Environmental Modelling and Software 143 (2021) 105125

Contents lists available at ScienceDirect

Environmental Modelling and Software

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

Set-membership parameter estimation based on Voronoi vertices

D. Reyes Lastiri [a], H.J. Cappon [b], K.J. Keesman [c,*]

[a] Farm Technology, Wageningen, UR, the Netherlands
[b] Environmental Technology, Wageningen, UR, the Netherlands
[c] Mathematical and Statistical Methods - Biometris, Wageningen, UR, the Netherlands

ARTICLE INFO

Keywords:
Set-membership
Voronoi diagram
Hyperspheres
Uncertainty propagation

ABSTRACT

Identification of complex systems often face structural modelling errors and limited or low quality data, which hinder statistical characterisations. An alternative is the set-membership approach, where errors are assumed unknown-but-bounded. Set-membership estimation aims to find a feasible parameter set (*FPS*), which produces model outputs that fit within given error bounds. Most algorithms are limited to linear models, small number of parameters, or to discrete approximation of the FPS. These limitations hinder parameter estimation for relatively complex systems. We present an efficient sampling-based set-membership algorithm with low computational complexity that improves the coverage of a discrete approximation of the *FPS*, characterised by hyperspheres from a Voronoi diagram of the parameter space. Additionally, we suggest a measure for set-membership accuracy based on deviations between the given error bounds and the feasible model output set. Our algorithm provides a balance between accuracy and computational complexity, and a tool to investigate practical identifiability.

1. Introduction

Mathematical models for biological and environmental systems often combine *a priori* knowledge and experimental data, requiring a calibration step to estimate the unknown parameters. Frequently, uncertainty of parameter estimates is based on a statistical characterisation of the error in experimental data.

However, biological and environmental systems often present the problems of structural modelling error, and limited availability or low quality of experimental data sets. Therefore, most often a statistical characterisation of the error is inappropriate (Norton, 1987; Keesman, 1990; Walter and Piet-Lahanier, 1990).

In this context of structural modelling error with limited and uncertain data, an alternative to the statistical characterisation is a so-called set-membership (SM) characterisation, where the errors are assumed to be unknown-but-bounded. In the SM approach, ranges of expected system outputs are predicted for each time instant, instead of single values. These ranges account for the effect of uncertainty in data and model outputs (Norton, 1987; Keesman, 1990; Walter and Piet-Lahanier, 1990; Milanese et al., 1996; Diniz, 2013).

Given a p-dimensional parameter vector ϑ, and the error between measurements y and model outputs $f(\vartheta)$ defined as $e(\vartheta) := y - f(\vartheta)$, in a set-membership approach the error is assumed to be point-wise bounded by a fixed positive number, i.e. $\|\|e(\vartheta)\|_\infty \leq \epsilon$. Within the SM approach, set-valued estimation algorithms aim to find an exact or approximate feasible parameter set (*FPS*). The *FPS* produces model outputs that fit inside the error bounds at all times: $FPS := \{\vartheta \in \mathbb{R}^p \mid \|\|y - f(\vartheta)\|_\infty \leq \epsilon\}$ (Fig. 1). For overviews, see Milanese et al. (1996); Keesman (2003); Norton (2003); Walter (2003); Keesman (2011); Cerone et al. (2014).

The simplest case of SM algorithms are based on models that are linear in the parameters. This type of algorithms is used in signal processing for e.g. adaptive filtering, providing reduced computational complexity by updating parameter estimates only when error bounds are exceeded (Diniz, 2013). Applying these methods to non-linear models, common in biological and environmental systems, would introduce additional linearisation steps.

In contrast, SM parameter estimation of non-linear dynamic systems is frequently applied to solve the problem of uncertain and limited data. Common algorithms rely on either interval-based or sampling-based methods.

For example, using an interval-based method, Marvel and Williams (2012) proposed a framework for SM estimation and experimental design to deal with small data sets in biological systems, based on set inversion via interval analysis (SIVIA) (Jaulin and Walter, 1993). Similarly, Rumschinski et al. (2010) proposed a partition algorithm for SM estimation and model invalidation to discriminate between

* Corresponding author.
E-mail address: karel.keesman@wur.nl (K.J. Keesman).

https://doi.org/10.1016/j.envsoft.2021.105125
Accepted 2 July 2021
Available online 8 July 2021

competing models in biochemical reaction networks. Interval-based algorithms can characterise the *FPS* for linear and non-linear models, and are capable of characterising non-convex parameter sets (Rumschinski et al., 2010), common in biological and environmental systems. The *FPS* characterised by interval-based methods is a collection of boxes with sizes determined by a desired accuracy. In general, they consist of a large number of boxes (hypercubes) that define the *FPS* boundary (ca. 20,000 boxes for a Lotka-Volterra test model with two estimated parameters by SIVIA (Marvel and Williams, 2012)). The number of boxes increases with the number of parameters and when higher accuracy is required (Jaulin and Walter, 1993). Furthermore, the number of model evaluations is proportional to the number of boxes characterising the *FPS*, resulting in an exponential computational complexity with respect to the number of parameters (Raïssi et al., 2009), i.e. a computational time with an order $O(n(boxes)^p)$.

The *FPS* characterised by interval-based methods provides an accurate mathematical description that can be used to simulate a model and its prediction uncertainty. However, defining a large number of boxes and simulating from them is computationally demanding.

In other examples, using sampling-based methods, Keesman (1990) proposed the characterisation of the *FPS* by ellipsoids for more effective sampling. Van Straten and Keesman (1991) applied the so-called Monte Carlo set-membership algorithm (MCSM) to calibrate a model for lake eutrophication, and to simulate its prediction uncertainty from the resulting discrete FPS. Similarly, Nurulhuda et al. (2017) applied a Latin hypercube SM algorithm for a flooded rice system, returning a discrete approximation of the *FPS* in a six-dimensional space. Given a fixed sample size per iteration, these algorithms have a linear computational complexity irrespective of the number of parameters, i.e. a computational time with an order $O(n(sample))$.

The *FPS* characterised by sampling-based methods is less computationally demanding than interval-based methods. However, random sampling may require a large number of iterations. Therefore, low computational times and a full coverage of the *FPS* cannot be guaranteed.

Additionally, models of complex biological and environmental systems are often split into sub-models, see for example Van Straten and Keesman (1991), Mohtar et al. (1997), Mocenni and Vicino (2006), and Reyes Lastiri et al. (2018). Uncertainty in the parameter estimates often propagates through sub-models.

In uncertainty propagation studies, the statistical approach uses methods such as first-order variance propagation or Monte Carlo simulation. First-order variance propagation is hindered by the non-linearity of complex systems. Similarly, Monte Carlo simulation, which requires sampling from a probability density function (*pdf*), is hindered by the uncertain and limited nature of experimental data, which may lead to inadequate *pdf* characterisation.

Alternatively, in the SM approach, interval-based methods preserve the uncertainty characterisation from the calibration step, but they are hindered by computational complexity. In contrast, sampling-based methods have less computational complexity, but simulating a discrete *FPS* without full coverage, and extending discrete feasible model output trajectories in time beyond the calibration interval, may lead to falsified predictions with underestimated uncertainty.

Consequently, given the aforementioned limitations, it is desired to have an efficient sampling-based algorithm that closely approximates the *FPS*, while preventing underestimation and overestimation of uncertainty propagation.

Therefore, the objective of the paper was to investigate and evaluate a novel set-membership estimation algorithm that meets the requirements of an uncertainty propagation study of complex systems with limited or low quality data.

2. Methods

The sampling-based SM estimation algorithm proposed in this paper aims at efficiently characterising an approximate *FPS*, using a relatively low number of elements. Additionally, deviations from error bounds are quantified. The algorithm was evaluated in four test cases.

The algorithm was developed in Python 3.5, using the SciPy functions for Delaunay triangulation and Voronoi diagrams based on the Qhull library (Barber et al., 1996). The algorithm tests were run in the IPython console of WinPython 3.5 in an Intel Core i7-4790 CPU computer at 3.60 GHz with 16.0 GB RAM operating on Windows 7. The models were simulated with the 'odeint' function in the SciPy package (v. 0.18.1) of Python using its default parameters.

