood functions, and process errorData weighting* 192: 16-27.

89. McCullagh, P., et J. A. Nelder. *Generalized Linear Models*. Chapman and Hall., 1989.

90. Nelder, J. A., & Mead, R. A simplex method for function minimization. The computer journal, 1965, 7(4), 308-313.

91. Ojeda,D., Le Rolle, V., Harmouche, M., Drochon, A., Corbineau, H., Verhoye, J.-P., Hernandez, A. I. Sensitivity Analysis and Parameter Estimation of a Coronary Circulation Model for Triple-Vessel Disease. IEEE transactions on biomedical engineering 2014, 61, no 4 : 1208–1219.

92. Peng, R. D.. Reproducible research and Biostatistics ». Biostatistics 2009, 10, no 3 : 405-8.

93. Picheny, V. ; Trépos, R. ; Casadebaig, P. Optimization of black-box models with uncertain climatic inputs. Application to sunflower ideotype design. PLOS One. 2017.

12(5): e0176815

94. Ponweiser, W., Wagner, T., Biermann, D., Vincze, M. Multiobjective Optimization on a Limited Budget of Evaluations Using Model-Assisted $\mathcal {S} $-Metric Selection. In International Conference on Parallel Problem Solving from Nature, 2008, pp. 784-794. Springer, Berlin, Heidelberg.
95. Powell, M. J. A direct search optimization method that models theobjective and constraint functions by linear interpolation, in Advances inOptimization and Numerical Analysis, eds. S. Gomez and J-P. Hennart,Kluwer Academic (Dordrecht), 1994, pp. 51–67.
96. Powell, M. J. The NEWUOA software for unconstrained optimization without derivatives. In Large-scale nonlinear optimization, 2006, (pp. 255-297). Springer US.
97. Railsback, S. F., Lytinen, S. L., Jackson, S. K. Agent-based Simulation Platforms: Review and Development Recommendations. 2006, SIMULATION 82, no 9 : 609-23.
98. Rios, L. M., and N. V. Sahinidis Derivative-free optimization: a review of algorithms and comparison of software implementations. Journal of Global Optimization 2013, 56, no 3: 1247–1293.
99. Rougier, T., Drouineau, H., Dumoulin, N., Faure, T., Deffuant, G., Rochard, E., Lambert, P. The GR3D model, a tool to explore the Global Repositioning Dynamics of Diadromous fish Distribution. Ecological Modelling 2014, 283 : 31–44.Rougier, T., Lassalle, G., Drouineau, H., Dumoulin, N. Faure, T., Deffuant, G., Rochard, E., Lambert, P. The Combined Use of Correlative and Mechanistic Species Distribution Models Benefits Low Conservation Status Species. PLoS ONE 2015, 10, no 10 : e0139194.
100. Saltelli, A., Tarantola, S., Campolongo, F. Sensitivity analysis as an ingredient of modelling ». Statistical Science, 2000, 15(4): 377-95.

101. Sambakhe D., Rouan L., Bacro J.N., Gozé E. Journal of Global Optimization, 2019, 73 (3) : p. 615-636.

102. Schaub, M., Gimenez, O., Schmidt, B. R., Pradel, R. Estimating Survival and Temporary Emigration in the Multistate Capture–Recapture Framework. Ecology, 2004, 85, no 8 : 2107–2113.

103. Shackell, N L, W T Stobo, K T Franck, et D Brickman. Growth of cod (Gadus morhua) estimated from mark-recapture programs of the Scotian Shelf and adjacent areas . ICES Journal of Marine Science 1997, 54, no 3 : 383–398.

104. Shan, S. and G. G. Wang. Survey of modeling and optimization strategies to solve high-dimensional design problems with computationally-expensive black-box functions. Structural and Multidisciplinary Optimization, 2010, 41, no 2 : 219–241.

105. Shin, Y.-J. and P. Cury. Using an individual-based model of fish assemblages to study the response of size spectra to changes in fishing. Canadian Journal of Fisheries and Aquatic Sciences 2004, 61, no 3 : 414–431.

106. Silvey, S. D. Statistical inference. Vol. 7. CRC Press. 1975.

107. Sterman, J. D. Appropriate summary statistics for evaluating the historical fit of system dynamics models. Dynamica, 1984, 10 (Winter), 51–66

108. Sudheer, K. P., Chaubey, I., Garg, V., Migliaccio,K. W. Impact of time-scale of the calibration objective function on the performance of watershed models. Hydrological Processes 2007, 21, no 25: 3409-19.

109. Thiele, L., Miettinen, K., Korhonen, P. J., Molina, J. A preference-based interactive evolutionary algorithm for multiobjective optimization. 2007. http://www.diva-portal.org/smash/record.jsf?pid=diva2:498868.

110. Van Laarhoven, P. J., and Aarts, E. H. Simulated annealing. In Simulated annealing: Theory and applications (pp. 7-15). Springer Netherlands. 1987.

111. Varelas, Konstantinos, Anne Auger, Dimo Brockhoff, Nikolaus Hansen, Ouassim Ait ElHara, Yann Semet, Rami Kassab, and Frédéric Barbaresco. A comparative study of large-scale variants of CMA-ES. In Parallel Problem Solving from Nature 2018, pp. 3-15. Springer.

112. Viallefont, A, J D Lebreton, A M Reboulet, et G Gory. Parameter identifiability and model selection in capture-recapture models: A numerical approach. Biometrical journal 1998, 40, no 3: 313–325.

113. Villemonteix, J., Vazquez, E. Walter, E. An Informational Approach to the Global Optimization of Expensive-to-Evaluate Functions. Journal of Global Optimization 2009, 44, no 4 : 509-34.

114. Walter, E., and Pronzato, L. Identification of parametric models from experimental data. Springer Verlag. 1997.

115. Wolpert,D. H., and Macready,W. G. No free lunch theorems for optimization. IEEE transactions on evolutionary computation 1997, 1.1 : 67-82.

116. Wong, C. S. Y., Al-Dujaili, A., Sundaram, S. Hypervolume-based DIRECT for multi-objective optimisation." In Proceedings of the 2016 on Genetic and Evolutionary Computation Conference Companion, 2016, pp. 1201-1208. ACM

117. Zhang, X., Srinivasan, R., Zhao, K., Van Liew, M. Evaluation of Global Optimization Algorithms for Parameter Calibration of a Computationally Intensive Hydrologic Model. Hydrological Processes 2009, 23, no 3 : 430-41.

118. Razavi, S., Tolson, B. A., Burn, D. H. (2012) Numerical Assessment of Metamodelling Strategies in Computationally Intensive Optimization . Environmental Modelling & Software 34: 67-86.

119. Matott, L. S., Tolson, B. A. and Asadzadeh, M. (2012) A Benchmarking Framework for Simulation-Based Optimization of Environmental Models.

Environmental Modelling & Software 35 : 19-30.

120. John Tukey, John Wilder, Exploratory Data Analysis, Addison-Wesley, 1977

121. Du Toit, S. H. C., Steyn, A. G. W., Stumpf, R. H. Graphical Exploratory Data Analysis. Springer-Verlag. 1986

122. Belotti, P., Kirches, C., Leyffer, S., Linderoth, J., Luedtke, J., & Mahajan, A. Mixed-integer nonlinear optimization. Acta Numerica, 2013, 22, 1-131. Supporting Information

