H2020-SFS-2018-2020

DECIDE

Data-driven control and prioritisation of non-EU-regulated contagious animal diseases

Deliverable D2.4

Open-source inference algorithm adaptable to specific cases

WP2 – Methods for data analysis and modelling

Authors Halifa Farchati (INRAE), Gaël Beaunée (INRAE), Pauline Ezanno (INRAE)
Lead participant INRAE
Delivery date 28 June 2024
Dissemination level Public
Type Other

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 101000494.

Revision History

Author Name (Partner short name)	Description	Date
Halifa Farchati (INRAE), Gaël Beaunée (INRAE), Pauline Ezanno (INRAE)	Draft deliverable	31.05.2024
Gerdien van Schaik (UU), Arjan Stegeman (UU), Jerrold M. Tubay (UU)	Revision	20.06.2024
Halifa Farchati (INRAE), Gaël Beaunée (INRAE), Pauline Ezanno (INRAE)	Final version	28.06.2024

Partner short names

Short name	Organisation
INRAE	Institut National de Recherche pour l'Agriculture, l'Alimentation et l'Environemment
UU	Utrecht University

Content

Executive Summary

Epidemic mechanistic models are crucial for understanding and predicting pathogen spread in host populations. To improve the robustness and precision of these models, parameter values should be inferred from highly heterogeneous field observations. We chose Approximate Bayesian Computation (ABC) methods, specifically ABC-regression and ABC-SMC, for their adaptability to complex models and ability to bypass likelihood computations.

This deliverable introduces "BRREWABC," a new open-source inference algorithm, along with a comprehensive tutorial for academics. It also covers the "abc" package as an alternative. We developed "BRREWABC" based on ABC-SMC methods and used the "abc" package for ABC-regression, the latter being developed by Katalin et al. (2012). To test both these ABC approaches in contrasted situations, we applied them to a stochastic epidemiological model of Bovine Respiratory Diseases (BRD) in young cattle, using synthetic data as observed ones for parameter estimation.

The results show well-defined peaks in posterior distribution densities, confirming algorithm convergence and robustness. Although there may be occasional under- or overestimations, model fidelity to epidemiological dynamics is strong, as shown by the close alignment of simulated and observed trajectories.

While "BRREWABC" requires significant computational resources and hardware investment, it provides robust and accurate estimates through parallelized ABC-SMC. The "abc" package, though demanding substantial time and resources initially and struggling with high-dimensional problems, offers ease of use and the ability to reuse initial samples, creating a valuable data library for future studies. The choice between these two packages clearly depends on study needs and available resources.

Objectives of the Deliverable

With the help of this deliverable, developers of epidemic mechanistic models will be able to better estimate parameter values of their model. It will also help users to implement these algorithms when studying specific (real) cases.

Activities

The deliverable examines the deployment processes of "abc" and "BRREWABC" packages, which allow for parameter estimation from any pre-defined mechanistic model. It can be seen as a tutorial and contains a R code adaptable to specific user case. It also highlights the key points, advantages, and limits of both methods. Additionally, to illustrate the deployment of both packages, they were applied to a BRD model in contrasted scenarios in terms of process stochasticity, type of observations, etc., and using synthetic data to control available data knowledge. The obtained results are presented and discussed, detailing the advantages and limits of each package.

Outcome

With the provided R scripts, which include walkthroughs for both inference processes and illustrative examples, readers should be able to apply these inference procedures to their own case studies.

Next steps

The next natural step is to apply the two inference algorithms described to real-world data related to BRD in cattle farms. This future step will help develop a support decision tool for on-farm BRD case management. Results will be disseminated through publications and scientific and professional communications.

Glossary

Prior: a prior distribution for the parameters represents our initial beliefs about the distribution of the parameters before considering the observed data.

Posterior: is the updated distribution for the parameter after considering the data. It reflects a combination of the prior beliefs and the new evidence provided by the data.

Parameter set: the set of parameters to be estimated.

Draw: a random sample of data generated from a probability distribution.

Summary statistics: measures of the essential characteristics of a data set that are used to summarize and understand the data without having to examine each individual observation.

Synthetic data: data generated from a specified statistical model rather than collected empirically.

Particles: set of parameter values

Kernels: In the ABC-SMC algorithm, kernels are propositional distributions used to define how new particles are sampled around existing particles to better represent the posterior distribution of parameters. Kernels can take different forms, such as Gaussian, uniform or based on specific optimization methods. For example, a Gaussian kernel adds Gaussian noise to existing particles to generate new particles, while a uniform kernel samples uniformly around current particles. The choice of kernel is crucial, as it influences the convergence speed of the algorithm and its ability to efficiently explore the parameter space.

Credibility Interval: Interval represents ranges of values within which the unknown parameter has a specified probability of being located, taking into account both observed data and a prior knowledge.

The posterior trajectories: are the set of trajectories that were simulated using the (last) particles that were accepted.

Abbreviations

Abbreviation	Description
ABC	Approximate Bayesian Computation
ABC MCMC	Approximate Bayesian Computation Markov Chain Monte Carlo
ABC-SMC	Approximate Bayesian Computation Sequential Monte Carlo
ABC-Regression	Regression approaches for Approximate Bayesian Computation
BRD	Bovine Respiratory Disease
BRSV	Bovine Respiratory Syncytial Virus
WP	Work Package

INTRODUCTION

Background

Mechanistic models are very useful to test, better understand, and explain the processes and mechanisms involved in population dynamics. They are also helpful to better understand and anticipate the spread of pathogens in host populations in order to prioritize prevention or control strategies. However, accounting for the inherent variability of biological and ecological systems often need the use of stochastic dynamic simulation models, with the major disadvantage that, to infer their parameter values from observed data, likelihood functions are difficult, if not impossible, to calculate explicitly. Nevertheless, the evaluation of their robustness and predictive ability is essential and should rely on calibrating such models with field observations, considering data (often heterogeneous) characteristics. To overcome this limitation and to meet the above-mentioned needs (calibration and quality assessment), several recent methods, among them Approximate Bayesian Computations, can be used, which do not require likelihood calculations and can be adapted to a wide range of situations.

Approximate Bayesian Computations are methods based on the comparison of observed (real) data with synthetic (simulated) data produced by a given model mimicking observed processes. This comparison is made by assessing the closeness between simulated and real data. The parameter values used to generate the synthetic data are successively accepted or rejected according to their ability to "faithfully" reproduce the observed data. This acceptance-rejection process is guided by an appropriate measure of the distance between simulated and real data. The set of accepted parameter values is used to approximate the posterior distribution of the target parameter set.

The evolution of these methods, well described in [1, 2], has been characterized by the emergence of several variants, such as Approximate Bayesian Computation Sequential Monte Carlo (ABC-SMC), Regression approaches for Approximate Bayesian Computation (ABC-Regression), Approximate Bayesian Computation Markov Chain Monte Carlo (ABC-MCMC), among others. Each variant proposes mechanisms for improving given aspects of approximate Bayesian inference, such as a more thorough exploration of the parameter space and more efficient selection of the parameter sets that will be used to construct the posterior distribution, two fundamental steps in the statistical inference process. Early approaches (ABC-rejection) used randomly generated samples of parameter values according to specific probability distributions and then compared simulated data with observed data [3, 4, 5]. The introduction of the ABC-SMC variant allowed a more accurate estimation of the posterior distribution using sequential sampling and weighted kernels [6]. Subsequently, ABC-Regression [7, 8] was developed to incorporate regression techniques into the estimation process and improve computational efficiency by approximating the distance function between simulated and observed data. Concurrently, ABC-MCMC [9], which uses Markov chain techniques to explore the parameter space, was developed. Other approaches using bootstrap techniques, Bayesian networks, neural networks, etc. have also been introduced. These relatively new methods are still the subject of active research to explore their capabilities and limitations, with the goal of making significant improvements in response to challenges in various fields.