2.1. Algorithm

We propose characterizing the approximate *FPS* based on (hyper)

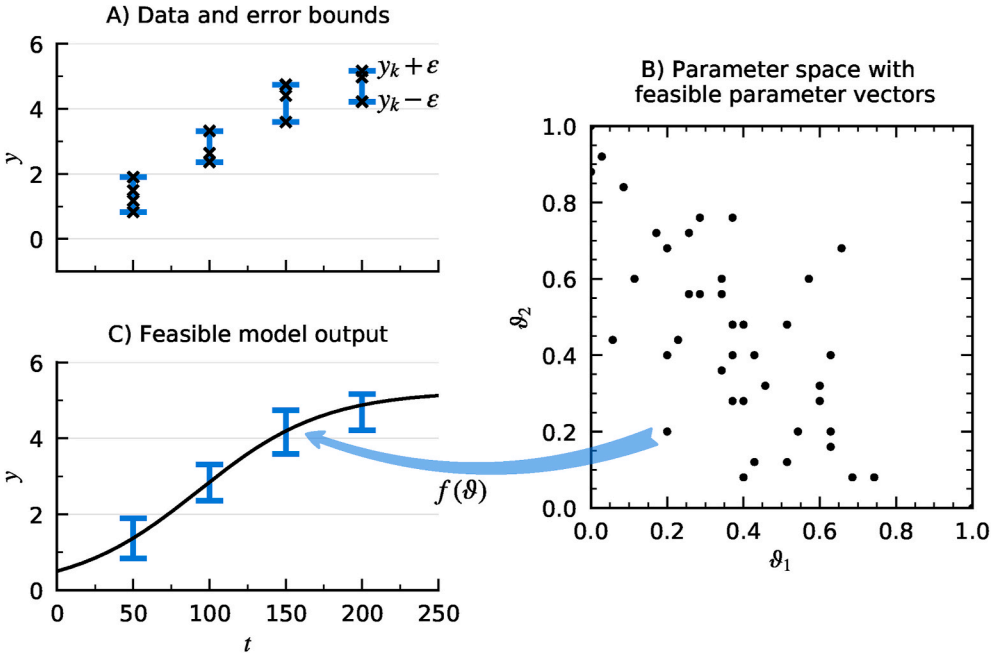

Fig. 1. Set-membership (SM) parameter estimation. **A)** Uncertainty in measured experimental data (y) for each measurement instant (k) is defined by the error bounds of interval $[y_k - \epsilon, y_k + \epsilon]$. **B)** Some points in the space ($\mathbb{R}^p$ with $p = 2$) represent a parameter vector ϑ that produce model outputs that fit within the error bounds for all measurement instants. **C)** Example of a feasible model output $f(\vartheta)$ fitting within the bounds: $\|\|y - f(\vartheta)\|_\infty \leq \epsilon$. The objective in SM estimation is to characterise the set containing all feasible ϑ in $\mathbb{R}^p$, known as the feasible parameter set (*FPS*), with its corresponding feasible model output set (*FMOS*).

spheres centred at Voronoi vertices. Parameters are assumed time-independent. The Voronoi diagram is generated using all known unfeasible parameter vectors as seeds, thus excluding them by definition.

Using spheres, complex *FPS* shapes can be approximated by two data vectors: centre coordinates (c) and corresponding radii (r). Therefore, non-convex parameter sets from non-linear models can be characterised with a relatively low number of elements.

Random sampling from spheres of the characterised *FPS* is computationally simple, and can then be used to simulate the model for $t \in [0, T]$, where T is the final simulation time, including the calibration time interval. Such sampling-based method from a low number of elements allows for relatively low computational times.

Convergence of the algorithm is not defined in the parameter space $\mathbb{R}^p$, but in the model output space in terms of deviations between the approximate feasible model output set (*FMOS*) and the data error bounds for all measurement instants. In this way, although sampling-based, the algorithm tries to prevent under- and overestimation of prediction uncertainty.

2.1.1. Voronoi diagram and Delaunay triangulation

In a Voronoi diagram (Fig. 2), a space is partitioned into regions based on the distance to a specified set of points. For each of these points, called seeds, there is a corresponding Voronoi region (- -) where any point is closer to its seed than to any other seed. The Voronoi diagram for a set of points is dual to its Delaunay triangulation (−). In a Delaunay triangulation no point is inside the circumcircle of any triangle. The triangle circumcentres are also the Voronoi vertices. Our algorithm is based on a Voronoi diagram from the collection of all known unfeasible parameter vectors, thus excluding them from the approximate *FPS*.

2.1.2. Definitions

For the algorithm implementation, we define ϑ_F and ϑ_U as a feasible and unfeasible parameter vector, respectively. Ω_{ϑ_F} and Ω_{ϑ_U} are then the collections of all known feasible and unfeasible parameter vectors.

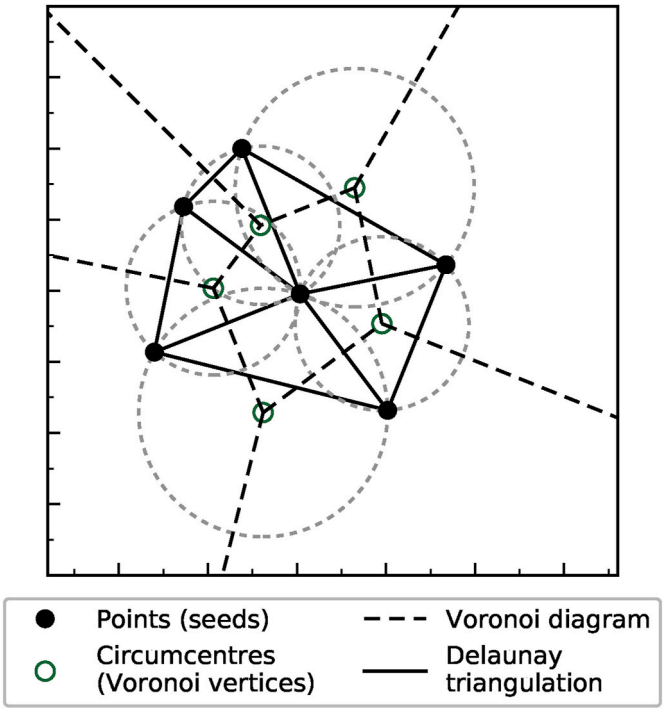

Fig. 2. Voronoi diagram and Delaunay triangulation. The calculated Voronoi vertices are also circumcentres of circles that do not contain any of the given set of points (seeds).

We also define S_ϑ as a random sample of parameter vectors from the set of spheres in $\mathbb{R}^p$. These sampled parameter vectors are evaluated individually to split the sample into a feasible S_{ϑ_F} and an unfeasible S_{ϑ_U} set of parameter vectors.

Furthermore, we define two types of deviations to measure accuracy of the approximate *FPS* at each measurement instant k. Overestimation (OE_k) is the maximum deviation between model output $f(\vartheta)$ and error bounds, related to ϑ_U sampled in the *FPS*. Similarly, underestimation (UE_k) is the maximum deviation between model output $f(\vartheta)$ and error bounds, related to all known ϑ_F. Thus, for measurement instants $k = 1, \ldots, M$, we define:

$$\begin{aligned} OE_k := & \max_{i_U} \left[f(\vartheta_{U,i_U})_k - (y_k + \epsilon)\right] \\ & + \max_{i_U} \left[(y_k - \epsilon) - f(\vartheta_{U,i_U})_k\right] \\ UE_k := & \min_{i_F} \left[(y_k + \epsilon) - f(\vartheta_{F,i_F})_k\right] \\ & + \min_{i_F} \left[f(\vartheta_{F,i_F})_k - (y_k - \epsilon)\right] \end{aligned} \tag{1}$$

Notice that OE_k is calculated for a sample of unfeasible parameters with size $n(S_{\vartheta_U})$ obtained from the approximate *FPS* in each iteration, i.e. $i_U = 1, \ldots, n(S_{\vartheta_U})$. In contrast, UE_k is calculated for the set of size $n(\Omega_{\vartheta_F})$ of all known feasible parameters up to the current iteration, i.e. $i_F = 1, \ldots, n(\Omega_{\vartheta_F})$. Fig. 3 illustrates these definitions.

Total OE and UE are defined as the sum of deviations for all measurement instants (M) (Eq. (2)):

$$\begin{aligned} OE &:= \sum_{k=1}^{M} OE_k \\ UE &:= \sum_{k=1}^{M} UE_k \end{aligned} \tag{2}$$

Finally, we define the scalar-valued weighted deviation (WD) from OE and UE. OE is weighted for the number of ϑ_U found in a sample inside the collection of spheres ($S_{\vartheta_U,in}$) at the corresponding iteration. UE is weighted for the number of ϑ_F found in a sample outside but near the collection of spheres ($S_{\vartheta_F,out}$) at the corresponding iteration. Consequently:

$$\begin{aligned} WD := & \left[n(S_{\vartheta_U,in})/n(S_{\vartheta,in})\right] OE \\ & + \left[n(S_{\theta_F,out})/n(S_{\vartheta,out})\right] UE \end{aligned} \tag{3}$$

2.1.3. Implementation

The algorithm consists of a main function 'Estimate' and four sub-functions: 'Initialize', 'Generate spheres', 'Sample', and 'Evaluate'. Implementation in pseudocode is shown in Algorithms 1 and 2. The functions are illustrated in Fig. 4 and described below. Supporting functions are provided in Algorithm A.1 in the Appendix. The approximate *FPS* is characterised by sets of centres and radii of spheres (c, r).