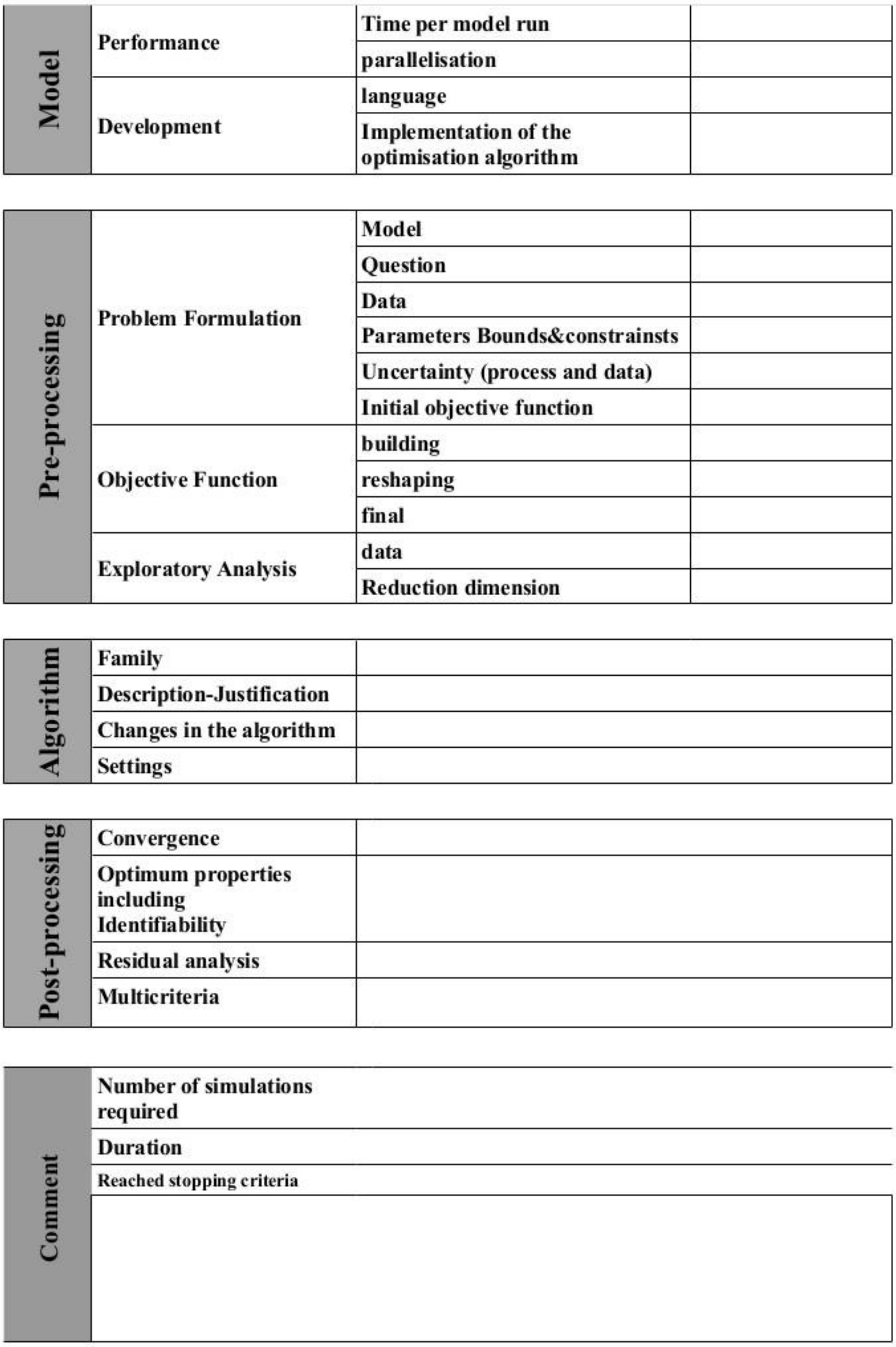
123. Wood, Simon N. Partially specified ecological models. Ecological Monographs, 2001, 71,1: 1-25.

124. Conn, Paul B., Devin S. Johnson, Perry J. Williams, Sharon R. Melin, and Mevin B. Hooten. A Guide to Bayesian Model Checking for Ecologists. Ecological Monographs, 2018, 88, 4 : 526-42.

125. Hauenstein, Severin, Julien Fattebert, Martin U. Grüebler, Beat Naef-Daenzer, Guy Pe'er, and Florian Hartig. Calibrating an Individual-based Movement Model to Predict Functional Connectivity for Little Owls. Ecological Applications, 2019, 29, 4 : e01873.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 December 2019 doi:10.20944/preprints201912.0249.v1

Table 1 : ODDO (Overview, Design, Details of Optimisation) template to report an optimisation project



Model	Performance	Time per model run	
		parallelisation	
	Development	language	
		Implementation of the optimisation algorithm	

Pre-processing	Problem Formulation	Model	
		Question	
		Data	
		Parameters Bounds&constrainsts	
		Uncertainty (process and data)	
		Initial objective function	
	Objective Function	building	
		reshaping	
		final	
	Exploratory Analysis	data	
		Reduction dimension	

Algorithm	Family	
	Description-Justification	
	Changes in the algorithm	
	Settings	

Post-processing	Convergence	
	Optimum properties including Identifiability	
	Residual analysis	
	Multicriteria	

Comment	Number of simulations required
	Duration
	Reached stopping criteria

List of Figures

Fig 7: Decision tree of the third post-processing step described in section 4.3.2 and 4.3.3 to point out identifiability problems and to perform the residuals analysis.

Fig 8: Workflow of complex model-based optimisation. It is a multi-stage process including prerequisites, three compulsory steps (pre-processing, optimisation *per se* and post-processing) and a final critical analysis. Black arrows show where in the process, components of the original optimisation problem are called into question and may be updated.

Fig 9: Plot of the nine optimisation case studies detailed in Table 1, against the number of parameters to be optimised and the number of allowable model simulations. The triangle symbol indicates multiple objectives while a star indicates a single objective function, the circle indicates constraints on the parameters , the square indicates discrete valued parameters and the star indicates a stochastic model.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 December 2019 doi:10.20944/preprints201912.0249.v1

Annex 1 : feedback from practitioners in ecological modelling

MEXICO[1] (Méthodes pour l'EXploratIon Informatique de modèles COmplexes)MEXICO (Méthodes pour l'EXploration Informatique de modèles COmplexes) is a French methodological research network in environmental science and ecology. It was created in 2006 by methodologists (applied mathematicians and computer scientists), modellers and model users to provide rigorous methods for analysing and using complex models. The group initially focused on sensitivity analysis [25] and extended its scope to all themas related to computer experiments and metamodelling. MEXICO is a multi-institute network; it is supported by the MIA (Applied Mathematics and Informatics) department of Inra, received funds also from RNSC (French National Network for Complex Systems) and occasionally by Ifremer, Irstea and Cirad. Questions arising from model users concerning optimisation of their complex models (estimation/calibration of parameters, optimisation of decision rules) pushed the group to organise a workshop to address this problem (MEOW, MExico Optimisation Workshop 2014 https://reseau-mexico.fr/node/290 , [83]). This workshop aimed to i) review the experience of model users, ii) present the mathematical basis of the most commonly used optimisation algorithms, and iii) provide a forum for discussing questions raised by modellers and solutions proposed by numerical methods specialists.

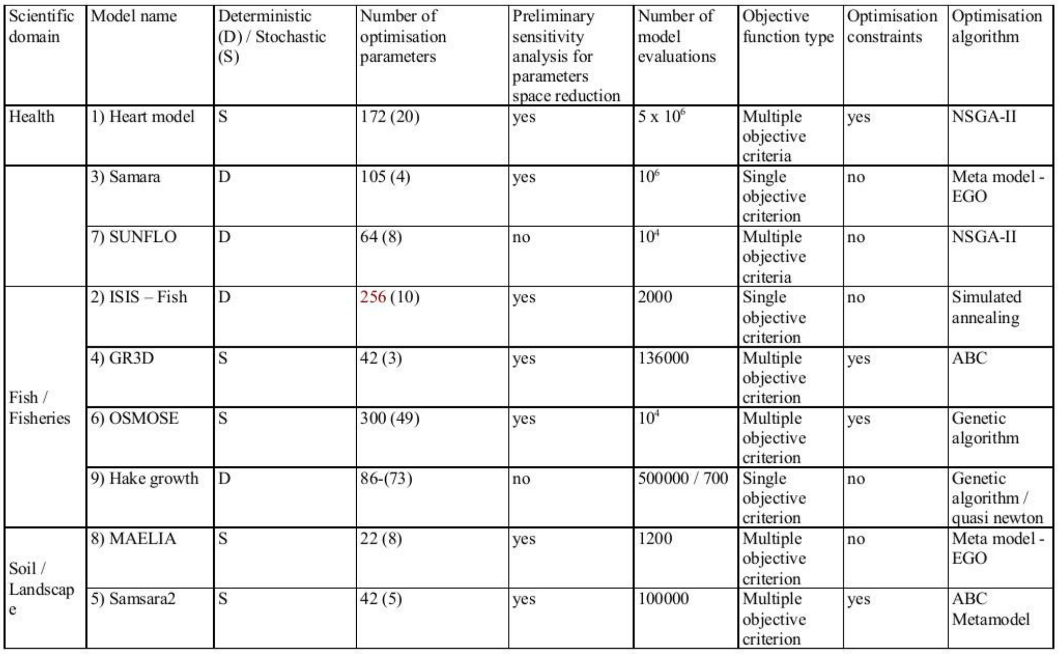
Nine case studies of optimisation in the health [91], plants and forests [93,102] , fisheries [82, 106, 20, 99,100] and soil and landscape [68] domains were presented using models with a wide