Deliverable objective & content

This DECIDE project deliverable D2.4 is both a new open-source inference algorithm and a tutorial to guide users, particularly those with a working knowledge of R and mechanistic modeling. In the report part, we describe these two inference algorithms, and how they can be used to facilitate the integration of heterogeneous or incomplete data sources in stochastic simulation models, with a focus on livestock epidemiology as case studies. While the considered methods are applied to stochastic models here, they are formulated with

flexibility in this report to ease their adaptation to specific contexts, diverse model types, and varying data characteristics. This will enhance their future application to novel challenges and promote their broader acceptance within the scientific community.

Among available inference methods, ABC methods prove to have a high adaptability to complex models, versatility in model selection, ability in dealing with diverse data, and proficiency in estimating complicated posterior distributions. We selected two different ABC variants, ABC-regression and ABC-SMC, because they cover a wide range of parameter space exploration techniques. ABC-Regression is characterized by its computational efficiency, using regression methods to approximate the divergence between simulated and observed data, thus reducing computational complexity. It is also robust to the choice of distance parameters. Its simple formulation facilitates implementation, making it a preferred option in resource-constrained scenarios. Conversely, ABC-SMC demonstrates remarkable adaptability to complicated models characterized by challenging posterior distribution exploitation, facilitated by its sequential and weighted sampling approach. It also demonstrates an ability to accommodate to complex data structures and multimodal posterior distributions, achieving rapid convergence to the desired posterior distribution. The selection of these two ABC variants is driven by their efficacy in effectively addressing the specific inference challenges posed by complex and heterogeneous contexts, and their easiness to be used by the scientific community in epidemiology. As a result, two R packages are used: the "abc" package (an R package for approximate Bayesian computation) developed by Katalin et al (2012) [10], and a new R package "BRREWABC" [11] (Batched Resilient and Rapid Estimation Workflow through Approximate Bayesian Computation), which has just been developed in DYNAMO team (BIOEPAR unit, Nantes) at INRAE.

This report is structured in 3 sections. Section 1 examines the deployment processes of the "abc" and "BRREWABC" packages. It provides a detailed analysis of the implementation of each package, covering aspects such as package installation, data preparation, tolerance selection, among others. Section 2 provides an in-depth exploration of four case studies. These examples demonstrate the use of one or both of the "abc" and "BRREWABC" packages in contrasted situations, covering a range of scenarios involving different types of observational datasets, as well as different circulating pathogens. In addition, particular attention is paid to highlighting the stochastic nature of the model, whether in the context of a simple or a structured host population. The chosen examples highlight the possibility of reusing previously obtained samples for future studies. Finally, the concluding section (section 3) provides a synthesis and thorough discussion of the procedures, highlighting the advantages and limitations of both approaches.

The structure of our directory (click here to access to the git repository) is divided into four main folders. `Example1`", "`Example2`", "`Example3`", "`Example4`", each containing all the necessary information to reproduce the corresponding example. Each of these folders contains three subfolders. The Git repository also includes a "`README.md`" file that provides a detailed explanation of the repository's contents.

`Model` folder contains the BRD model along with all the dependencies necessary for its operation:

- "**BRD_model.yaml**": This file contains the BRD model used for the examples in Section 2.
- "**multibatch_simplified.py**": This file contains the code required for using the "BRD_model.yaml" file.
- "**run_emulssion.R**": R script which contains the `toy_model` function, which is used to run BRD_model.yaml, the epidemiological model build using an INRAE software, EMULSION [13]. The toy_`model` function simulates BRD pathogen spread for one of two possible pathogens, BRSV and *M. haemolytica*. In this function, the value of "`batch_size`" can be specified to the number of animals per batch (e.g., 10); the value of "`number_batches`" can be specified to the number of batches (e.g., 3); the Boolean "`viral_infection`" indicates if BRSV is circulating (1: yes); the boolean "`pasteurella`" indicates if *M. haemolytica* is circulating (1: yes).

`Scripts` folder contains useful scripts for tasks of the deliverable (both scripts if both methods were used; otherwise, it contains only one script named **runABCpackage.R**):

- **"runABCpackage.R":** R script running the "abc" package. It uses simulated data to illustrate the steps involved in running the inference process.
- "**runBRREWABCpackage.R**": R script running the "BRREWABC" package using the BRD model.

`Data` folder contain the essential data needed to execute illustrative selected examples, thereby facilitating example understanding and application to future practical case studies. In this folder, we do not specify each content, for more details, please refer to the "`README.md`" file.

SECTION 1 - IMPLEMENTATION PROCESS OF PACKAGES « abc » AND « BRREWABC »

Overview of ABC approaches

Approximate Bayesian Computation (ABC) is a class of computational methods used for performing approximate Bayesian inference (i.e. estimate the parameter values of a given model) in scenarios where the likelihood function is analytically intractable or computationally expensive to evaluate. ABC methods are particularly useful for models based on stochastic simulations, such as those often used in epidemiology and to study other complex systems.

The basic idea behind ABC is to generate simulated data from the model with different parameter values and then compare these simulated data to the observed data using summary statistics. Parameter values that produce simulated data "close" to the observed data, as measured by the distance between their summary statistics, are retained as samples from an approximation to the posterior distribution.

While there are differences in the way they estimate the posterior distributions, **ABC-SMC (package "BRREWABC") and ABC-Regression ("abc" package)** do share several common steps in their process (Figure 1).

ABC algorithms typically involve the following steps:

1. **Draw a particle**: sample a candidate parameter set from the prior distributions. The prior distribution represents the synthesis of prior knowledge about the parameters from various sources, such as previous data, similar studies, expert opinion, or theoretical hypotheses. This distribution reveals our initial uncertainty about the parameter values and provides a probabilistic basis for adjusting our beliefs according to the observed data. If no prior knowledge is available, uniform distributions are used.
2. **Generate simulated data** using the model and the particle (sampled parameter value).
3. **Calculate summary statistics** for the simulated data and for the observed data.
4. **Compare summary statistics**: if the distance between observed and simulated summary statistics is lower or equal to the tolerance threshold, accept the particle as a sample from the approximate posterior distribution. Otherwise, reject the particle.
5. Repeat steps 1-4 until the desired number of accepted particles is obtained.
6. **Estimate the posterior distribution**: The resulting set of accepted particles is then used to estimate the posterior distribution of the model parameters estimated.

ABC methods have several advantages, including their ability to handle complex models and their flexibility in choosing summary statistics. However, they also have limitations, such as the potential loss of information due to the use of summary statistics and the need to choose appropriate distance functions and tolerance levels.

Notation:

D: the observed data set

D': the simulated data set

θ: the particle (parameter set)

$S = s(D)$: the observed summary statistics

$S' = s(D')$: the simulated summary statistics

d: the distance function

ϵ : the tolerance threshold

Inputs:

π: the prior distribution of parameters

M: the model

Procedure:

1. Draw a particle θ_i, $i = 1, \ldots, k$ from the prior distribution π
2. Generate simulated data D_i' using the model M and particle θ_i
3. Calculate the summary statistics S and S'
4. Accept θ_i if S_i' "resembling" S $(d(S, S') \leq \epsilon)$
5. Repeat steps 1-4 until the desired number of accepted parameter values is obtained
6. Use all θ_i accepted to construct the posterior distribution of θ

Figure 1. Principles of approximate Bayesian methods (ABC)

Although the **ABC-SMC ("BRREWABC" package)** and **ABC-Regression ("abc" package)** approaches share a common process, it is essential to highlight the differences between them.

- In ABC-Regression ("abc" package): in step 6, regression models are deployed:

Use of regression models: After particles acceptance and rejection, regression models are employed. These models are trained, using the summary statistics of the simulated datasets of those accepted particles as predictors and the corresponding particle values as the response variable. Following model training, the regression models can predict particle values based on the summary statistics of new simulated datasets, obviating the necessity for a predefined acceptance threshold. The predicted particle values furnished by the regression models serve as estimates and are used to form the posterior distribution (step 8) of the parameter target.

- In ABC-SMC ("BRREWABC" package), an iterative approach is used with two additional steps:

Particle weighting: After step 4 (comparing the summary statistics), a weight is given to the particle based on its fit to the observed data, where particles yielding simulated data more similar to the observed data receive higher weights, while those producing less similar data receive lower weights.