1) **Initialize** (Fig. 4.1): The sample set is initially defined as a (hyper) cube grid ($\Omega_{\vartheta,ini}$). Latin hyper-cube sampling is applied iteratively on the grid $\Omega_{\vartheta,ini}$, evaluating N_{ini} points per iteration. Sampled points are evaluated as feasible or unfeasible, initializing the sets Ω_{ϑ_F} and $\Omega_{\vartheta,U}$. Sampling and evaluation continues until a desired size of Ω_{ϑ_F} is obtained. The sets are subsequently normalized to Ω_{θ_F} and Ω_{θ_U} based on the minimum and maximum feasible coordinates found in the sample. Normalization helps maintaining the approximate *FPS* near the range [0,1] for all parameters. If not enough ϑ_F or ϑ_U are found, the algorithm must be restarted. This step is purely random, and therefore governed by the choice of parameter ranges. To decrease randomness, it is best to have prior knowledge on the model behaviour, and parameters with physical meaning (constrained to a certain region). If no prior knowledge is available, a large $\Omega_{\vartheta,ini}$ with large N_{ini} can be used, which is subsequently narrowed by trial and error using this step alone.

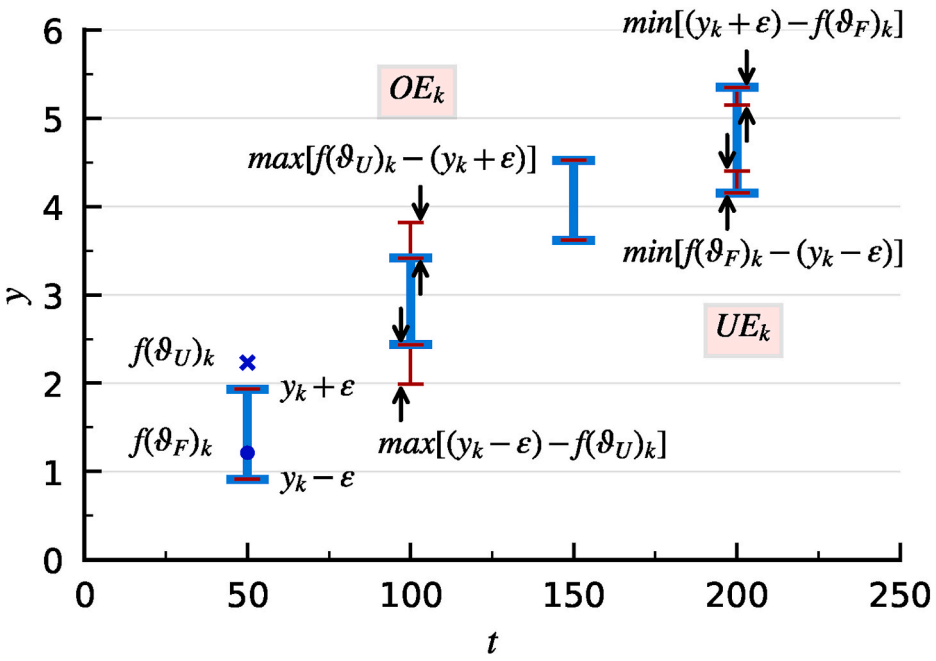

Fig. 3. Overestimation (*OE*) and underestimation (*UE*). At each measurement instant k, bounds are given by $y_k + \epsilon$ and $y_k - \epsilon$. An example of feasible ($f(\vartheta_F)$) and unfeasible ($f(\vartheta_U)$) model outputs is shown at $k = 50$. OE_k shown for $k = 100$ is the sum of deviations outside the data error bounds. UE_k shown for $k = 200$ is the sum of deviations inside the data error bounds. Convergence of the algorithm is based on reducing OE_k and UE_k for all k, to a given tolerance value for the total weighted deviation (WD_{tol}).

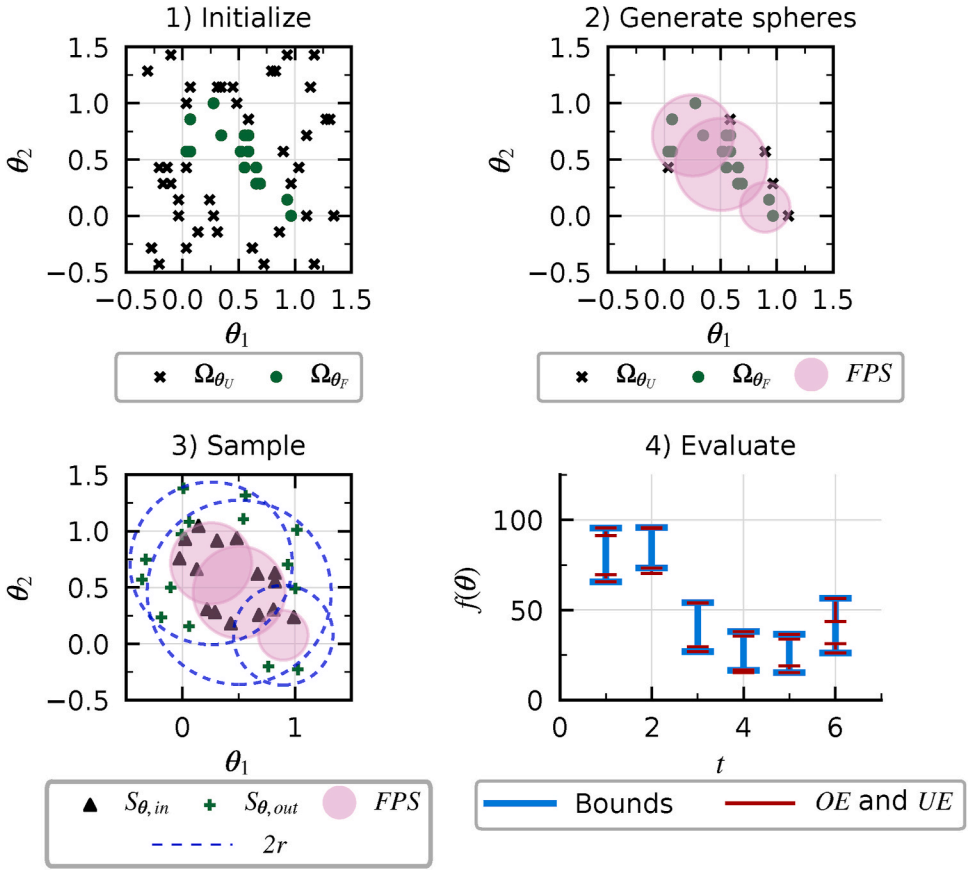

Fig. 4. Algorithm functions. Example on a parameter space $\mathbb{R}^p$ with $p = 2$.**1)** Latin hypercube sampling is applied on $\mathbb{R}^p$. Normalization of $\mathbb{R}^p$ helps enclosing the approximate *FPS* near the interval [0,1] for all coordinates. **2)** A Voronoi diagram is generated with Ω_{θ_U} as seeds. Subsequently, spheres are generated with centres in Voronoi vertices. In the first iteration, only θ_U circumscribed in the approximate *FPS* spheres are kept. **3)** Two samples are obtained from the approximate *FPS*. The first sample inside the sphere set ($S_{\theta,in}$) searches for new θ_U. The second sample outside the sphere set ($S_{\theta,out}$) searches for new θ_F at a distance up to $2r$ from each sphere. **4)** Both samples are evaluated and split into feasible ($S_{\theta,F}$) and unfeasible ($S_{\theta,U}$), and appended to Ω_{θ_F} and Ω_{θ_U}, respectively. Subsequently, *OE* and *UE* are calculated to determine *WD* (Eqn. (3)).

2) **Generate spheres** (Fig. 4.2): A Voronoi diagram is generated using the SciPy function 'scipy.spatial.Voronoi' (The SciPy community, b), with Ω_{θ_U} as seeds. The Voronoi vertices generated constitute sphere centres. By definition, those spheres exclude any known θ_U. Given minimum and maximum radii ($r_{\min}$, $r_{\max}$), and constraints based on prior knowledge ($\vartheta_{\min}$, $\vartheta_{\max}$), an optional filter can be applied to remove spheres with $r < r_{\min}$ and $r > r_{\max}$, and spheres with any centre coordinate $c_j < \vartheta_{\min,j}$ and $c_j > \vartheta_{\max,j}$. Subsequently, spheres that do not contain any θ_F are removed. Finally, redundant spheres are also removed, i.e. spheres containing only θ_F included in larger spheres. These filters ensure that the number of spheres is kept low and their size is kept large, reducing memory use. In the first

iteration, the set Ω_{θ_U} is reset to contain only θ_U on which spheres with θ_F are circumscribed. This reduces the amount of data stored and focuses the next steps on the region where θ_F are found.