[1] 1Methods for numeric EXploratIon of COmplex models

range of modelling paradigms and approaches (stochastic / deterministic, dynamic / static, agent-based / mechanistic, cf. Table 1). The characteristics of the optimisation configurations presented during MEOW are thought to be representative of the diversity of optimisation problems encountered in complex systems modelling (e.g. [23]). Most participants had resorted to optimisation to estimate parameters to fit the model to observed data, either because they were directly interested by the value of the parameters or to enhance the predictive ability of their model. Most of the models involved in these studies were treated as black-boxes. We have primarily classified the optimisation methods used according to the number of parameters to be estimated and the number of model evaluations performed, which are the two main characteristics readily available for black-box models (Fig 1-Annex1). The number of evaluations varied from thousands to millions. The number of parameters to be estimated was between 5 and 73, representing sometimes less than 5% of the total number of model parameters. Four other questions have been identified as critical to the definition of the optimisation problem and the choice of the optimisation algorithm [87,59]: i) Does the objective function have one or many criteria? ii) Is the model deterministic or stochastic? iii) Are the parameter values continuous, discrete or mixed? iv) Are there constraints on these parameters and on internal or output variables? Fig 1 shows these four significant features in the test cases studied. Firstly, there was no case with discrete valued parameters although some models used these. Secondly, there were few optimisation problems subject to constraints. Thirdly, multi-criteria objective optimisation is dominant (which may explain the small number of constrained problems) but the objective function was often reduced to a single criterion by aggregation. Fourthly, the number of parameters was always less than hundred, lower than the total number of parameters in the models.

Fig 9.

Table 1-Annex1. Models and optimisation configurations for the MEOW case studies



Scientific domain	Model name	Deterministic (D) / Stochastic (S)	Number of optimisation parameters	Preliminary sensitivity analysis for parameters space reduction	Number of model evaluations	Objective function type	Optimisation constraints	Optimisation algorithm
Health	1) Heart model	S	172 (20)	yes	5 x 10^6	Multiple objective criteria	yes	NSGA-II
	3) Samara	D	105 (4)	yes	10^6	Single objective criterion	no	Meta model - EGO
	7) SUNFLO	D	64 (8)	no	10^4	Multiple objective criteria	no	NSGA-II
Fish / Fisheries	2) ISIS – Fish	D	256 (10)	yes	2000	Single objective criterion	no	Simulated annealing
	4) GR3D	S	42 (3)	yes	136000	Multiple objective criterion	yes	ABC
	6) OSMOSE	S	300 (49)	yes	10^4	Multiple objective criterion	yes	Genetic algorithm
	9) Hake growth	D	86-(73)	no	500000 / 700	Single objective criterion	no	Genetic algorithm / quasi newton
Soil / Landscape	8) MAELIA	S	22 (8)	yes	1200	Multiple objective criterion	no	Meta model - EGO
	5) Samsara2	S	42 (5)	yes	100000	Multiple objective criterion	yes	ABC Metamodel

MEOW revealed that particular attention was paid to the formulation of the optimisation problem, but that the selection of the algorithms was more arbitrary. Often the choice of algorithm was dictated by the model user's previous experience and network rather than by an integrated problem and method evaluation. Moreover, the performance of the algorithms and, more specifically, the quality of the solutions were difficult to assess due to a lack of practical evaluation methods. At the end of the optimisation process, model users were generally left with doubts about whether the solution was optimal and whether the objective function was an adequate representation of the problem, in particular when the convergence was sensitive to the initial conditions. Another observation arising from the discussions is that tuning the optimisation algorithms was not a straight path and often required several trials. Many pragmatic changes that were not immediately obvious were made during the process, for example: changing the algorithm used, adding constraints to the algorithm, changing weights in the objective function, and changing some algorithm settings. Finally, this workshop highlighted that, despite the absence of practical guidelines for optimisation, model users obtained acceptable solutions to their optimisation problems using already known or simplified formulations. Nevertheless, these default formulations did not always reflect the model users' initial question. Actually when a default formulation or method did not match the initial goal, model users often accept to simplify their problem instead of resorting more sophisticated but unfamiliar methods.

1.1. Pre-processing practices at MEOW

Pre-processing operations, except dimension reduction, appeared in the form of implicit methods empirically acquired over time. However, most modellers were concerned about the lack of automated tools for pre-processing. Since the Mexico group has worked a lot on the dissemination and application of sensitivity analysis to complex models [25], it is not surprising that nearly all the group members used these techniques for reducing problems dimensions (Table 1-Annex1), i.e., to select a limited number of parameters from a larger group before the optimisation process (Fig 1-Annex1). Despite the use of sensitivity analysis, meta-models were never used to explore the shape of the objective function before the optimisation. Interestingly, all model calibrations were based on single criterion objective functions, sometimes combining different datasets, while decision and design optimisations were based on multiple criteria (Table 1-Annex1). Pre-processing is an important phase of any optimisation procedure but it tends to be sometimes neglected or not reported explicitly because it is considered to be time-consuming. As a consequence, pre-processing is often carried out only in the second iterations

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 December 2019 doi:10.20944/preprints201912.0249.v1

after a first test of the optimisation fails. However, the time devoted to pre-processing is often compensated for by the reduction in the number of iterations of the optimisation workflow before the optimisation problem is properly formulated (see Section 4). Hence, the development of an automated pre-processing toolbox would be very valuable to modellers and would prevent from discouraging the pre-processing stage.

1.2. Optimisation algorithms employed by MEOW members

The practices of the MEOW's group in terms of algorithm choice can be seen through the font choice in Fig 1. Overall, the algorithms chosen conform to the recommendations of Fig 4 in terms of numbers of parameters and optimisation budgets (number of possible solution) as presented in Fig 4.

Despite the large variety of problems encountered (see Table 1_Annex1), most of the optimisation case studies are located in the same part of the graph of Fig 4: a limited number of parameters and a relatively large computational budget. This can be partially attributed to the use of sensitivity analysis as described in Section 3.1. However, discussions revealed that the restriction of the number parameters by model users (for instance, by fixing many parameters to an arbitrary value) was also an important factor.

Local sampling methods and, in particular, metaheuristics (three cases studies used genetic algorithms and one used simulated annealing) are over-represented in comparison to other approaches. The use of the more local searches, such as trust regions, has been mostly neglected, while global approaches (EGO and ABC) were used in four cases (Fig 1-Annex1).

It appears, therefore, that model users chose global searches even if this restricted the number of parameters. Seeking global solutions is naturally more attractive to model users that cannot easily know the mathematical properties of the optimisation criteria they deal with, but one may question whether shifting this trade-off towards more local searches with a larger number parameters would be beneficial.

Several reasons explain the limited number of algorithms tried within the group. Firstly, there may be a strong cultural influence, the popularity of algorithms depending on the application field. For example, metaheuristics are popular in studies of water resources [80]. It seems that many model users, that are not experts in numerical optimisation, are more familiar with metaheuristics and global optimisation methods than with local methods. This may seem surprising when considering algorithm availability as most default optimisers in scientific computing environments (scipy.minimize in Python, optim in R, fminunc in Matlab, Octave, Scilab, ...) are BFGS based. Secondly, implementation issues dissuade model users from switching algorithms and create conservatism: software is platform-dependent, sometimes prohibitively expensive [9], implemented in specific programming languages, as standalone programs or as a library routines with non-standard call methods. Furthermore, reformulating the problem would require considerable work, particularly if the optimisation platform is not sufficiently flexible [9]. Finally, as discussed in the next Section, there is a lack of tools that can be used to assess whether switching algorithms is worth the additional effort.