Particle resampling: At each iteration (except the first one), new particles are generated by random sampling from the weighted particles of the previous accepted population. This approach targets the generation of

new particles towards regions of the parameter space best suited to reproduce the observed data, while allowing for broader exploration of this space. Finally, the sample of particles from the last iteration is used to form the posterior distribution (step 6) of the target parameters.

The ABC-Regression (through “abc” package) offers the advantage of being able to produce new estimates very quickly, assuming that new data have the same structure and that the model considered remains the same as the one used to construct the reference table (set of particles tested, steps 1 to 3 in Figure 1). The ABC-SMC algorithm ("BRREWABC" package) is reputed to be generally more computationally efficient and to converge more quickly towards the posterior distribution, and therefore seems more appropriate when the data structure and/or the model change frequently.

Using the « abc » package

Let's consider the scenario where we aim to estimate the uni-dimensional or multi-dimensional parameter θ using the **"abc" package**.

Installation of the package

Install the “abc” package and load the package into the R environment using the following command:

```
install.packages("abc") # Package installation
library("abc") # loading package: every time you want to use the "abc"
```

Preparation of data for ABC-regression method

Preparing your data for ABC analysis is the most important step, and one that sometimes requires a great deal of time and care. To use the package, the following R objects should be prepared:

The model (called “toy.model” in this document and associated codes) and prior distribution definitions, simulated parameters “par.sim”, simulated summary statistics “stat.sim”, and observed summary statistics “stat.obs”

We start by defining the model, named “toy.model” throughout this document, which is a function which corresponds to the mechanistic model to be used, that takes a parameter vector as input. It executes a set of tasks defined by the user based on the study being conducted. Then, it returns summary statistics based on the obtained results.

```
#-------------------------------------------------------------------
#  Model definition 'toy.model'
#-------------------------------------------------------------------
toy.model <- function(parameters_vector)
{
  ... # Perform a series of tasks based on the study
  ... # Calculate summary statistics 'stat.sim'

 return (stat.sim)
}
```

Then, it is necessary to select the prior distribution for the parameter set to be estimated, based on information available. If we have access to specific knowledge about the distribution of parameters, we can use a particular distribution. Otherwise, we choose a uniform distribution, which is poorly informative but necessary and sufficient if the parameter distribution is unknown.

As for the choice of the size n of the dataset of parameters to be generated, it is left to the discretion of the user to choose a sufficiently large number. This choice must be judicious, as a high number means more computing time, while a small number may not be sufficient to obtain good estimates or to avoid an error message. The higher the number of parameters and summary statistics to be estimated, the larger the value of n must be. Overall, to determine an optimal value for n, begin with small values (e.g., n=100) and gradually increase, evaluating both computation time and the quality of the estimates. Once a balance between computation time and estimation quality is achieved, select the value of n that offers the best compromise. This value will depend on the specific requirements of your ABC regression analysis, including model complexity and available computational resources.

So, we generate an n-size sample of parameter values drawn from the prior distribution. This sample is transformed into a matrix of the simulated parameter values. This matrix contains n rows, where each row corresponds to a parameter set and each column corresponds to a parameter if the parameter is multi-dimensional.

```
#-------------------------------------------------------------------------
#Function to simulate the parameters set par.sim with values drawn from the pr
ior distributions of each parameter
#-------------------------------------------------------------------------------

# Function to Generate a matrix of size (sample size 'n", number of parameters
'k')
#with values drawn from the prior distributions of each parameter
#(e.g., uniform distribution, exponential distribution, gamma distribution).

parameter.sim <- function(n,k){
  # Generate the first parameter from a uniform distribution
  parameter_1 <- runif(n = n, min = a, max = b)
  # Generate the second parameter from an exponential distribution
  parameter_2 <- rexp(n = n, rate = lambda)
  ..... # Add more parameters as needed
  # Generate the k^th parameter from a gamma distribution
  parameter_k <- rgamma(n = n, shape = alpha, rate = beta)
  # Group parameters into a matrix with k columns (if k parameters to estimate)
  par.sim <- matrix( c(parameter_1, parameter_2,..., parameter_k),  ncol = k)
  # Return the matrix of parameter samples

return(par.sim)
}
```

Then, for each parameter set, a simulated dataset is generated using the model "`toy.model`", to form a simulated dataset of size n. Summary statistics are calculated for each simulated dataset. These are transformed into a matrix called "`stat.sim`", where n rows correspond to n simulation of the summary statistics using `par.sim` and each column corresponds to one summary statistic.

```
#-------------------------------------------------------------------------
#Function to simulate summary statistics "stat.sim" from parameter samples p
ar.sim
#-------------------------------------------------------------------------

statistic.sim <- function(par.sim) {
  # Initialize an empty matrix for storing summary statistics
  stat.sim <- NULL
  # Iterate over each row of parameter samples
  for (i in 1:nrow(par.sim)) {
    # Call the toy.model function to simulate summary statistics for  each p
arameter set
    stat <- toy.model(par.sim[i,])
    # Append the summary statistics to the matrix
    stat.sim <- rbind(stat.sim, stat)
  }

  # Name the columns of the summary statistics matrix
  name.col <- NULL
  for (i in 1:NumberOfSummaryStatistic) {
    tmp <- paste0("stat", i)
    name.col <- c(name.col, tmp)
  }
  colnames(stat.sim) <- name.col

  # Return the matrix of summary statistic samples
  return(stat.sim)
}

# Function execution (don't forget to set the values of n and k)
stat.sim <- statistic.sim (par.sim)
write.csv(stat.sim, paste0(Path.data,"statsim.csv"), row.names=FALSE, quote=
FALSE)
```

Furthermore, for the observed data, we calculated the corresponding summary statistics, called "`stat.obs`", a vector whose length is equal to the number of summary statistics l.

```
#-------------------------------------------------------------------
#Function to calculate the summary statistics of the observed data
#-------------------------------------------------------------------
statistic.obs <- function(dataObserved) {
# Calculate summary statistics from observed data
# Return a vector of summary statistics observed
  return(stat.obs)
}

# Function execution (don't forget to set the values of n and k)
stat.obs <- statistic.obs (dataObserved)
write.csv(stat.obs, paste0(Path.data,"statobs.csv"), row.names=FALSE, quote=
FALSE)
```

Verification of data structure

Check that: a) the length of $stat.obs$ is equal to the number of columns in the matrix $stat.obs$; b) the number of rows in the matrix $stat.obs$ is equal to the number of rows in the matrix $par.sim;$ *c)* make sure all values are numeric; d) the structure is as follows :

```
#-------------------------------------------------------------------
# Verification of data structure
#-------------------------------------------------------------------
stat.sim <- read.csv(paste0(Path.data,"statsim.csv"))
par.sim <- read.csv(paste0(Path.data,"parsim.csv"))
stat.obs <- read.csv(paste0(Path.data,"statobs.csv"))

str(stat.obs)
# int [1:l] 0 1 9 10 12 12 10 13 15 17 ...
str(par.sim)
 # num [1:n,1:k]1.55 2.47 3.36 4.02 1.54 …
 # - attr(*,"dimnames")=List of k
 # ..$: NULL
 #  ..$: chr [1:k] "parameter_1" "parameter_2" .. "parameter_k"
 str(stat.sim)
# num [1:n,1:l]0 0 0 0 0 0 0 0 0 0 …
# - attr(*,"dimnames")=List of k
#  ..$: NULL
# ..$: chr [1:l] "stat0" "stat1" ... "statl"
```

Choice of the method

The "abc" module offers a variety of four algorithms specific to ABC, namely "`rejection`", "`loclinear`", "`neurralnet`" and "`ridge`", each of which provides a different approach to estimate the posterior distribution of model parameters. The "rejection" method is the classic ABC approach, directly identifying accepted parameters based on their proximity to the observed data. On the other hand, `loclinear` uses the flexibility of local linear regression to adjust the parameters accepted by the `rejection` method, allowing more accurate modeling of the relationships between model parameters and observations. Similarly, `neurralnet` takes advantage of the ability of neural networks to capture nonlinear structures in the data. Finally, `ridge` proposes a regularization method to reduce the sensitivity to perturbations in the data.