3) **Sample** (Fig. 4.3): Two random uniform samples of parameter vectors with size $n(S)$ are obtained from the set of spheres. Both samples are achieved by combining a normal distribution and its inverse (Madan, 2017). The first sample ($S_{\theta,in}$) is obtained inside the spheres to search for new θ_U. The second sample ($S_{\theta,out}$) is obtained in a ring outside but near the spheres, with radii ranging from r to $2r$, to search for new θ_F. Redundant θ sampled from overlapping spheres are discarded, helping both samples to concentrate near the approximate *FPS* boundary to improve its accuracy.
4) **Evaluate** (Fig. 4.4): Feasibility of sampled parameter vectors is evaluated for both samples $S_{\theta,in}$ and $S_{\theta,out}$. Weighted deviation (WD) is calculated from Eqn. (3). The algorithm stops if $WD \leq WD_{tol}$. Otherwise, unfeasible parameter vectors found inside the

Algorithm 1 VorSetMembership

Require: $\Omega_{\vartheta,ini}, n(\Omega_{\theta_F,ini}), WD_{tol}, y_{min}, y_{max}$

function ESTIMATE FPS($\Omega_{\vartheta,ini}$)
  $\Omega_{\theta_F}, \Omega_{\theta_U}, \vartheta_{min}, \vartheta_{max} \leftarrow$ INITIALIZE($\Omega_{\vartheta,ini}$)
  while $WD < WD_{tol}$ **do**
    $c, r, \Omega_{\theta_F} \leftarrow$ SPHERE SET($\Omega_{\theta_F}, \Omega_{\theta_U}$)
    $S_{in}, S_{out} \leftarrow$ SAMPLE($c, r, n(S)$)
    $\Omega_{\theta_F}, \Omega_{\theta_U}, WD \leftarrow$ EVALUATE(S_{in}, S_{out})
  end while
  return c, r
end function

function INITIALIZE($\Omega_{\vartheta,ini}, n(\Omega_{\theta_F,ini})$)
  while $n(\Omega_{\theta_F}) < n(\Omega_{\theta_F,ini})$ **do**
    $S_\vartheta \leftarrow LHCS(\Omega_{\vartheta,ini})$ ▷ Latin Hypercube Sampling
    for ϑ in S_ϑ **do**
      if $y_{min,k} \leq f(\vartheta)_k \leq y_{max,k}\ \forall\ k$ **then**
        $\Omega_{\vartheta_F} \leftarrow append(\vartheta)$
      else
        $\Omega_{\vartheta_U} \leftarrow append(\vartheta)$
      end if
    end for
  end while
  $\vartheta_{min}, \vartheta_{max} = \min_j(\Omega_{\vartheta_{F,j}}), \max_j(\Omega_{\vartheta_{F,j}})$ ▷ Min and Max of each coordinate j
  $\Omega_{\theta_F}, \Omega_{\theta_U} \leftarrow$ NORMALIZE($\Omega_{\vartheta_F}, \Omega_{\vartheta_U}, \vartheta_{min}, \vartheta_{max}$) ▷ Normalize to [0,1] (see Appendix A)
  return $\Omega_{\theta_F}, \Omega_{\theta_U}, \vartheta_{min}, \vartheta_{max}$
end function

function GENERATE SPHERES($\Omega_{\theta_F}, \Omega_{\theta_U}$)
  $V_P, V_V \leftarrow$ VORONOI(Ω_{θ_U}) ▷ One point V_P matching each vertex V_V. SciPy function (The SciPy community, b)
  $c, r \leftarrow V_V, \|V_P - V_V\|_2$
  $c, r \leftarrow$ OPT FILTER(c, r) ▷ Remove small, large, and distant spheres (see Appendix A)
  $c\theta_F \leftarrow$ CDIST(c, θ_F)$/r$ ▷ 0,1 matrix of θ_F per sphere, SciPy function (The SciPy community, a)
  $c, r \leftarrow$ FILTER(c, r) ▷ Remove empty and redundant spheres (see Appendix A)
  return c, r
end function

function SAMPLE($c, r, n(S)$)
  for c_i, r_i in c, r **do**
    $n(S_i) \leftarrow n(S)\ (r_i/\text{SUM}(r))$
    $S_{\theta,in}, S_{\theta,out} \leftarrow$ SAMPLE RING($c_i, r_i, n(S_i)$) ▷ Based on normal distribution sample (Madan, 2017)
    $S_\theta \leftarrow S_\theta$ where $\left\|S_\theta - c_{ii\neq i}\right\|_2 > r_{ii\neq i}$ ▷ Removes θ contained also in spheres $\neq i$
  end for
  return $S_{\theta,in}, S_{\theta,out}$
end function

approximate *FPS* ($S_{\theta_{in,U}}$) are appended to the set Ω_{θ_U}, feasible parameter vectors found outside the approximate *FPS* ($S_{\theta_{out,F}}$) are appended to the set Ω_{θ_F}, and a new iteration starts.

Algorithm 1. VorSetMembership

Algorithm 2. VorSetMembership continue

2.2. Test model cases

The algorithm was tested in four cases based on three non-linear dynamic models with biological and environmental components: a Lotka-Volterra population model, a microbial growth and degradation with under Monod kinetics, and the grass growth model in GRASIM (Mohtar et al., 1997).

For each case, artificial noisy measurements were generated around the base model outputs using a zero-mean uniform distribution with $\epsilon = (1/4)\bar{y}$, where $\bar{y}$ is the time-averaged value of the model output. All model cases were initialized in parameter space $[0.5\vartheta^*, 1.5\vartheta^*]$, where ϑ^* is the vector of nominal, reference parameter values used to generate the artificial noisy measurements. The initial number of feasible parameter vectors required was $n(\Omega_{\vartheta_F,ini}) = 15$. Sampling size for each iteration was: $n(S_{\theta,in}) = n(S_{\theta,out}) = 50$. Details for each test case are provided in Table 1. See also Appendix B for a list of symbols for each model.

2.2.1. Lotka-Volterra

The Lotka-Volterra model is a canonical biological model that describes the dynamics of predator-prey populations. It is represented by the differential equations:

$$\begin{aligned} \dot{x_1} &= x_1(p_1 - p_2 x_2) \\ \dot{x_2} &= -x_2(p_3 - p_4 x_1) \end{aligned} \tag{4}$$

where the states are prey (x_1) and predator (x_2) populations, and the four parameters (p) represent rates of population change due to growth, death, and predation. This model has been used previously to test algorithms for set-membership parameter estimation (Marvel and Williams, 2012; Raissi et al., 2004).

We used the Lotka-Volterra model to analyse the algorithm behaviour on two identifiable cases: bounding one state (x_1) and two states (x_1, x_2).

Table 1
Description of the test model cases.

Case	Bounded states	Estimated parameters	WD_{tol}
(1) Lotka-Volterra	x_1	p_2, p_4	0.005
(2) Lotka-Volterra	x_1, x_2	p_2, p_4	0.005
(3) Monod	X	μ_{max}, K_S, Y, k_d	0.5
(4) GRASIM	W_{DM}	a, β, k, φ	0.5

2.2.2. Growth with Monod kinetics

Microbial growth and degradation with Monod kinetics is represented by the differential equations:

$$\begin{aligned} \dot{S} &= -\mu_{max}\frac{X}{Y}\frac{S}{K_S + S} \\ \dot{X} &= \mu_{max}\frac{S}{K_S + S}X - k_d X \end{aligned} \tag{5}$$

where the states are concentrations of substrate S and bacteria X, with interactions described by four parameters related to growth, decay, yield efficiency, and substrate limitation.

The Monod kinetics is theoretically identifiable from noise-free measurements, but its parameters cannot be uniquely identified from noisy measurements (Holmberg, 1982). Therefore, we used this model to analyse the algorithm behaviour on one unidentifiable case.

2.2.3. GRASIM

The grass growth model in GRASIM splits the plant into components for storage and structure (Mohtar et al., 1997):

$$\begin{aligned} \dot{W}_s &= f_P - f_{SR} - f_G - f_{MR} \\ \dot{W}_g &= f_G - f_S \\ [8pt] W_{DM} &= W_g/0.4 \end{aligned} \tag{6}$$

with carbon mass flows for photosynthesis (f_P), shoot respiration (f_{SR}), structure growth (f_G), maintenance respiration (f_{MR}), and senescence (f_S). The model output (W_{DM}) is the dry matter of plant structure mass, assuming 40% C content in dry matter. See Appendix C for the detailed model.

We used the GRASIM model to test the applicability of our algorithm on a larger model, aiming towards implementation in biological and

Algorithm 2 VorSetMembership continued

function EVALUATE($S_{\theta,in}$, $S_{\theta,out}$)
  $S_{\vartheta,in}$, $S_{\vartheta,out} \leftarrow$ NOMINAL($S_{\theta,in}$, $S_{\theta,out}$, ϑ_{min}, ϑ_{max})
  S_{ϑ_F}, $S_{\vartheta_U} \leftarrow \varnothing$
  for ϑ in $S_{\vartheta,in}$ **do**
    if not $y_{min,k} \le f(\vartheta)_k \le y_{max,k}\ \forall\ k$ **then**
      $S_{\vartheta_U} \leftarrow append(\vartheta)$
    end if
  end for
  for ϑ in $S_{\vartheta,out}$ **do**
    if $y_{min,k} \le f(\theta)_k \le y_{max,k}\ \forall\ k$ **then**
      $S_{\vartheta_F} \leftarrow append(\vartheta)$
    end if
  end for
  Ω_{ϑ_F}, $\Omega_{\vartheta_U} \leftarrow append(S_{\vartheta F}, S_{\vartheta_U})$
  $WD \leftarrow$ WD($OE(S_{\vartheta_U})$, $UE(\Omega_{\vartheta_F})$) ▷ Eqs. 1 - 3
  Ω_{θ_F}, $\Omega_{\theta_U} \leftarrow$ NORMALIZE(Ω_{ϑ_F}, Ω_{ϑ_U}, ϑ_{min}, ϑ_{max}) ▷ Normalize to [0,1]. (See Appendix A)
  return Ω_{θ_F}, Ω_{θ_U}, WD
end function

environmental systems.