1. Post-processing customs at MEOW

In general, the case studies did not follow specific guidelines for post-processing. Even when several optimisations were performed, only the one which produced the “best” solution was presented and the iterative “trial-and-error” process between the first formulation and final post-processing analyses were not reported. Although several case studies used the local model approach, only one (case 9 in Table 1-Annex1 and Fig 1-Annex1) used the Hessian matrix to investigate identifiability. No other explicit post-processing tools were applied to assess the robustness of the obtained solutions or to check the relevance of the problem formulation.

In summary, notwithstanding the useful information that is provided by optimisation software, we can deplore that this information is rarely used. Two reasons are apparent: firstly, there is an absence of guidelines, procedures and toolboxes for automating the analysis of this information and, secondly, there is an unquestioning acceptance of the solutions provided, as if the algorithms were “magic potions”. This clearly appears to be a gap in the panel of tools available to practitioners, and could be a significant research topic. In particular, meta-models and fitness landscape concepts may still offer great potential for progress. We strongly encourage model users to make better use of algorithm traces and call for the development of guidelines and toolboxes targeted at model users for automating post-processing.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 December 2019 doi:10.20944/preprints201912.0249.v1

8 Annex 2 : example of ODDO filling with the ISIS-Fish model

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 December 2019 doi:10.20944/preprints201912.0249.v1

MODEL	**Performance**	Time per model run	15min
		parallelisation	Yes (10 runs in parallel)
	Development	language	java
		Implementation of the optimisation algorithm	Recoding of the algorithm in Java

Pre-processing	**Problem Formulation**	**Model**	ISIS-Fish model of the mixed demersal fishery in the Eastern Channel
		Question	Estimating accessibility parameters for cod
		Data	Time series of commercial landings in kg per trip reported in logbooks 2008-2012, tansformed into landings in number at age according to sampling of commercial landings and age-length key, then aggregated per age and year.
		Parameters **Bounds&constrainsts**	6 bounded ([1e-7;1e-4]) continous parameters (accessibility at age, constant over time) Positive values, values should be consistent across ages

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 December 2019 doi:10.20944/preprints201912.0249.v1

		Uncertainty (process and data)	Data uncertainty: declarative data, possible bias, errors in the age-length key and in distribution across ages due to the sampling scheme
		Initial objective function	Sum of squarred relative errors values at age et per year $\mathrm{Sum}_{year,age}[(\mathrm{Data}_{year,age} – \mathrm{Model}_{year,age})/\mathrm{Data}_{year,age}{}^{2}]$
	Objective Function	**building**	FO include data disaggregated at age to match parameter scale. Although kept constant in the model, accessibilities are likely to vary accross years. FO is disaggregated per years so estimated accessibility is an average of yearly accessibilities. Each model-data distance is standardised by data value to give equal weight to each age and age.
		reshaping	Yes. Simulated landings tend to be underestimated, standardisation by the value of the observation was removed in the final FO to give more weight to higher landings. Distance between model and data was divided by 1000 to prevent numerical problems.
		final	$\mathrm{Sum}_{year,age}[((\mathrm{Data}_{year,age} – \mathrm{Model}_{year,age})/1000)^{2}]$
		data	Check of data consistency after transformation and disaggregation

	Exploratory Analysis	**Reduction dimension**	No

Algorithm	**Family**	Evolutionary strategy with starting points forced with a LHS.
	Description-Justification	Evolutionary strategy is used to have a good exploration of the search space. It is easy to understand and runs can later be used to explore the robustness of the result.
	Changes in the algorithm	no
	Settings	• Evolutionnary strategy ○ Population size: 20, 4 best kept at each generation ○ Mutation rate and threshold: 0.8 and 0.5 ○ Crossover method: 2X ○ Maximum number of generations: 500 ○ stopping criteria: none

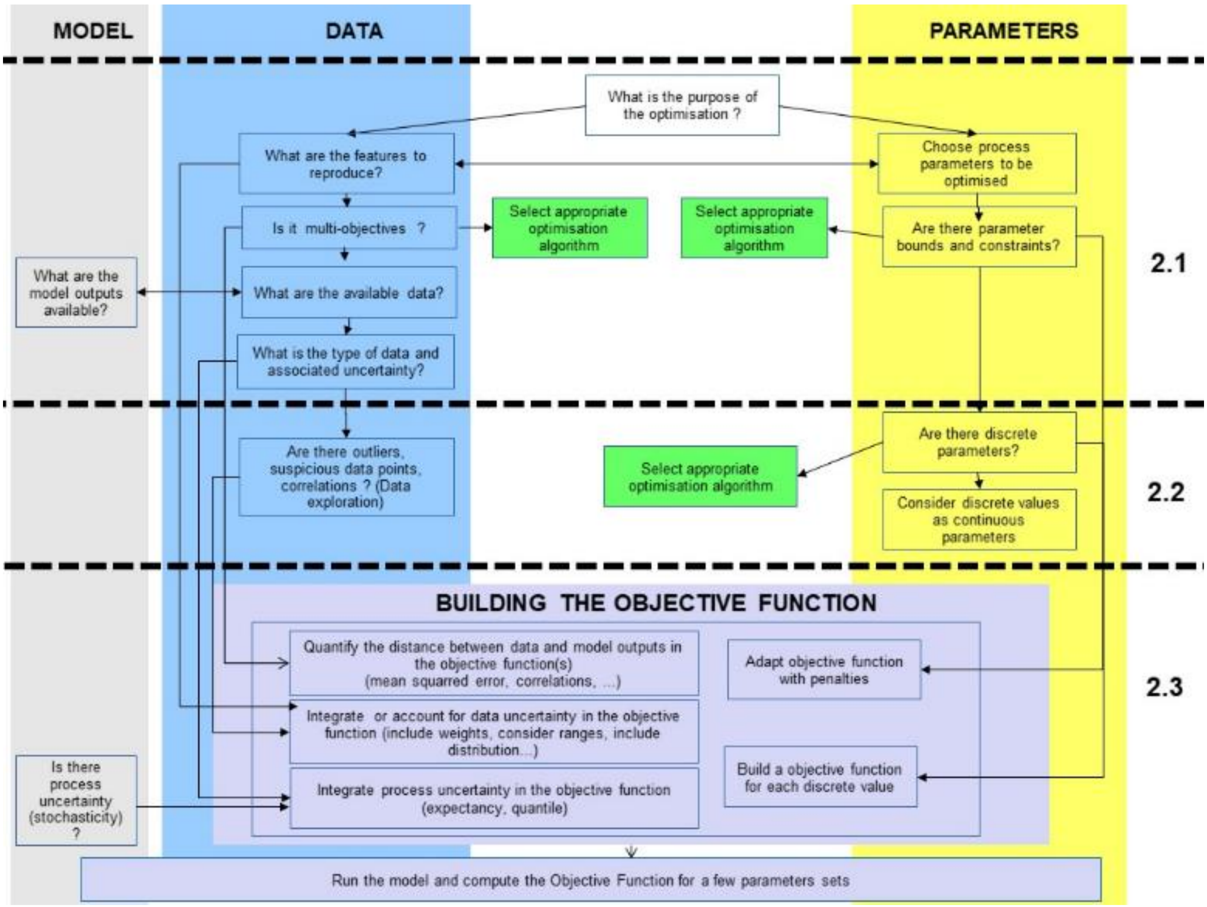
Pos-processing	**Convergence**	Visual examination of the FO and parameter values in course of generations. Examination of the range of variation of parameters in the last generation and in the 100 best solutions. Comparison with previous optimisation exercise (using a simplex) and comparison of values between ages. => First run: Simulated landings tend to be underestimated => Second run: failure dur to numerical problems => Third run: better distribution of residuals. Validation exercise using landings data aggregated at different scales and abundance data.
	Optimum properties including Identifiability	Fo and parameter values stable for the last 100 generations
	Residual analysis	Visual analysis of residuals to detect patterns
	Multicriteria	No

Optimisation process	**Number of simulations required**	10000
	Duration	36h
	Reached stopping criteria	Yes

Comments & failures	A second run was launched to try and solve the underestimation problem. FO was revised (see final): the standardisation by the value of the data was removed in the FO to give more weight to higher landings. Starting values were the results of a LHS with refined bounds according to a previous optimisation exercise. This run failed because of numerical approximation problems (several parameter sets with exact same FO). A third run was launched with distance between model and data was divided by 1000 to prevent numerical problems.