The peculiarity of the `loclinear` method chosen in our case lies in its use of local linear regression, which better adapts to local fluctuations in the data and allows to capture complex, non-linear relationships between model parameters and observations. This approach is particularly relevant in situations where the relationships between parameters and summary statistics are non-linear or have complex structures. By using the `loclinear` method, it is possible to obtain more precise estimates of the posterior distribution, thus promoting a better understanding of the uncertainties associated with model parameters.

As mentioned above, the "abc" package includes methods for estimating the posterior, so you need to specify which method you want to use in the `abc()` function. So, in our case the `method` parameter of the `abc()` function is set to `loclinear` (`method` = "`loclinear`")

Choice of the tolerance value

Tolerance is the proportion of accepted parameters that are closest to the target values. Its choice is very important, because the lower its value, the better our estimates will be, since they will be closer to the target value. However, there is a bias-variance trade-off: decreasing the tolerance value requires more simulations to reduce the variance of the regression fit, but also increases the bias due to uncorrected deviations from additivity and linearity. What is suggested in this case is to vary the tolerance and data size, simulate and compare the results, for example by calculating the relative root mean square error, and take the case that minimizes it. This value is defined by assigning a positive value between 0 and 1, representing a percentage, to the "`tol`" parameter of the `abc()` function. For example, `tol=0.1` indicates that we accept `10%` of the best generated parameters, those that give simulated statistics very close to the observed statistics.

Transformation of the parameter before and after the estimation

An optional step can be the transformation of the parameters before the estimation and the return to their original scale after regression. The `transf` parameter of the `abc()` function dictates these transformations through a string vector input, specifying the type of transformation applied to the parameter values. Options include "`none`" for no transformation, "`log`" for logarithmic transformation, and "`logit`" for logit transformation. For example, if you have three parameters to estimate, you can choose to leave the first parameter untouched, apply a logarithmic transformation to the second, and a logit transformation to the third, resulting in `transf= c("none", "log", "logit")` within the `abc()` function. Note that the logit transformation is tailored for probability modeling in logistic regression, while the logarithmic transformation is versatile, serving purposes such as variance stabilization and data distribution improvement.

Inference execution

Once the parameters and their respective settings have been determined (model, simulated parameters, simulated and observed summary statistics, methods, any transformations, tolerance level), the `abc()` function is called with these arguments, starting the estimation process. Upon completion, the function returns the estimated parameters, adjusted to their original scale if transformations have been applied, allowing the user to interpret and use the results effectively. Going back to our previous example, we can run the estimate in the following way:

```
#------------------------------------------------------------------
# Using the abc function to perform ABC-regression analysis
#------------------------------------------------------------------
lin <- abc(
  # The target value to be achieved in the ABC analysis, typically the actual
observed statistics
  target = stat.obs,
  # The parameter samples generated by simulation
  param = par.sim,
  # The simulated summary statistics derived from the parameter samples
  sumstat = stat.sim,
  # The specified tolerance for accepting simulations based on their proximity
to the target value e.g 0.1(to modify)
  tol = 0.1,
  # Specifies whether rejection correction for homogeneity will be applied
  hcorr = FALSE,
  # The method used for adjustment of parameters after particles acceptance an
d rejection

 method = "loclinear",
  # The transformations to apply to the data before ABC analysis (in this exam
ple, "none" and "log" transformations are suggested for two parameters to be e
stimates)
  transf = c("none", "log")
)
```

Access to results

The variable "`lin`", which stored the results obtained, is an object of the class "`abc`". This object contains several components, the most important of which are the `adj.values`. These values represent the adjustments made by the regression and correspond to the particles accepted in the ABC regression process. For more details on the different components of this object, please refer to the documentation of the package used [10]. The package includes functions that allow you to synthesize information from accepted particles (e.g., `summary()`), visualize results (e.g., `hist()`), and assess the quality of results (via plots such as the density of the anterior and posterior distribution, a scatter plot of Euclidean distances as a function of parameter values, as well as a normal Q-Q plot of regression residuals) with the `plot()` function. However, you can also extract and store fitted values using the following syntax:

```
posterior <- lin1$adj.values
posterior <- as.data.frame(posterior)
```

This allows one to organize, represent, and conduct statistical analyses according to our specific needs.

Alert point

When using the Monte Carlo Acceptance-Rejection method followed by regression to fit a posterior distribution, it is possible for some posterior values to fall outside the bounds specified by a uniform prior. This poses a problem because such values are theoretically unacceptable within the strict Bayesian framework. Practical and theoretical solutions have been proposed, for example (i) rejecting out-of-bounds values, but this can reduce efficiency and potentially bias the posterior estimate if a significant proportion of samples are rejected; (ii) re-evaluating the prior bounds, possibly extending the uniform prior bounds based on prior knowledge or observed data, but this requires rerunning simulations, which can be costly; (iii) using transformations (e.g., logit) but this approach can complicate the interpretation of the parameters and may require careful handling to ensure the transformed parameters follow the desired prior distribution. The choice of a solution often depends on the application context and the available data.

Using the "BRREWABC" package

Installation of package

Install the "BRREWABC" package, and load the package into the R environment using the following command:

```
install.packages("devtools")
library(devtools)
devtools::install_github("GaelBn/BRREWABC")
library (BRREWABC)
# to access package documentation
help(package = "BRREWABC")
```

Model definition

The definition of the model is made in the same way as it is explained above. As a reminder, it is a function which corresponds to the mechanistic model to be used, that takes a list of parameters named `parameters_vector` as input and returns simulated summary statistics `stat.sim`.

```
#-----------------------------------------------------------------
#  Model definition 'toy.model'
#-----------------------------------------------------------------
toy.model <- function(parameter_vector)
{
  ... # Perform a series of tasks based on the study
  ... # Calculate summary statistics 'stat.sim'

 return (stat.sim)
}
```

Compute observed summary statistics

Defining the observed summary statistics is done in the same way as it is explained above. From the observation data, we calculate the statistics that summarize it.

```
#-----------------------------------------------------------------
#Function to calculate the summary statistics of the observed data
#-----------------------------------------------------------------
statistic.obs <- function(dataObserved) {
# Calculate summary statistics from observed data
# Return a vector of summary statistics observed
  return(stat.obs)
}

# Function execution (don't forget to set the values of n and k)
stat.obs <- statistic.obs (dataObserved)
write.csv(stat.obs, paste0(Path.data,"statobs.csv"), row.names=FALSE, quote=
FALSE)
```

Distance between simulated and observed summary statistics

This section covers the definition of the distance metric used to compare the simulated summary statistics with the observed ones. This task is performed by a function called `compute.dist`, which takes as input a particle (set of parameter values) and the observed summary statistics (`parameters_vector, stat.obs`). Within `compute.dist()`, `toy.model()` is run to generate simulated summary statistics `stat.sim`, followed by the computation of the distance between the output of `toy.model()` and the observed summary statistics. The choice of a distance metric (example: an Euclidean distance, often used because of its simplicity and intuitiveness, other more complex distances can be used to take into account specific characteristics of the data) is at the discretion of the user. We then define the `model_list` object, containing the model to be considered during the inference procedure.

```
#------------------------------------------------------------------
#function to calculate the distance distance between stat.sim and stat.obs
#------------------------------------------------------------------

compute.dist <- function (parameters_vector, stat.obs){
 stat.sim <- toy.model(parameters_vector)
 # Write the steps to calculate the distance between stat.sim and stat.obs
 # by showing what corresponds dist
dist <- distance between stat.obs and stat.sim

return (c(dist))
}
model_list <- list("m1" = compute_dist)
```