Commonly, estimating multiple parameters in one model is preceded by a sensitivity analysis, to find the most dominant parameters. If however, more than four dominant parameters are found, the computational complexity of the algorithm will further increase for a given WD_{tol} and *FPS* coverage. Consequently, for higher dimensional parameter cases, accuracy of the algorithm can be relaxed by increasing WD_{tol} and N_{ini}, ultimately leading to the MCSM algorithm by Keesman and Van Straten (1990), demonstrated on a 16-dimensional parameter case by Van Straten and Keesman (1991).

3. Results and discussion

The main results for each case are shown in Table 2. Due to the sample-based nature of the algorithm, results varied between runs, but the behaviour and final accuracy remained consistent.

It is important to note that in spite of the geometrical relation, the proposed characterisation by spheres is different from the characterisation by ellipsoids. The spheres proposed here are solely based on the Voronoi tessellation of θ_U, and do not provide a direct description of the direction or shape of the approximate *FPS*. In contrast, ellipsoids use tools like principal component analysis (PCA) of θ_F to describe the directionality of the approximate *FPS*.

The choice of WD_{tol} is arbitrary. It depends on previous experience with the model and on the desired accuracy.

Notice that we assumed an error of fixed width for all artificial measurement instants, which constitutes a worst-case scenario for underestimation (*UE*).

3.1. Case 1) Lotka-Volterra

1 output and 2 estimated parameters.

For case 1, we assumed that only x_1 was measured and bounded ($y = x_1$). The approximate *FPS* was characterised by a maximum of 10 spheres (Fig. 5.1). The algorithm converged after a maximum of 14 iterations (1400 model simulations). Deviations between *FMOS* and error bounds show a negligible *OE* (Fig. 5.2). However, the convergence history shows a stagnant underestimation (Fig. 5.3), which means that the model cannot reproduce the entire error bounds (which in general is the case). Nonetheless, *WD* converged because there were no new θ_F found outside the sphere set in the last iteration.

Uncertainty propagation from Ω_{θ_F} on x_2 (Fig. 5.4) is similar to the one obtained by the SIVIA algorithm (Marvel and Williams, 2012). However, due to random sampling, any ϑ_U in the approximate *FPS* could introduce a deviation in x_2 beyond the propagation from ϑ_F. This deviation and the uncertainty propagation itself can be avoided by bounding x_2. We show this in the next case.

Table 2
Results [min,max] from five runs of the algorithm on four cases: number of spheres ($n(sph)$), overestimation (*OE*), underestimation (*UE*), weighted deviation (*WD*), number of iterations ($n(it)$) and computational time (t_{comp}). Case 3 did not converge at the given WD_{tol}. Case 4 includes results with and without convergence.

Case	$n(sph)$	*OE*	*UE*	*WD*	$n(it)$	t_{comp} [s]
(1) L-V	[6, 10]	[1.8, 5.2] $\times 10^{-2}$	[1.1, 1.5]	[1.5, 4.2] $\times 10^{-3}$	[4, 14]	[1.8, 3.5]
(2) L-V	[7, 11]	[3.8, 9.8] $\times 10^{-3}$	[2.5, 3.1]	[0.1, 7.2] $\times 10^{-4}$	[9, 15]	[2.3, 3.3]
(3) M	[296, 362]	[1.2, 1.4]	[14, 16]	[10, 12]	70	[348, 356]
(4) G	[430–599]	[1.2, 3.2]	[1.8, 1.8]	[0.5, 1.4]	[88, 100]	[2280, 2700]

3.2. Case 2) Lotka-Volterra

2 outputs and 2 estimated parameters.

For case 2, we assumed error bounds on both states ($y = [x_1, x_2]^T$. The approximate *FPS* was characterised by a maximum of 11 spheres (Fig. 6.1). The algorithm converged after a maximum of 15 iterations, for a total of 1500 model simulations.

When bounding two states, there is a reduced degree of freedom with respect to case 1, which results in a *FMOS* with smaller intervals at each measurement instant k, and thus larger *UE* (Fig. 6.2). However, as expected, the bounds on x_2 led to a smaller *OE* on this state compared to case 1.

3.3. Case 3) biodegradation and growth with Monod kinetics

1 output and 4 estimated parameters. Unidentifiability.

We tested the algorithm under a well-known case of an unidentifiable model structure. We assumed that only bacterial biomass was measured and bounded ($y = X$). The estimation of all four parameters in the Monod kinetics model resulted in an approximate *FPS* that extends indefinitely with subsequent iterations (Fig. 7.1). The projection of parameter combination ($\overline{\mu}_{max}$, $\overline{K}_S$) shows an elongated set, because estimates of these parameters are highly correlated. This result is in accordance with Holmberg (1982). Similarly, the projection of parameter combination ($\overline{Y}$, $\overline{k}_d$) also shows an elongated set, indicating unidentifiability. Practical unidentifiability of this model is also reflected in high *OE* peaks, resulting from continuously finding new areas with θ_F that must be reshaped to exclude θ_U (Fig. 7.2 and 7.3).

Therefore, high peaks of *OE* through convergence history combined with a *FPS* that extends far beyond the normalized coordinates [0,1] constitute a novel test for practical identifiability. These peaks also illustrate the opportunity to improve the mathematical rigour of the algorithm by developing a function to describe the convergence trend, for example by a moving average filter of the historical *OE* and *UE*.

3.4. Case 4) crop growth model. 1-Output bounded and 4 estimated parameters

Finally, we tested the algorithm on a model with higher complexity, GRASIM. Steps often trivialized prior to calibration of complex models are a sensitivity analysis and the implementation of prior knowledge (parameter constraints).

We performed a Sobol global sensitivity analysis on the model (Sobol, 2001) using the Python package SALib (Herman and Usher, 2017). The sensitivity analysis revealed that six out of the ten model parameters have a global sensitivity coefficient at least one order of magnitude larger than the rest. From those six parameters, a combination of four parameters is chosen with the lowest second-order sensitivity coefficients (reflecting low correlation in the parameter estimates). This pre-selection helps improving model identifiability. The selected parameters were: structural specific leaf area (a), senescence rate (β), extinction coefficient of canopy (k), and photosynthetic fraction available for growth (φ).

Subsequently, we defined parameter constraints for the algorithm based on physical definitions and on reported ranges in literature (Mohtar et al., 1997; Zhai et al., 2004). These constraints limit the region where spheres will characterise the approximate *FPS*: $a \in [0.002, 0.008]$, $\beta \in [0.02, 0.1]$ $k \in [0.25, 0.90]$, and $\varphi \in [0.75, 0.99]$.

In this case, simulation time of the model governs computational time in the algorithm implementation (ca. 2 min per iteration). Therefore, we set maximum iterations to 100. The algorithm converged to $WD_{tol} = 0.5$ in 2 out of 5 runs. In case of convergence, the approximate *FPS* was characterised by a maximum of 430 spheres in 88 iterations.

The approximate *FPS* is bounded within the constraints provided (Fig. 8.1). Notice that the projection of (a, k) is a non-convex set, in a