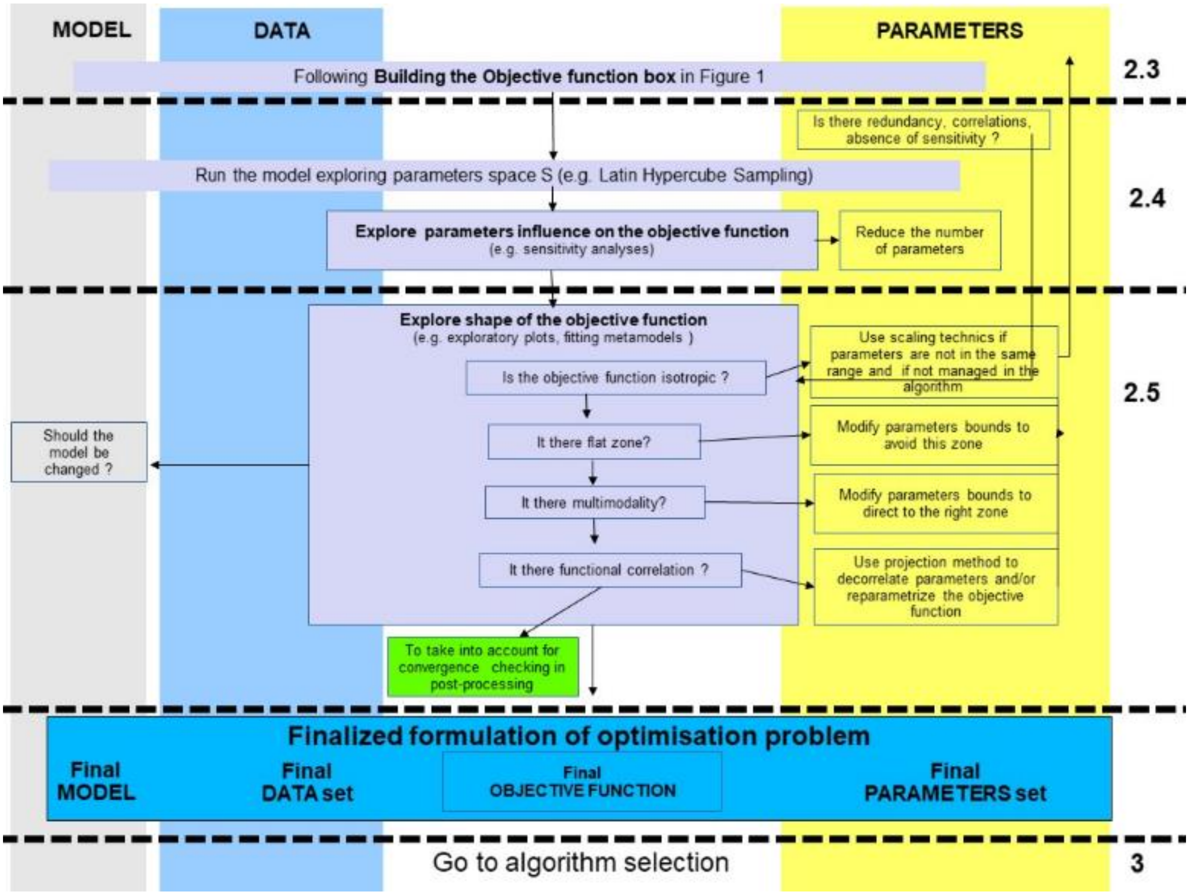
Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 December 2019 doi:10.20944/preprints201912.0249.v1

Fig 1.


MODEL
DATA
PARAMETERS
What is the purpose of the optimisation ?
What are the features to reproduce?
Choose process parameters to be optimised
Is it multi-objectives ?
Select appropriate optimisation algorithm
Select appropriate optimisation algorithm
Are there parameter bounds and constraints?
2.1
What are the model outputs available?
What are the available data?
What is the type of data and associated uncertainty?
Are there outliers, suspicious data points, correlations ? (Data exploration)
Select appropriate optimisation algorithm
Are there discrete parameters?
Consider discrete values as continuous parameters
2.2
BUILDING THE OBJECTIVE FUNCTION
Quantify the distance between data and model outputs in the objective function(s) (mean squarred error, correlations, ...)
Adapt objective function with penalties
2.3
Integrate or account for data uncertainty in the objective function (include weights, consider ranges, include distribution...)
Build a objective function for each discrete value
Is there process uncertainty (stochasticity) ?
Integrate process uncertainty in the objective function (expectancy, quantile)
Run the model and compute the Objective Function for a few parameters sets

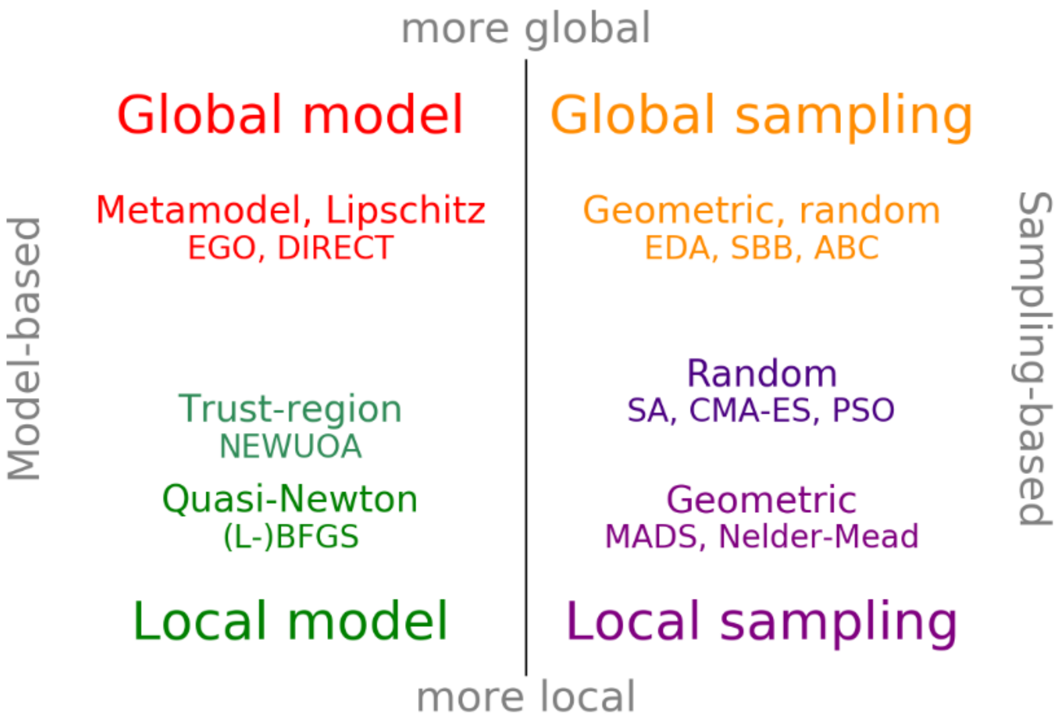
Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 December 2019 doi:10.20944/preprints201912.0249.v1

Fig 2.