Define prior distribution

The same procedure as described above is used to select the prior distribution.

```
#------------------------------------------------------------------
# Define prior distribution
#------------------------------------------------------------------

prior.dist <- list("m1" = list(
                              # uniform distribution for parameter_1
                               c("parameter_1", "unif", a, b),
                              # exponential distribution for parameter_2
                               c("parameter_2", "exp", rate),
                               ...
                              # gamma distribution for parameter_k
                               c("parameter_k", "gamma", shape, rate)))
```

Running the inference procedure

```
#------------------------------------------------------------------
# Run abc smc procedure
#------------------------------------------------------------------

res <- abcsmc(model_list = toy.model,
              prior_dist = prior.dist,
              ss_obs = stat.obs,
              max_number_of_gen = 20,
              nb_acc_prtcl_per_gen = 500,
              new_threshold_quantile = 0.8,
              verbose = FALSE)
```

Once all the elements have been defined, parameter estimation can be launched. However, when using the ABC-SMC algorithm, there are several critical points of vigilance to ensure the quality of the estimation. Key parameters that need careful consideration include the number of iterations of the algorithm (max_number_of_gen), the number of particles to accept at each iteration (nb_acc_prtcl_per_gen), and the reduction of the threshold used for accepting particles (new_threshold_quantile). Refer to the section 1 “Critical Points to Watch in ABC-SMC for more details.

Access and plot results

The resulting object, `res`, is a list containing two data: the first data includes the thresholds used in each iteration to compare the observed summary statistics with the simulated summary statistics. The second data contains the particles accepted in each iteration.

To visually verify convergence, you can access the different thresholds with `res$thresholds` and then plot them. Additionally, you can access the posterior distributions of each iteration with `res$particles` and generate the density plots of the desired distributions.

```
#-----------------------------------------------------------------------
# Access results
#-----------------------------------------------------------------------
all_accepted_particles <- res$particles
all_thresholds <- res$thresholds


#-----------------------------------------------------------------------
# plot results
#-----------------------------------------------------------------------

# Plot the density of the distribution of the last accepted particles
plot_abcsmc_res(data = all_accepted_particles, prior = prior_dist,
        filename = "smpl/res/figs/smpl_pairplot_all.png", colorpal = "YlGnBu")


# Plot the density ridges of the accepted particles for each iteration
plot_densityridges(data = all_accepted_particles, prior = prior_dist,
        filename = "smpl/res/figs/smpl_densityridges.png", colorpal ="YlGnBu")


# Plot the evolution of the thresholds used in the algorithm, this shows how th
e threshold values decrease  over iterations of the algorithm.
plot_thresholds(data = all_thresholds, nb_threshold = 1,
          filename = "smpl/res/figs/smpl_thresholds.png", colorpal = "YlGnBu")
```

Critical Points to Watch in ABC-SMC

Number of Iterations

The number of iterations in the ABC-SMC algorithm significantly impacts the accuracy and convergence of the posterior distribution. Each iteration refines the set of accepted particles, ideally moving closer to the true posterior distribution. Too few iterations may lead to poor approximation and a posterior that fails to converge. Conversely, too many iterations can be computationally expensive without substantial gains in accuracy beyond a certain point. It is essential to balance the computational cost with the desired accuracy, often determined through preliminary experiments or domain-specific knowledge.

Number of Particles

The number of particles accepted at each iteration determines the sample size for the next round of simulations. A larger number of particles can provide a more accurate representation of the posterior distribution, reducing the variance and improving the estimation. However, this also increases the computational load. A smaller number of particles might speed up computations but at the risk of higher variance and a less accurate posterior. The choice of this parameter often depends on the complexity of the model and the available computational resources.

Reduction of the Threshold

The threshold used for accepting particles is a critical parameter in ABC-SMC. This threshold, often defined by a quantile of the distances of accepted particles, controls how similar the simulated data must be to the observed data. As the iterations proceed, this threshold should be gradually reduced to refine the approximation of the posterior distribution. If the threshold is reduced too quickly, the algorithm may reject too many particles, leading to insufficient samples and potential convergence issues. On the other hand, if the

threshold is reduced too slowly, the algorithm may not sufficiently refine the posterior distribution, leading to a broader and less accurate estimation.

Effects on Estimation Quality

1. Iteration Count: Insufficient iterations may result in a posterior distribution that is not well-converged, leading to biased estimates. Excessive iterations, while improving accuracy, can be computationally prohibitive.
2. Number of Particles: Too few particles increase the risk of a high-variance posterior with poor accuracy. Sufficient particles are crucial for a stable and accurate estimation but require significant computational resources.
3. Threshold Reduction: Improper reduction rates can either lead to premature convergence with poor accuracy or extended computation times with diminishing returns on accuracy. Optimal reduction strategies balance the precision of acceptance with computational efficiency.

A careful tuning of the number of iterations (`max_number_of_gen`), the number of particles (`nb_acc_prtcl_per_gen`), and the threshold reduction (`new_threshold_quantile`) is essential for the effective application of the ABC-SMC algorithm. These parameters collectively influence the balance between computational efficiency and the quality of the posterior estimation. Experimentation and domain knowledge play a crucial role in finding the optimal settings for a given problem. Although the effects described above depend on the application context (model and summary statistics used), the user can refer to the package documentation [11] for some examples.

SECTION 2 - ILLUSTRATIVE CASES

Foreword

In this section, we are illustrating the use of the previously defined inference packages on specific case studies. We chose bovine respiratory diseases (BRD) as the illustrative case, as it is one of the major enzootic diseases affecting young cattle in Europe. This disease leads to significant economic losses, animal welfare problems and extensive use of antimicrobials. To better understand and anticipate BRD occurrence in young cattle populations under different conditions (farming practices and biosecurity, as well as detection methods and treatments), INRAE previously developed a stochastic epidemiological model [12] using the EMULSION framework [13]. This model is adapted to three different pathogens (bovine respiratory syncytial virus (BRSV), *Mannheimia haemolytica*, *Mycoplasmas bovis*), which are the best representatives of the three main groups of pathogens frequently involved in BRD.

To date, the model parameters have been calibrated for each of the pathogens using published knowledge on BRD. To increase the robustness, realism and accuracy of the model, and thus its usefulness for end-users (farmers, vets), model parameter values and initial conditions could be inferred from observed data, where available. To do so, however, inference algorithms should be used, a task often hard for non-modelers. In the previous section, we introduced two of such inference algorithms, and explained their functioning. Here, we use them on synthetic data generated with the INRAE BRD model in various situations to explain the type of results that can be obtained and how these results should be interpreted and used.

The INRAE pathogen-specific BRD model has from 8 to 13 epidemiological parameters depending on the pathogen considered (we focused here on BRSV and *M. haemolytica*). While most parameters are well defined (Table 1), the transmission rate, the basic reproduction number R_0 in a naïve population, and related parameters (reduction factors) remain poorly quantified. For more details on the model, see the APPENDIX section, extracted in full and without modification from B. Sorin-Dupont et al. (2023) [12] as they describe processes and transitions of the model.

Table 1. Model parameters and their certainty levels according to available information in the literature and DECIDE experts' opinion (* Low confidence, ** Medium confidence, *** High confidence)

Parameters	Pathogens	Confidence
R_0 in a naïve population	BRSV	*
Reduction factor of susceptibility for a second infection	BRSV	**
Reduction factor of infectiousness for re-infected individuals	BRSV	*
R_0 in an endemic situation	BRSV	***
Transmission rate (/h)	BRSV & *M. haemolytica*	*
Spontaneous shedding probability	BRSV & *M. haemolytica*	***
Initial proportion of partially immune animals	BRSV	***
Initial proportion carrier animals	BRSV & *M. haemolytica*	**
Clinical sign duration	BRSV & *M. haemolytica*	***
Infectious period duration	BRSV & *M. haemolytica*	***
Asymptomatic period duration	BRSV & *M. haemolytica*	***
Probability of successful treatment	BRSV & *M. haemolytica*	***
Probability of severe forms	BRSV & *M. haemolytica*	***

The ability to estimate all parameters of a model depends on several factors, including model complexity, data availability, and technical limitations of estimation techniques. Some models with many interdependent parameters may require extensive data sets and advanced optimization techniques, which can be challenging to implement. Certain parameters that are sensitive to data fluctuations or specific initial conditions may be difficult to estimate, although they represent a small fraction of the total parameters, they significantly influence model results. Non-identifiability problems can arise when multiple parameter combinations yield similar results, making it impossible to uniquely determine parameter values from observed data. Therefore, it may be necessary to select a limited set of parameters to be estimated. For this, a sensitivity analysis can also help to select the most influential parameters that would also be the most uncertain.