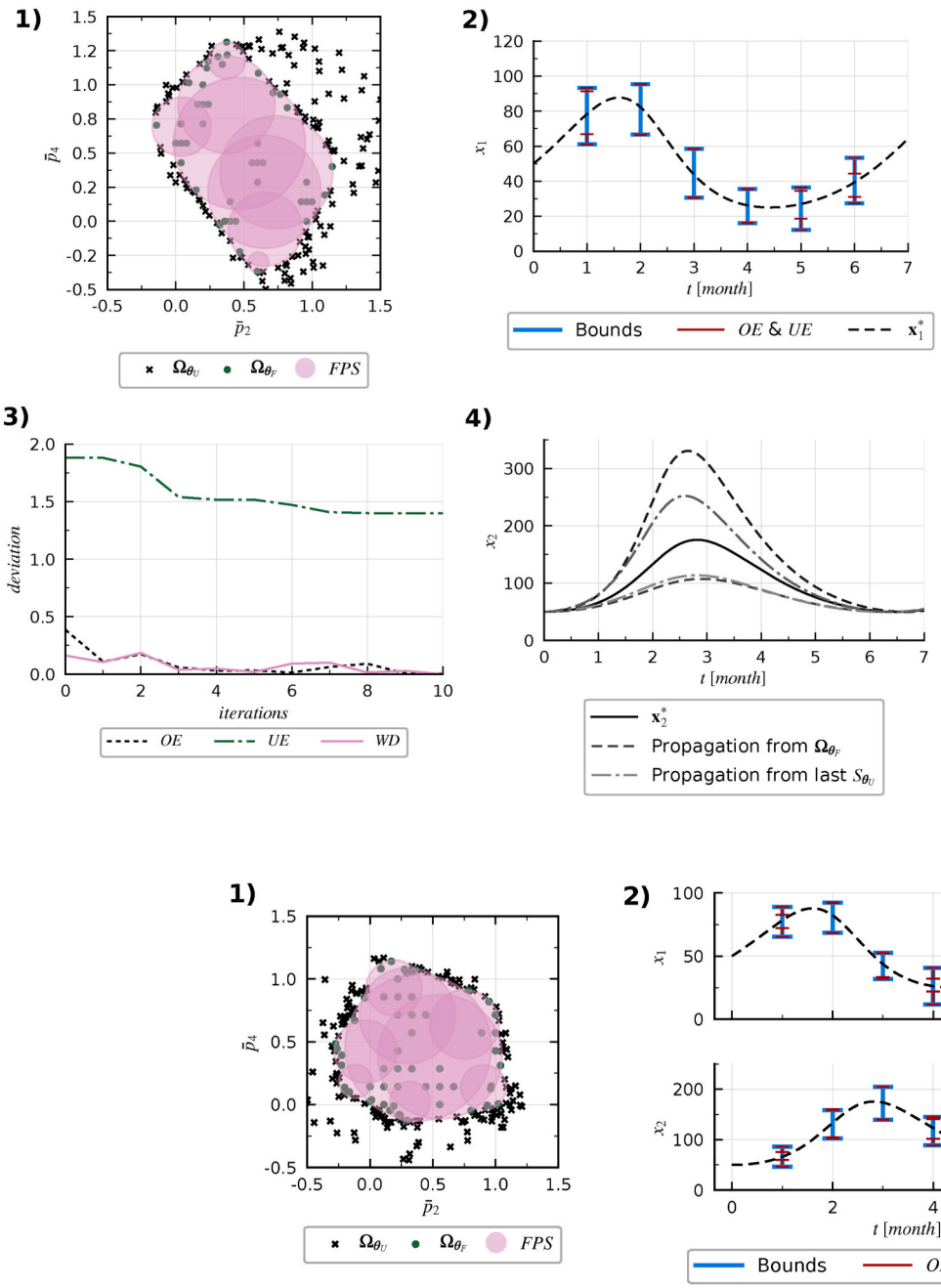

Fig. 5. Case 1. Lotka-Volterra with 1 output and 2 estimated parameters. Example results from one algorithm run. **1)** Approximate *FPS*. Normalized $\mathbb{R}^2$ ($\bar{p}_2$,$\bar{p}_4$) from $p_2 \in [0.007, 0.012]$ and $p_4 \in [0.019, 0.022]$. **2)** Reference model output, overestimation (*OE*) and underestimation (*UE*). *OE* is negligible. *UE* in the first and last two measurement instants indicate that the model cannot fully reproduce the data error bounds. **3)** Convergence. *UE* stays high but convergence is reached because few or no new θ_F were found in the last iterations, reducing *WD*. **4)** Uncertainty propagation on x_2 from the approximate *FPS*.

Fig. 6. Case 2) Lotka-Volterra with 2 outputs and 2 estimated parameters. Example result from one algorithm run. **1)** Approximate *FPS*. Normalized $\mathbb{R}^2$ ($\bar{p}_2$,$\bar{p}_4$) from $p_2 \in [0.009, 0.011]$ and $p_4 \in [0.019, 0.022]$. **2)** Reference model output, overestimation (*OE*) and underestimation (*UE*). Compared to case 1, the *FMOS* covers a lower range between the bounds in x_1 (higher *UE*), but uncertainty propagation on x_2 decreases to a negligible *OE*.

narrow space, which can still be characterised by the algorithm, but requires a relatively high number of spheres. This projection also indicates a hyperbolic (inversely proportional) relationship between a and k: higher values for specific leaf area (a) indicate thinner grass leaves, which result in lower values for canopy extinction of light (k). Notice a visible *OE* at t_k equal to 150 and 180 d (Fig. 8.2), when growth rates are maximum. Parameters a, k, φ are directly related to growth rate, and small changes in their values impact model output. It may be possible to decrease *OE* by defining sub-models for different time ranges (growth stages), thus introducing a time-varying *FPS*. However, this study focuses on the characterisation of a time-invariant *FPS*.

The model itself appears to be unidentifiable for the four parameters chosen, but the implementation of constraints based on the physical meaning of parameters prevents an unbounded *FPS*.

This case shows the relevance of a sensitivity analysis prior to parameter estimation. A similar estimation of 3 parameters for this model, resulted in a similar *OE* after convergence to *WD* < 0.5. Therefore, adding parameters for estimation may require more computational time and more memory, with little improvement in the *FMOS* accuracy, and may introduce unidentifiability to the model.

4. Conclusions

Our objective was to investigate and evaluate an efficient sampling-based method for set-membership parameter estimation. We presented a set-valued algorithm that characterises an approximate feasible parameter set (*FPS*) of (hyper)spheres, and provides a balance between accuracy and computational complexity. The spheres were obtained

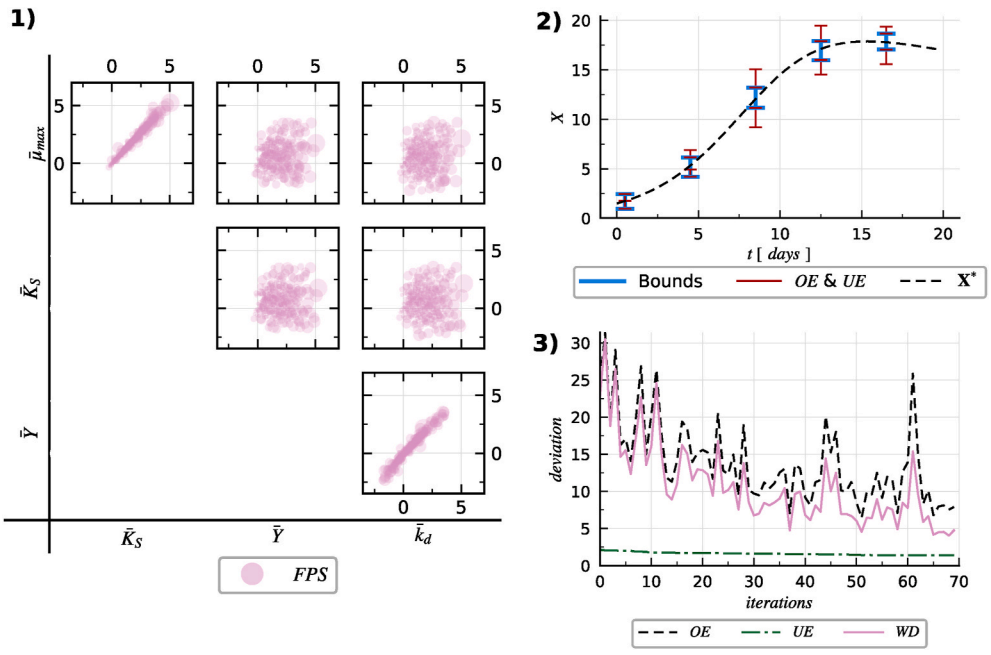

Fig. 7. Case 3) Microbial growth with Monod kinetics, with 1 output and 4 estimated parameters. Example result from one algorithm run after 70 iterations. Unidentifiable model. **1)** Approximate *FPS*. ($\mathbb{R}^2$ projections from $\mathbb{R}^4$ space). Normalized parameter space ($\bar{\mu}_{max}, \bar{K}_S, \bar{Y}, \bar{k}_d$) from $\mu_{\max} \in [0.79, 1.49]$, $K_S \in [3.09, 6.87]$, $Y \in [9.60, 11.07]$, $k_d \in [1.41 \times 10^{-2}, 2.71 \times 10^{-2}]$. Due to unidentifiability of the model structure, the approximate *FPS* continues to extend with every subsequent iteration far beyond the normalized coordinates [0, 1]. **2)** Reference model output, overestimation (*OE*) and underestimation (*UE*). Compared to the previous cases, *OE* cannot be neglected. **3)** Convergence. After 50 iterations, *OE* and weighted deviation (*WD*) stop decreasing because new θ_U and θ_F continue to be found with every subsequent iteration. The approximate *FPS* is thus not bounded, reflecting unidentifiability of the model.