MODEL
DATA
PARAMETERS
Following Building the Objective function box in Figure 1
2.3
Is there redundancy, correlations, absence of sensitivity ?
Run the model exploring parameters space S (e.g. Latin Hypercube Sampling)
2.4
Explore parameters influence on the objective function (e.g. sensitivity analyses)
Reduce the number of parameters
Explore shape of the objective function (e.g. exploratory plots, fitting metamodels)
Is the objective function isotropic ?
Use scaling technics if parameters are not in the same range and if not managed in the algorithm
2.5
Should the model be changed ?
It there flat zone?
Modify parameters bounds to avoid this zone
It there multimodality?
Modify parameters bounds to direct to the right zone
It there functional correlation ?
Use projection method to decorrelate parameters and/or reparametrize the objective function
To take into account for convergence checking in post-processing
Finalized formulation of optimisation problem
Final MODEL
Final DATA set
Final OBJECTIVE FUNCTION
Final PARAMETERS set
Go to algorithm selection
3

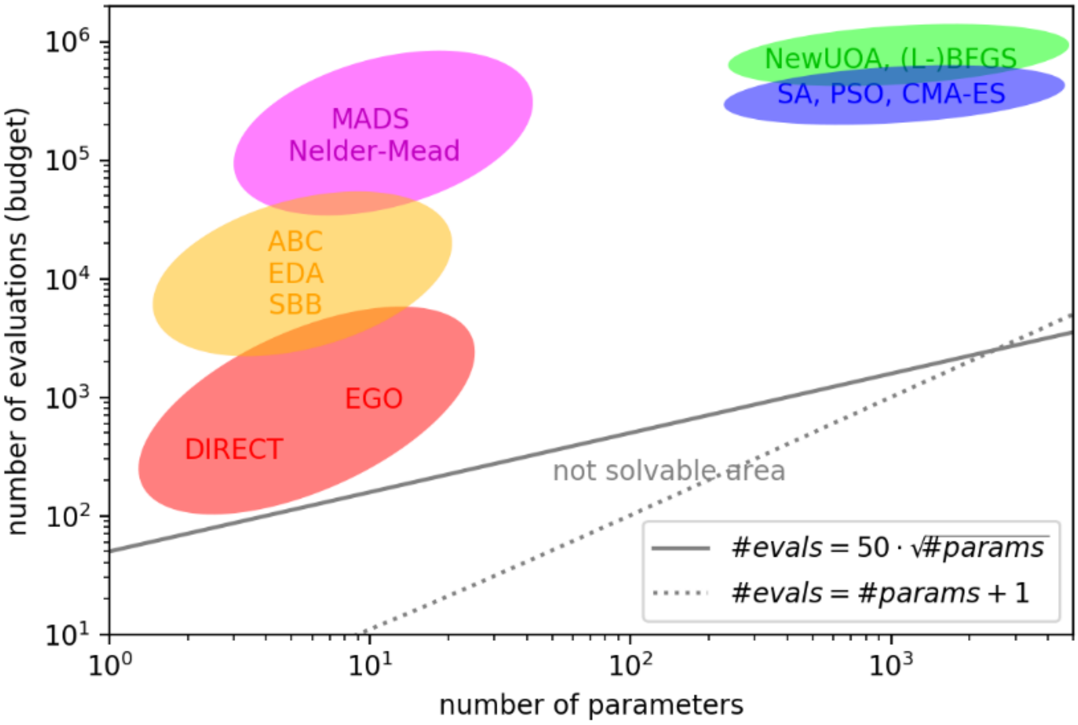
Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 December 2019 doi:10.20944/preprints201912.0249.v1

Fig 3.



1510 Fig 4.

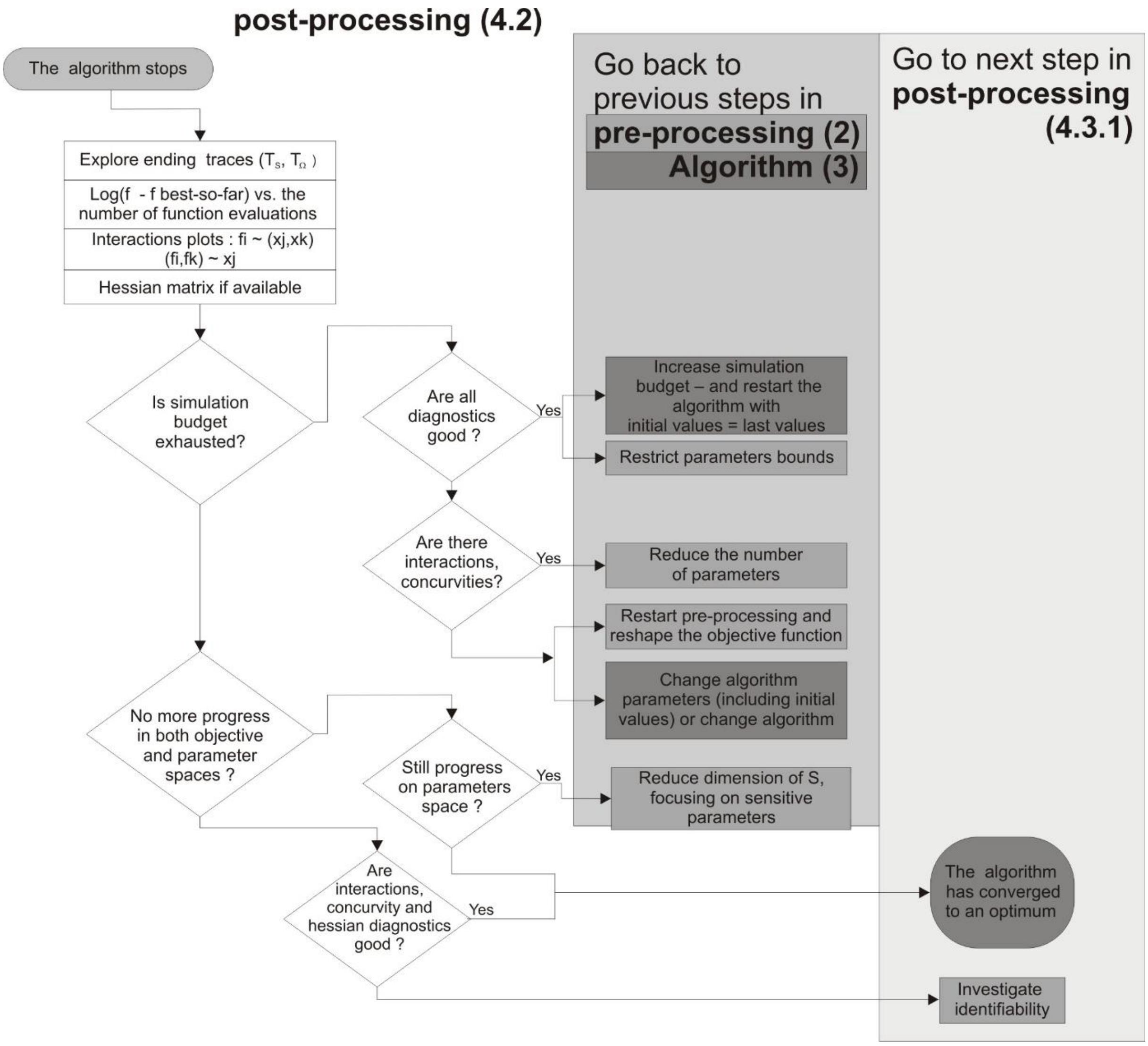
1511



1512

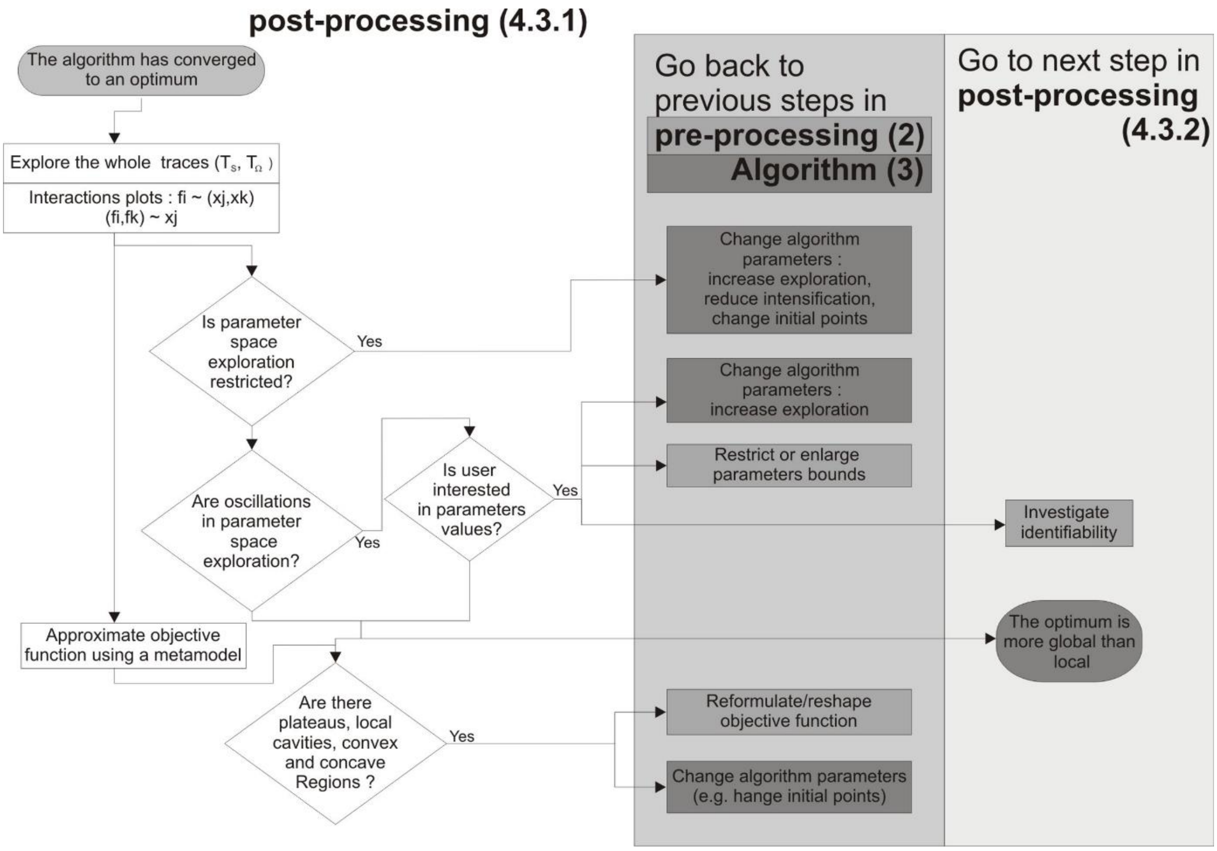
Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 December 2019 doi:10.20944/preprints201912.0249.v1