Here, the biological system is highly variable, as farms are highly structured systems, with animals raised in small groups, and as the initial conditions could be any as possible (from no infected animal at start to a large proportion of them). As a result, the model representing this system is highly stochastic. In addition, some parameters interact. For example, the two following situations might generate similar observations: (1) a high initial proportion of highly susceptible animals exposed to a lowly virulent pathogen; (2) a low initial proportion of highly susceptible animals exposed to a highly virulent pathogen. This clearly indicates that estimating simultaneously the initial proportion of highly susceptible animals and the transmission rate (closely related to pathogen virulence) might be very challenging, while these two parameters are often the main unknown ones.

Despite this challenge, we focused in our examples the parameter inference to these most unknown parameters, i.e. related to transmission and initial conditions: the R_0 and the initial proportion of partially immune animals for BRSV (respectively named: `R0_naive` and `prop_lowrisk`); the transmission rate and the

initial proportion of non-carrier animals for *M. haemolytica* (respectively named: `transmission_lowrisk` and `prop_lowrisk`). All other parameters were assumed to be sufficiently known and their values were fixed.

Instead of using observed data to infer parameter values, we used here **synthetic data**, as it enables us to have a **perfect knowledge of the process that generated data** and thus helped explain possibilities and limits in result interpretation and usage. Using such synthetic data allows a precise control over inference conditions, including the knowledge of 'real' parameter values and relationships between variables, facilitating the exploration of different scenarios and conditions. Using synthetic data makes it possible to validate the performance of the inference methods, thereby increasing confidence in their ability to handle real-world data.

However, we want to draw the reader's attention to the fact that using synthetic data generated by simulation has its limitations when it comes to testing an inference algorithm. Thus, not recovering the exact values used to simulate the synthetic data does not necessarily constitute an issue. Indeed, the main objective of inference is to determine the best sets of parameters that allow reproducing the observed trajectory (obtaining a good fit between observations and simulation), rather than retrieving the precise values used for the simulation. This distinction is particularly important in the context of stochastic models. Stochasticity introduces variability that can result in different sets of parameters generating similar trajectories. Therefore, a good inference algorithm should be able to recognize this variability and provide parameter estimates that are consistent with the observed trajectories, even if these parameters do not exactly match those used for the simulation of the synthetic observed data set.

We explored five scenarios involving two pathogens (BRSV; *M. haemolytica*), two management strategies for young cattle herds (a single batch of 50 animals; five batches of 10 animals each), and two different observation outcomes (animals detected as infected; truly infected animals). In each batch, animals are classified into two risk levels (low and high) of contracting the disease. This classification determines the likelihood of infection and pathogen transmission, in relation with the animal stress level and susceptibility. An animal is considered sick if it has an elevated temperature or clinical signs (mild or severe, such as destress or anorexia). It is important to note that an animal may contract the disease several times and may be sick without being detected. Once detected, sick animals are treated individually, avoiding re-treatment of animals that have already received the maximum number of treatments allowed. Animals are monitored for 30 days with a time step of 12 hours, corresponding to farmers' observation intervals. At each time step, risk status, hyperthermia, health status (infected or not), clinical signs, detection (detected or not), and treatment are monitored.

For these results and all those that follow: In each case, the parameters used to simulate the synthetic observations are included in the posterior distributions (distributions of particles accepted).

EXAMPLE 1: Large batch and complete observed data (summary statistics: all infected animals)

This example illustrates a situation where biological variability plays as minor a role as possible. We focused on a large unique batch of 50 animals in which BRSV is circulating. Also, we assumed to have access to the best observed data as possible, i.e., all infected animals were assumed to be detected with a perfect sensitivity and specificity every 12 hours (summary statistics for the observed and the simulated datasets). We applied the two inference procedures (the ABC-regression and the ABC-SMC) to estimate the value of the R_0 in a naïve population ($R0\_naive$) and the initial proportion of partially immune animals in the batch ($prop\_lowrisk$) to compare the estimators obtained using these two processes. More precisely, we aimed to estimate the values of these parameters that best represent the trajectory.

To define the synthetic observations of infected cases over time, we first simulated 1,000 model trajectories with similar parameter values ($R0\_naive = 3.1$ and $prop\_lowrisk = 0.4$). We then calculated the median of these trajectories. From the 1000 simulated trajectories, we finally selected as the observed (*synthetic*) data the simulated trajectory with the shortest distance to the median (trajectory near median; Figure 2). We thus chose here an observed (`stat.obs`) situation well represented by the BRD model.

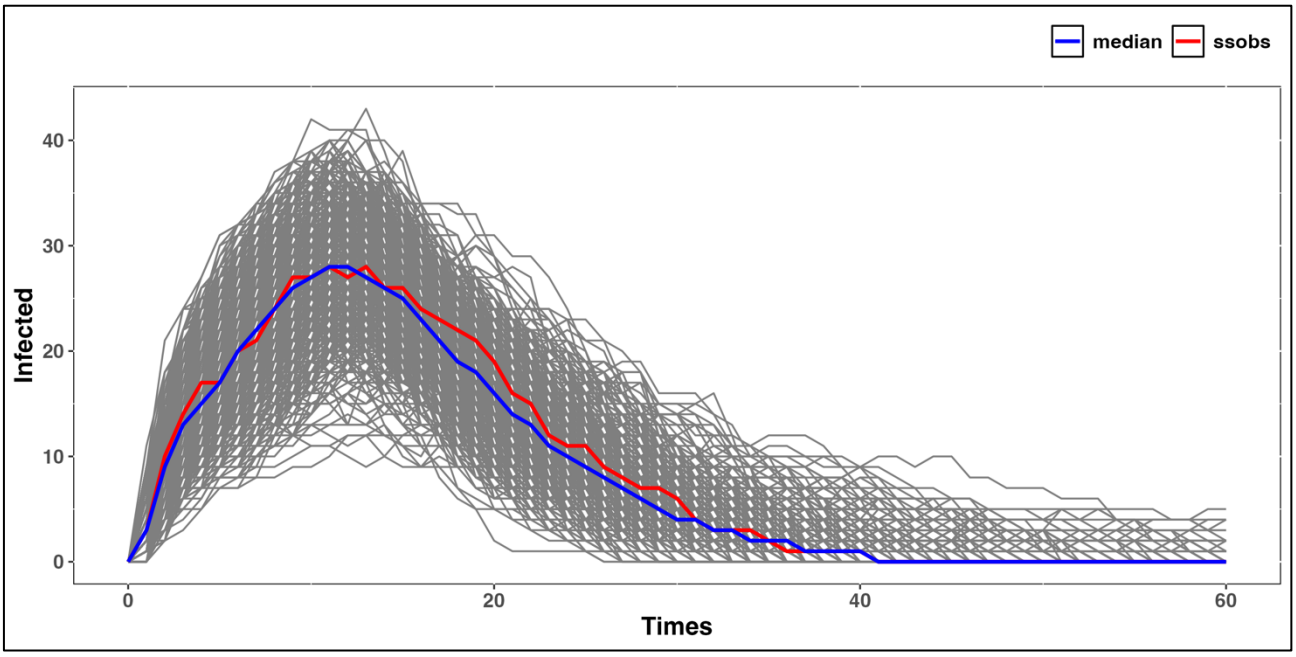

Figure 2. **Scenario One Batch - All Infected - BRSV**: Visualization of the 1000 simulated trajectories (gray) with the target parameters (R_0_naive = 3.1 and prop_lowrisk = 0.4); of the median derived from the ensemble of these 1000 trajectories (blue); of the synthetic data (simulated trajectory the closest as possible to the median) considered as the observed summary statistics (red).