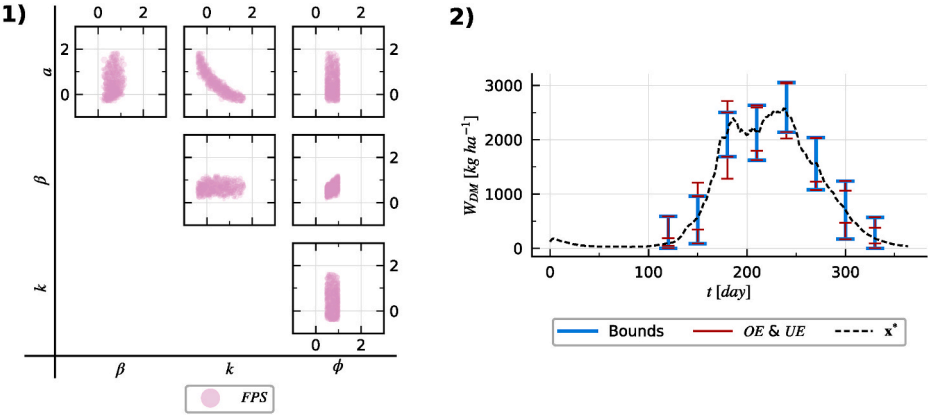

Fig. 8. Case 4) Grass growth in GRASIM model, with 1 state bounded and 4 estimated parameters. Example result from one algorithm run. **1)** Approximate *FPS*. ($\mathbb{R}^2$ projections from $\mathbb{R}^4$ space). Normalized parameter space ($\bar{\alpha}, \bar{\beta}, \bar{K}, \bar{\varphi}$) from $\alpha \in [0.0030, 0.0056]$, $\beta \in [0.027, 0.061]$, $k \in [0.394, 0.688]$, $\varphi \in [0.614, 0.963]$. Although unidentifiable, an approximate *FPS* can be characterised due to the constraints on parameter values based on literature and their physical meaning. **2)** Reference model output, overestimation (*OE*) and underestimation (*UE*). At $t_k = 150, 180$ *d*, when growth rates are higher, *OE* becomes larger.

using randomly sampled unfeasible parameter vectors as seeds for a Voronoi diagram of the parameter space, and using the resulting Voronoi vertices as centres for the spheres. The algorithm works with a fixed number of sampled parameter vectors per iteration, i.e. the computational time has an order $O(n(sample))$. Although lacking an exact mathematical description, the approximate *FPS* can generally be characterised by a low number of spheres.

We demonstrated an implementation on a 2-parameter case, achieving lower computational requirements and similar accuracy, as compared with the SIVIA algorithm. We also demonstrated implementation on a 4-parameter case of a relatively complex model.

Additionally, we proposed a measure of the accuracy based on deviations between feasible model output set (*FMOS*) and data error bounds. When analysing both *FPS* and these deviations, it is possible to indicate practical identifiability of a model, as we demonstrated on a 3-parameter case.

Funding

This work was done as part of the project INAPRO, with funding from the European Union's Seventh Framework Programme for research, technological development and demonstration under Grant Agreement No. 619137.D:\MYFILES\ELSEVIER\ENSO\00105125\SCE\gs2

Software availability

The source code is available at https://sourceforge.net/p/vorsetmembership/code/ci/master/tree/

Credit authorship contribution statement

D. Reyes Lastiri: Conceptualization of this study, Writing, Methodology, Software. **H.J. Cappon:** Methodology, Analysis of results, Writing. **K.J. Keesman:** Methodology, Analysis of results, Writing.

Declaration of competing interest

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

A. Algorithm supporting functions

Algorithm A.1. Supporting functions

Algorithm A.1 Supporting functions

Require: $r_{min}, r_{max}, \vartheta_{min}, \vartheta_{max}$
  function OPT FILTER(c, r)
    ▷ Optional filter to remove small, large and distant spheres
    $r, c \leftarrow r, c$ where $[r > r_{min}$ and $r < r_{max}$ and $all(c_j < \vartheta_{max,j})$ and $all(c_j > \vartheta_{min,j})]$
  end function

Require: $c, r, c\theta_F$
  function FILTER($c, r, c\theta_F$)
    ▷ Filter to remove empty spheres
    $c, r, c\theta_F \leftarrow c, r, c\theta_F$ where ROWSUM($c\theta_F$) > 0
    ▷ Filter to remove redundant spheres
    $Sum\theta_F \leftarrow$ COLSUM($c\theta_F$)
    $SumRed \leftarrow$ ZEROS($(1, p)$)
    for $c\theta_{F,i}$ in $c\theta_F$ **do**
      $RedTest \leftarrow c\theta_{F,i} \times (Sum\theta_F - SumRed)$
      if ANY($RedTest_j == 1$) **then**
        $c_{i,filtered}, r_{i,filtered}, \theta_{F,i,filtered} \leftarrow c_i, r_i, c\theta_{F,i}$
      else
        $SumRed \leftarrow SumRed + c\theta_{F,i}$
      end if
    end for
  end function

Require: $\vartheta_F, \vartheta_U, \vartheta_{F,min}, \vartheta_{F,max}$
  function NORMALIZE($\vartheta_F, \vartheta_U, \vartheta_{F,min}, \vartheta_{F,max}$)
    ▷ Normalize nominal parameter sets to initialized coordinates of ϑ_F
    $\theta_F \leftarrow (\vartheta_F - \vartheta_{F,min})/(\vartheta_{F,max} - \vartheta_{F,min})$
    $\theta_U \leftarrow (\vartheta_U - \vartheta_{F,min})/(\vartheta_{F,max} - \vartheta_{F,min})$
  end function

Require: $\theta_F,\ \theta_U$
  function NOMINAL($\theta_F, \theta_U, \vartheta_{F,min}, \vartheta_{F,max}$)
    ▷ Returns normalized parameters to nominal values, based on initial coordinates of θ_F
    $\vartheta_F \leftarrow \theta_F(\vartheta_{F,max} - \vartheta min) + \vartheta_{F,min}$
    $\vartheta_U \leftarrow \theta_U(\vartheta_{F,max} - \vartheta min) + \vartheta_{F,min}$
  end function

B. List of symbols and Model parameters

Symbol	Description
$\|\|\|e_\infty$	∞-norm of the error vector, defined as $\|\|\|e_\infty := \max_{1 \le i \le n} \|e_i$
FPS and *FMOS*	Feasible parameter set, and feasible model output set
k, M	Measurement instant, and total of measurement instants
$n(\,)$	Cardinality of a set (number of elements)
$O(\,)$	Order of the algorithm (time complexity)
OE, *UE*, *WD*	Overestimation, underestimation and weighted deviation
$\mathbb{R}^p$	p-dimensional parameter space
S_ϑ	Sample of parameter vectors
ϵ	Upper error bound
ϑ, (θ) - *F* or *U*	Parameter vector (normalized). Feasible or unfeasible (subscripts)
Ω_ϑ, (Ω_θ)	Set of all known parameter vectors (normalized)
Lotka-Volterra model	
x_1, x_2 (50, 50)	State variables (initial value), populations of prey and predator
t_{sim} = [0, 1, …, 7*30] *day*	Simulation time

(*continued on next page*)

(*continued*)

Symbol	Description
$t_k = [1, 2, \ldots, 6]$ *month*	Artificial measurement instants
$\mathbf{x}_1^*, \mathbf{x}_2^*$	Reference model outputs
Parameters (ref. value)	
$\overline{p}_2, \overline{p}_4$	Estimated, normalized parameters
p_1 ($\mathbf{p}_1^* = 1.0\ month^{-1}$)	Birth rate of prey
p_2 ($\mathbf{p}_2^* = 0.01\ month^{-1} pred.^{-1}$)	Population decrease of prey due to predators
p_3 ($\mathbf{p}_3^* = 1.0\ month^{-1}$)	Predator death rate
p_4 ($\mathbf{p}_4^* = 0.02\ month^{-1} prey^{-1}$)	Population increase of predators due to prey
Microbial growth model	
$[S, X]$ (2.0, 1.5) mg/L	State variables (initial value), concentrations of substrate and bacteria
$t_{sim} = [0, 1/24, \ldots, 20]\ d$	Simulation time
$t_k = [0.5, 3.5, \ldots, 16.5]\ hr$	Artificial measurement instants
$\mathbf{S}^*$, $\mathbf{X}^*$	Reference model outputs
Parameters (ref. value)	
$\overline{k}_d, \overline{K}_S, \overline{Y}, \overline{\mu}_{max}$	Estimated, normalized parameters
k_d ($\mathbf{k}_\mathbf{d}^* = 0.021\ d^{-1}$)	Endogenous bacterial decay
K_S ($\mathbf{K}_\mathbf{S}^* = 5.14\ mgL^{-1}$)	Half-velocity constant
Y ($\mathbf{Y}^* = 10.34\ mg_X mg_S^{-1}$)	True microbial yield per substrate consumed
μ_{max} ($\boldsymbol{\mu}_{\boldsymbol{max}}^* = 1.15\ d^{-1}$)	Maximum rate of substrate use
GRASIM model	
$[W_s, W_g]$ (50, 50) $kgC\ ha^{-1}$	State variables (initial value), storage and structure mass
$t_{sim} = [0, 1, \ldots, 365]\ d$	Simulation time
$t_k = [120, 150, \ldots, 330]\ hr$	Artificial measurement instants
$\mathbf{W}_\mathbf{DM}^*\ kgDM\ ha^{-1}$	Reference model output, dry matter of plant structure mass
Parameters (ref. value)	
$\overline{a}, \overline{\beta}, \overline{k}, \overline{Y}$	Estimated, normalized parameters
a ($\mathbf{a}^* = 0.004$ ha kg^{-1})	Structural specific leaf area
α ($\boldsymbol{\alpha}^* = 2.4 \times 10^{-9}\ kg\ J^{-1}$)	Leaf photosynthetic efficiency
β ($\boldsymbol{\beta}^* = 0.05$)	Senescence rate
k ($\mathbf{k}^* = 0.5$)	Canopy extinction coefficient
m ($\mathbf{m}^* = 0.1$)	Leaf transmission coefficient
μ_m ($\boldsymbol{\mu}_\mathbf{m}^* = 0.5\ d^{-1}$)	Max. structural specific growth rate
φ ($\boldsymbol{\varphi}^* = 0.9$)	Photosynthetic fraction available for growth
P_m ($\mathbf{P}_\mathbf{m}^* = 10000\ kg\ ha^{-1}\ d^{-1}$)	Max. photosynthesis
Y ($\mathbf{Y}^* = 0.75$)	Structure fraction from storage
M ($\mathbf{M}^* = 0.02\ d^{-1}$)	Maintenance respiration coefficient