Fig 5.


post-processing (4.2)
The algorithm stops
Explore ending traces (T_s, T_Ω)
Log(f - f best-so-far) vs. the number of function evaluations
Interactions plots : fi ~ (xj,xk) (fi,fk) ~ xj
Hessian matrix if available
Is simulation budget exhausted?
Are all diagnostics good ?
Yes
Are there interactions, concurvities?
Yes
No more progress in both objective and parameter spaces ?
Still progress on parameters space ?
Yes
Are interactions, concurvity and hessian diagnostics good ?
Yes
Go back to previous steps in pre-processing (2) Algorithm (3)
Increase simulation budget – and restart the algorithm with initial values = last values
Restrict parameters bounds
Reduce the number of parameters
Restart pre-processing and reshape the objective function
Change algorithm parameters (including initial values) or change algorithm
Reduce dimension of S, focusing on sensitive parameters
Go to next step in post-processing (4.3.1)
The algorithm has converged to an optimum
Investigate identifiability

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 December 2019 doi:10.20944/preprints201912.0249.v1

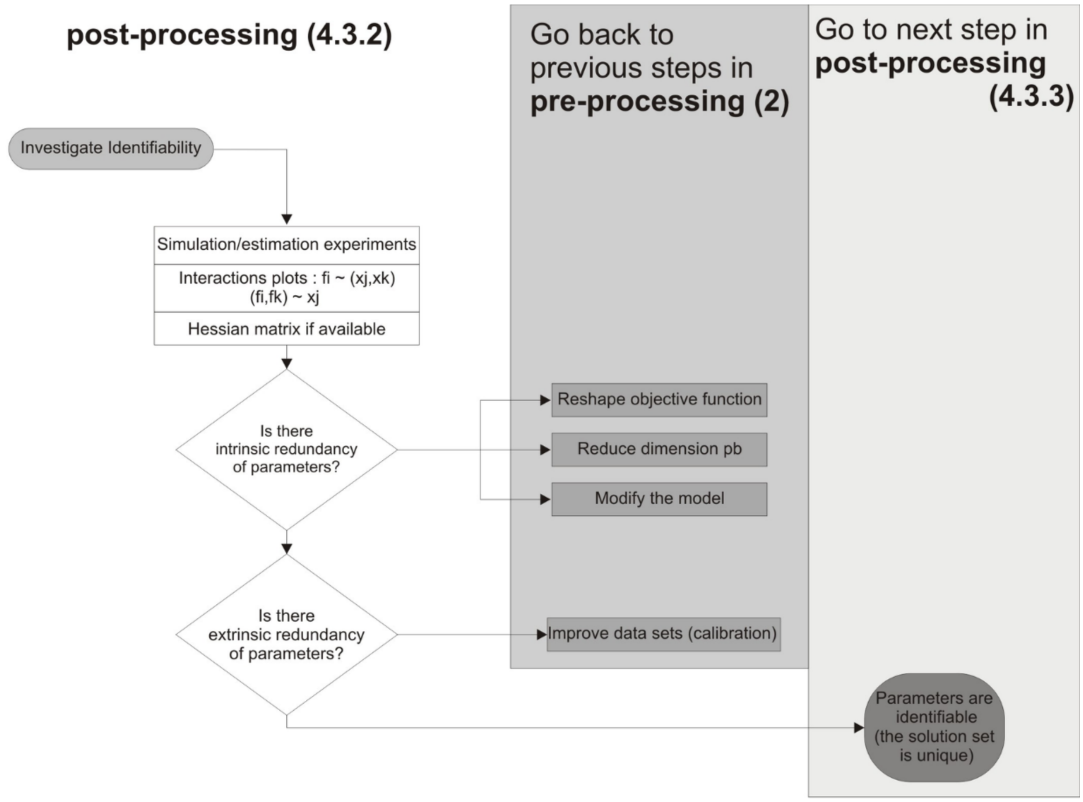
Fig 6.

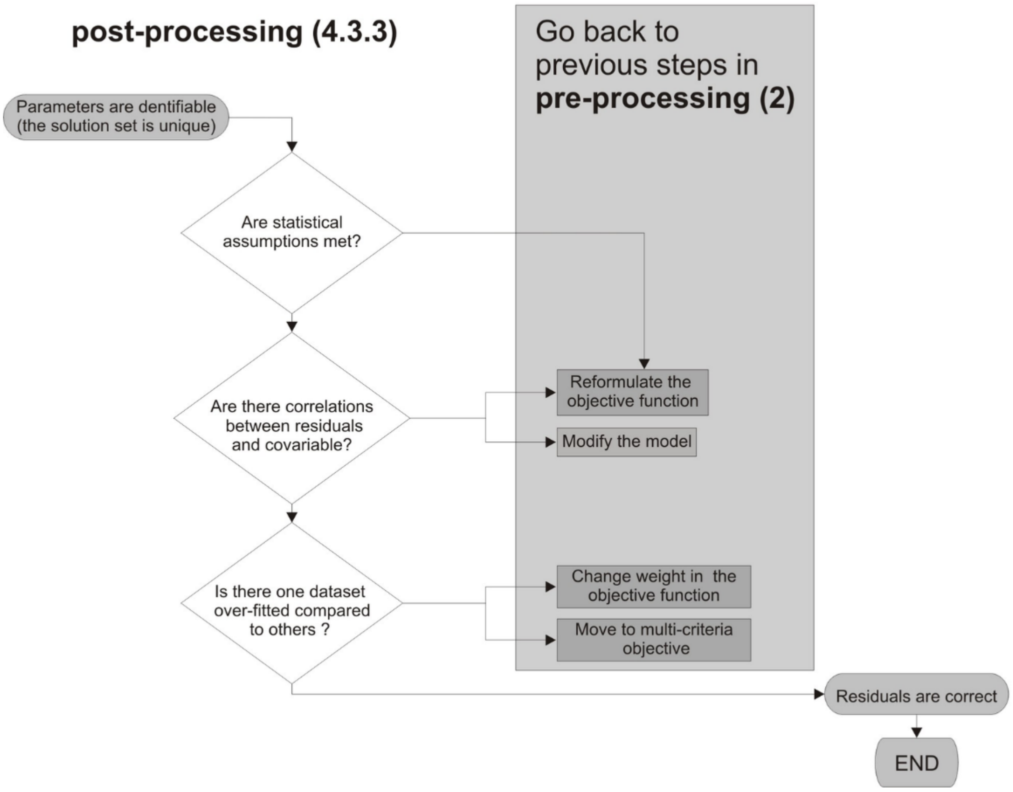

post-processing (4.3.1)
The algorithm has converged to an optimum
Explore the whole traces (T_s, T_Ω)
Interactions plots : fi ~ (xj,xk) (fi,fk) ~ xj
Is parameter space exploration restricted?
Yes
Are oscillations in parameter space exploration?
Yes
Is user interested in parameters values?
Yes
Approximate objective function using a metamodel
Are there plateaus, local cavities, convex and concave Regions ?
Yes
Go back to previous steps in
pre-processing (2)
Algorithm (3)
Change algorithm parameters : increase exploration, reduce intensification, change initial points
Change algorithm parameters : increase exploration
Restrict or enlarge parameters bounds
Reformulate/reshape objective function
Change algorithm parameters (e.g. hange initial points)
Go to next step in
post-processing (4.3.2)
Investigate identifiability
The optimum is more global than local