Using the "BRREWABC" package

The package "BRREWABC" was applied with the following specificities:

- a number of particles to be accepted equal to `3000`;
- a squared Euclidean distance (the sum of squares of the differences between the simulated and observed summary statistics);

- a minimum acceptance rate equal to 0.001;
- a maximum number of iterations equal to 20;
- a maximum number of simulations per iteration of 300,000 (a stopping criterion to ensure each iteration completes within a reasonable time, avoiding excessive overruns).

The results are presented in Figure 3 in the left. We initially assumed uniform prior distributions, on the range [0, 5] for R_0_naive and [0, 1] for prop_lowrisk, respectively.

Over iterations, the posterior distributions become increasingly precise, (cf. Figure 3A, D), providing a clear picture of the density of the estimates, especially for prop_lowrisk. The concentration of the density indicates a convergence of the algorithm towards specific values, suggesting stability and robustness in the estimates. The distributions of the last accepted particles show a peak around 3.132 for R_0_naive and a well-defined peak at 0.404 for prop_lowrisk indicating accurate parameter estimate. However, despite this convergence, the 90% credible intervals for both parameters remain very wide for R_0_naive and to a lower extent for prop_lowrisk (Table 2). This wide range of values suggests significant uncertainty in the parameters needed to reproduce the observed trajectory, due to the strong stochasticity of the model.

Observing the significant reduction in the tolerance margin with each iteration (Figure 3E), and the fact that most of the tolerances are small, we can see that most of the simulations are very close to the observed data. This demonstrates the ability of the algorithm to efficiently converge to simulations that are most representative of the observed data.

The simulations obtained from the last accepted particles closely follow the observed curve, demonstrating that the estimated parameters provide a faithful model of the observed epidemiologic dynamics (Figure 3B). This fit of the simulations to the observed data reinforces the validity of the estimated parameters. In addition, the posterior trajectories near the 10th and 90th quantiles also adequately enclose the observed trajectory (cf. Figure 3C).

Using the “abc” package

The "abc" package was applied with the following characteristics:

- a set of particles simulated par.sim of size 99999;
- a set of summary statistics simulated using the BRD model stat.sim of size 99999;
- a tolerance tol=0.005, meaning that we accept 0.5% of 99999 particles, either 500 particles;
- a Euclidean distance between simulated and observed summary statistics, as used in the abc() function;
- a set of non-transformed parameters before and after estimation (transf=c("none", "none"));
- a choice of the "loclinear" method.

The results are presented in Figure 3 in the right.

In this case, to manage the out-of-bounds values of the uniform prior, we chose to reject these values, since the proportion of samples requiring rejection was not significant.

Compared to the "BRREWABC" package, the posterior distributions show similar peaks, but with slightly different shapes, which may reflect differences in the Bayesian approximation approaches used by the two processes (Figure 3F). Specifically, for the parameters R0_naive and prop_lowrisk, whose initial values

were 3.1 and 0.4, respectively, the posterior distributions of accepted particles obtained with the "abc" package have modes of 3.1174 (vs. 3.073 with the BRREWABC package) and 0.410 (vs. 0.400 with the BRREWABC package). The 90% credible intervals are also slightly wider for `R0_naive` and `prop_lowrisk`, respectively (Table 2).

As with the "BRREWABC" package, the posterior trajectories simulated from the accepted particles closely match the observed trajectory (cf. Figure 3G). However, the posterior trajectories near the 10th and 90th percentiles enclose the observed trajectory slightly less satisfactorily than with the "BRREWABC" package (cf. Figure 3H).

In conclusion, in this example, both inference approaches, in "abc" and "BRREWABC" (ABC SMC) packages, are effective in estimating the parameters of the epidemiological model. Given that each method has its own characteristics and nuances in terms of precision and variability of estimates, the choice between the two may depend on the specificities of the study, the need for precision or the representation of uncertainty. The ABC SMC method, with its low tolerances and its ability to generate simulations close to observations, offers a particular robustness that could be advantageous for certain epidemiological applications.

To compare the computation time of each package, we will use the required number of model executions, as the computation time strongly depends on the model execution time. Thus, we required approximately 100,000 executions for the "abc" package and 480,408 for the "BRREWABC" (ABC SMC) package.

Using «BRREWABC» package

Using «abc» package

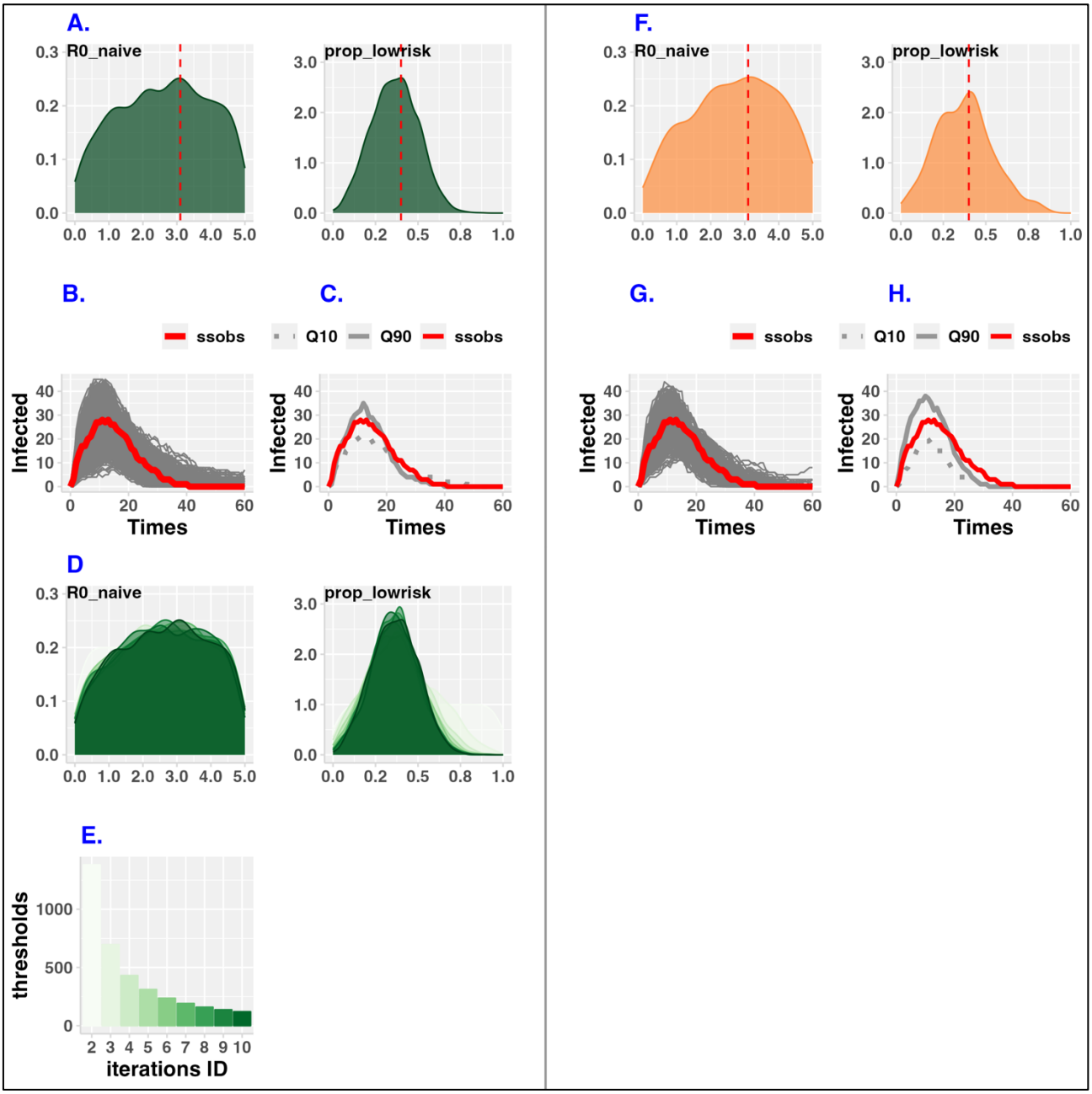

Figure 3. **Scenario One Batch - All Infected - BRSV**: Results using the "abc" (right) and "BRREWABC" (left) packages. A-D (resp. F): density of particles accepted at each iteration and density of the last 3,000 (resp. 500) particles accepted, respectively, which form the posterior of parameters R0_naive and prop_lowrisk; the vertical line corresponds to the target parameter value. E: Reduction in tolerance margin over iterations. B (resp. G): Trajectories obtained from the 3000 (resp. 500) accepted particles (gray) and observed trajectory (red). C (resp. H): quantile trajectories (gray; Q10 in dashed line, Q90 in solid line) obtained from the accepted particles and observed trajectory (red).