C. Grasim auxiliary equations

Photosynthetic growth is defined as:

$$f_P = (12/44)\varphi P \tag{A.1}$$

where (12/44) is a conversion factor for mass from CO_2 to C.

The photosynthesis rate is defined as:

$$\begin{aligned} P &= \int_0^{LAI} \int_0^h Pg\, \mathrm{d}l\, \mathrm{d}\tau \\ Pg &= \frac{\alpha I P_m}{\alpha I + P_m} \\ I &= \frac{k}{1-m} I_0 e^{-kl} \end{aligned} \tag{A.2}$$

where LAI is the leaf area index, h is the daytime duration, P_g is the gross photosynthetic rate, I is the light intensity over a leaf, and I_0 is the irradiance.

Assuming $h = 24$hr and $I_0 = DLI$ (day light integral), it is possible to obtain an approximate analytical solution for P:

$$\begin{aligned} P &= \frac{P_m}{k} ln\left(\frac{A + Pm}{Ae^{-k*LAI} + Pm}\right) \\ A &= \alpha\ DLI \frac{k}{1-m} \end{aligned} \tag{A.3}$$

where DLI for the test case was taken from the weather station De Bilt for year 2001 (KNMI).

The leaf area index is:

$$LAI = a\ W_g \tag{A.4}$$

The net mass conversion from storage to structure is:

$$f_G = \mu_m \frac{W_g W_s}{W_g + W_s} \tag{A.5}$$

Shoot respiration from storage mass is:

$$f_{SR} = \mu_m \frac{1 - Y}{Y} \frac{W_g W_s}{W_g + W_s} \tag{A.6}$$

Maintenance respiration (covered entirely by storage mass) is:

$$f_{MR} = M\ W_g \tag{A.7}$$

Senescence is:

$$f_S = \beta\ W_g \tag{A.8}$$

References

Barber, C.B., Dobkin, D.P., Huhdanpaa, H., 1996. The quickhull algorithm for convex hulls. ACM Trans. Math Software 22, 469–483. https://doi.org/10.1145/235815.235821.

Cerone, V., Lasserre, J.B., Piga, D., Regruto, D., 2014. A unified framework for solving a general class of conditional and robust set-membership estimation problems. IEEE Trans. Automat. Contr. 59, 2897–2909. https://doi.org/10.1109/TAC.2014.2351695.

Diniz, P.S., 2013. Adaptive Filtering, 4 ed. Springer. https://doi.org/10.1007/978-1-4614-4106-9.

Herman, J., Usher, W., 2017. Salib: an open-source python library for sensitivity analysis. J. Open Source Software 2. https://doi.org/10.21105/joss.00097.

Holmberg, A., 1982. On the practical identifiability of microbial growth models incorporating michaelis-menten type nonlinearities. Math. Biosci. 62, 23–43. https://doi.org/10.1016/0025-5564(82)90061-X.

Jaulin, L., Walter, E., 1993. Set inversion via interval analysis for nonlinear bounded-error estimation. Automatica 29, 1053–1064. https://doi.org/10.1016/0005-1098(93)90106-4.

Keesman, K., 1990. Membership-set estimation using random scanning and principal component analysis. Math. Comput. Simulat. 32, 535–543. https://doi.org/10.1016/0378-4754(90)90009-8.

Keesman, K., 2003. Bound-based identification: nonlinear-model case. In: Unbehauen, H. (Ed.), Encyclopedia of Life Science Systems. UNESCO EOLSS (chapter 6).43.11.2.

Keesman, K., Van Straten, G., 1990. Set membership approach to identification and prediction of lake eutrophication. Water Resour. Res. 26, 2643–2652. https://doi.org/10.1029/WR026i011p02643.

Keesman, K.J., 2011. System Identification: an Introduction. Springer Science & Business Media, London.

KNMI, . Daggegevens van het weer in nederland. URL: https://knmi.nl/nederland-nu/klimatologie/daggegevens.

Madan, H., 2017. Sampling uniformly in a multidimensional ring without rejection. URL: https://stackoverflow.com/questions/47472123/sample-uniformly-in-a-multidimensional-ring-without-rejection/47492146.

Marvel, S.W., Williams, C.M., 2012. Set membership experimental design for biological systems. BMC Syst. Biol. 6, 21. https://doi.org/10.1186/1752-0509-6-21.

Milanese, M., Norton, J., Piet-Lahanier, H., Walter, E., 1996. Bounding Approaches to System Identification. Plenum Press, New York, NY.

Mocenni, C., Vicino, A., 2006. Modelling ecological competition between seaweed and seagrass: a case study. IFAC Proc. 39, 732–737. https://doi.org/10.3182/20060329-3-AU-2901.00114.

Mohtar, R., Buckmaster, D., Fales, S., 1997. A grazing simulation model: grasim a: model development. Trans. ASAE 40, 1483–1493. https://doi.org/10.13031/2013.21370.

Norton, J., 1987. Identification and application of bounded-parameter models. Automatica 23, 497–507. https://doi.org/10.1016/0005-1098(87) 90079-3.

Norton, J., 2003. Bound-based identification: linear-model case. In: Unbehauen, H. (Ed.), Encyclopedia of Life Science Systems. UNESCO EOLSS (chapter 6).43.11.2.

Nurulhuda, K., Struik, P., Keesman, K., 2017. Set-membership estimation from poor quality data sets: modelling ammonia volatilisation in flooded rice systems. Environ. Model. Software 88, 138–150. https://doi.org/10.1016/j.envsoft.2016.11.002.

Raissi, T., Ramdani, N., Candau, Y., 2004. Set membership state and parameter estimation for systems described by nonlinear differential equations. Automatica 40, 1771–1777. https://doi.org/10.1016/j. automatica.2004.05.006.

Raïssi, T., Ramdani, N., Candau, Y., 2009. Set membership parameter estimation in the frequency domain based on complex intervals. Int. J. Contr. Autom. Syst. 7, 824. https://doi.org/10.1007/s12555-009-0515-y.

Reyes Lastiri, D., Geelen, C., Cappon, H.J., Rijnaarts, H.H., Baganz, D., Kloas, W., Karimanzira, D., Keesman, K.J., 2018. Model-based management strategy for resource efficient design and operation of an aquaponic system. Aquacult. Eng. 83, 27–39. https://doi.org/10.1016/j.aquaeng.2018.07.001.

Rumschinski, P., Borchers, S., Bosio, S., Weismantel, R., Findeisen, R., 2010. Set-base dynamical parameter estimation and model invalidation for biochemical reaction networks. BMC Syst. Biol. 4, 69. https://doi.org/10.1186/1752-0509-4-69.

Sobol, I.M., 2001. Global sensitivity indices for nonlinear mathematical models and their Monte Carlo estimates. Math. Comput. Simulat. 55, 271–280. https://doi.org/10.1016/S0378-4754(00)00270-6.

The SciPy community, b. Scipy voronoi. URL: https://scipy.github.io/devdocs/generated/scipy.spatial.Voronoi.html.

Van Straten, G., Keesman, K.J., 1991. Uncertainty propagation and speculation in projective forecasts of environmental change: a lake-eutrophication example. J. Forecast. 10, 163–190. https://doi.org/10.1002/for.3980100110.

Walter, E., 2003. Bound-based identification. In: Unbehauen, H. (Ed.), Encyclopedia of Life Science Systems. UNESCO EOLSS (chapter 6).43.11.2.

Walter, E., Piet-Lahanier, H., 1990. Estimation of parameter bounds from bounded-error data: a survey. Math. Comput. Simulat. 32, 449–468. https://doi.org/10.1016/0378-4754(90)90002-Z.

Zhai, T., Mohtar, R., El-Awar, F., Jabre, W., Volenec, J., 2004. Parameter estimation for process-oriented crop growth models. Trans. ASAE 47, 2109. https://doi.org/10.13031/2013.17796.