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 December 2019 doi:10.20944/preprints201912.0249.v1

1521 Fig 7.

1522

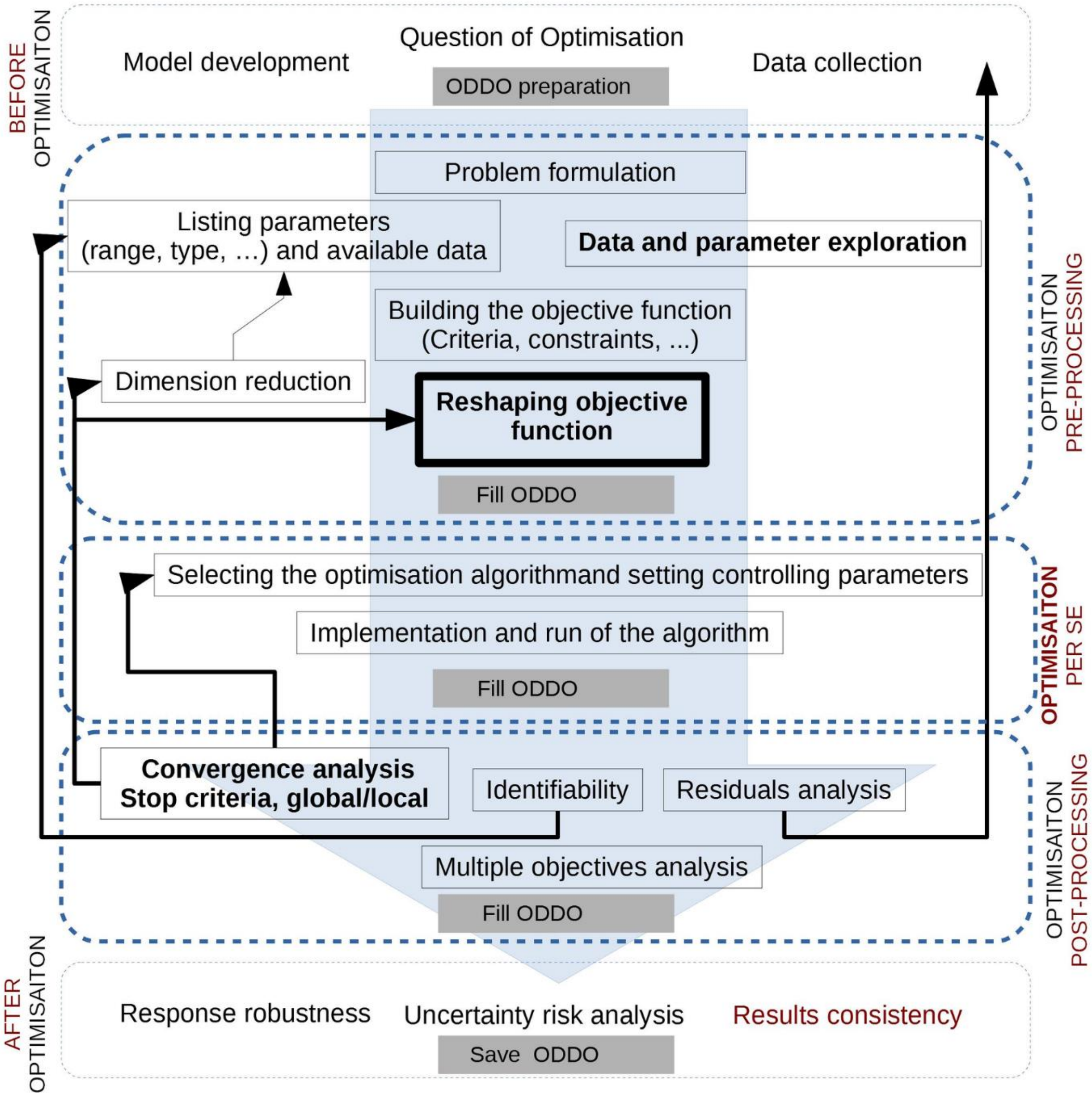

post-processing (4.3.2)
Go back to previous steps in pre-processing (2)
Go to next step in post-processing (4.3.3)
Investigate Identifiability
Simulation/estimation experiments
Interactions plots : fi ~ (xj,xk)
(fi,fk) ~ xj
Hessian matrix if available
Is there intrinsic redundancy of parameters?
Reshape objective function
Reduce dimension pb
Modify the model
Is there extrinsic redundancy of parameters?
Improve data sets (calibration)
Parameters are identifiable (the solution set is unique)


post-processing (4.3.3)
Go back to previous steps in pre-processing (2)
Parameters are dentifiable (the solution set is unique)
Are statistical assumptions met?
Reformulate the objective function
Are there correlations between residuals and covariable?
Modify the model
Is there one dataset over-fitted compared to others ?
Change weight in the objective function
Move to multi-criteria objective
Residuals are correct
END

1523

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 December 2019 doi:10.20944/preprints201912.0249.v1

1524 Fig 8.

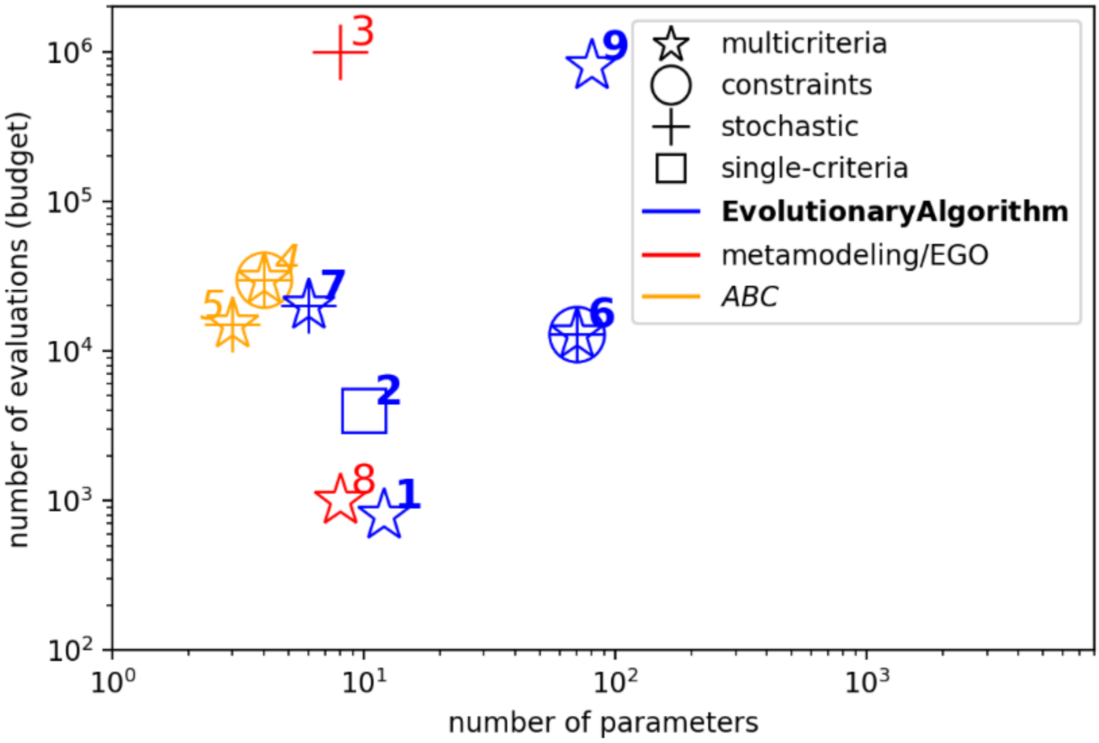


1525

1526

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 19 December 2019 doi:10.20944/preprints201912.0249.v1

1527 Fig 9.



1528