EXAMPLE 2: Influence of data degradation (summary statistics: all animals detected as infected)

Unlike the previous example where all and only infected animals were considered to be observed every 12 hours, here we assumed only detected animals were observed every 12 hours. As detection was associated to hyperthermia and an on-farm visual appraisal of clinical signs (assuming lethargy is the most significant and mild clinical signs can be detected but with a smaller sensitivity than severe clinical signs), false positives were possible, as well as infected animals remaining undetected (thus unobserved).

As all scenario information remains the same as in the previous example, we used the same observed trajectory (Figure 4, orange line), but using detected cases instead of infected ones (Figure 2, red line) as the summary statistics.

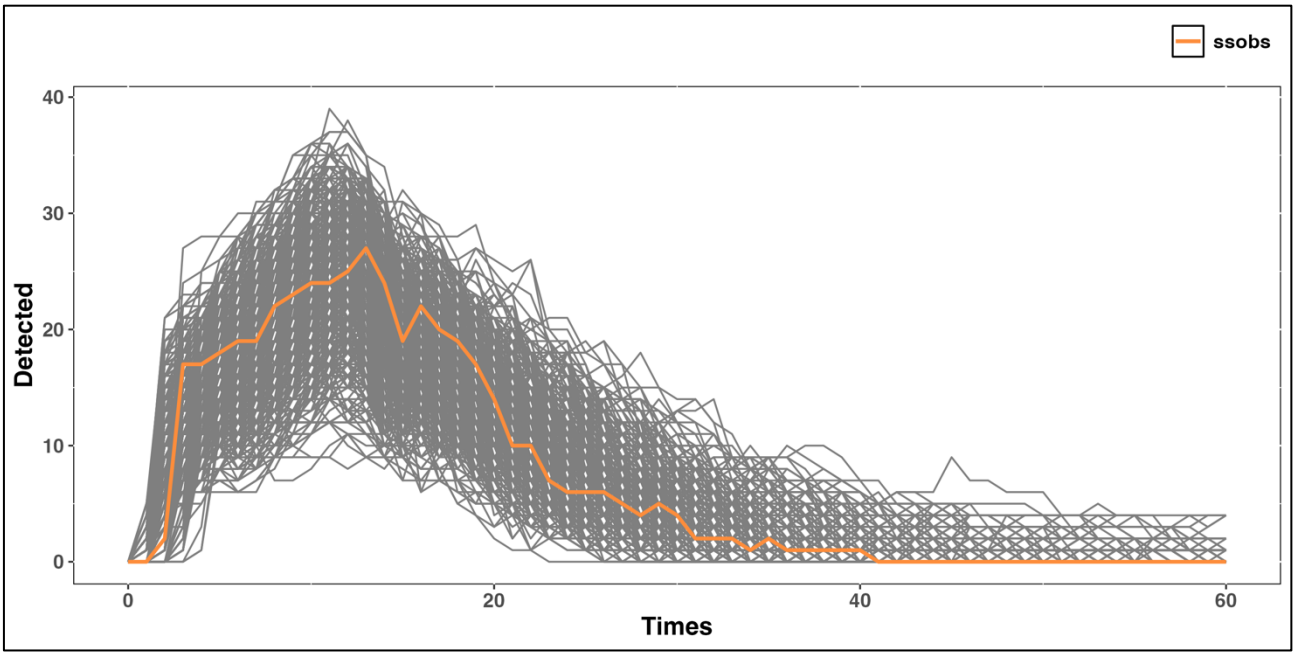

Figure 4. **Scenario One Batch - Detected - BRSV**: Visualization of the 1000 simulated trajectories (gray) with the target parameters (R_0_naive = 3.1 and prop_lowrisk = 0.4); of the synthetic trajectory as the observed summary statistics representing detected cases (orange).

The simplified information, which considers only detected individuals as observed data at each time step, results in less accurate results for both packages compared to those obtained in Example 1, where the observed data included all infected individuals, with no issue in detection sensitivity and specificity.

Using the "BRREWABC" package

The package "BRREWABC" was applied with the same parameters as before, e.g. distance, acceptance rate, number of particles accepted, prior etc. Results are presented Figure 5 on the left.

The posterior distributions of the parameter `R0_naive` show a bimodal shape (Figure 5 A, D), possibly due to an insufficient number of iterations to achieve stabilization. Improving this limitation (performing more iterations) would require considerably more time, as the method has already converged to an acceptable level (Figure 5E). This observation can also be explained by our stochastic model, which is able to capture

substantial variations in the parameters influencing disease transmission. In addition, the posterior distribution of `prop_lowrisk` shows a well-defined peak with a mode at 0.3180318, which is lower than the value initially used (Table 2). However, these results do not affect the efficiency of the procedure, as all simulated posterior trajectories using the latest accepted particles closely surround the observed trajectory (Figure 5B, C).

Compared to Example 1, where the observed trajectory included all infected cases, the posterior distributions of the parameters differ significantly (Kolmogorov-Smirnov test, p-value = 3.48 e-10 for `R0_naive` and p-value = 1.487e-06 for prop_lowrisk) when only the detected cases are considered. As expected, in contrast to Example 1, the obtained estimates provide a range of values that differ from the values originally used to generate the observed data. These results confirm the hypothesis that the inference estimates vary considerably depending on the observed trajectories, despite the use of a similar model and identical parameterization. Thus, the trajectory used (observed data) plays a crucial role in determining the model parameters.

Using the “abc” package

The "abc" package was applied with the same parameters as before, e.g. simulated data size, tolerance, number of particles accepted, etc. Results are presented Figure 5 on the right.

In this case, to manage the out-of-bounds values of the uniform prior, we chose to reject these values, since the proportion of samples requiring rejection is not significant.

The posterior distributions of the parameters `R0_naive` and `prop_lowrisk` obtained exhibit a similar pattern to those obtained with with "BRRWABC", albeit with slight variations in distribution shapes (Figure 5A, F). Both demonstrate peaks at similar values for `R0_naive` (Table 2), although the "BRRWABC" procedure appears to have a slightly wider distribution. While simulations closely align with observed data, "BRRWABC" exhibits more visible variability (Figure 5B, G). Notably, both processes indicate that observed data generally fall within quantiles (Figure 5C, H), indicating good model-to-data correspondence.

In summary, ABC regression and ABCSMC methods yield consistent results with observed data, albeit with some variability and differences in posterior distribution shapes (comparison of the posterior distributions of `R0_naive` and `prop_lowrisk` obtained in example 1 with those obtained in example 2 using the Kolmogorov-Smirnov test: p-value = 1.998e-15 for prop_lowrisk and p-value = 7.372e-05 for `R0_naive`). ABC regression displays a more concentrated distribution around estimated parameters, while ABCSMC shows greater dispersion with mode values closer to the initial values compared to ABC regression. This suggests that ABCSMC captures more uncertainty in parameters but tends to converge towards the target value. Furthermore, comparisons between observed and simulated cases indicate the effectiveness of both methods in reproducing observed trends, albeit with differing levels of precision and variability. Compared to Example 1, we have observed the same conclusions as above with the "BRRWABC" package.

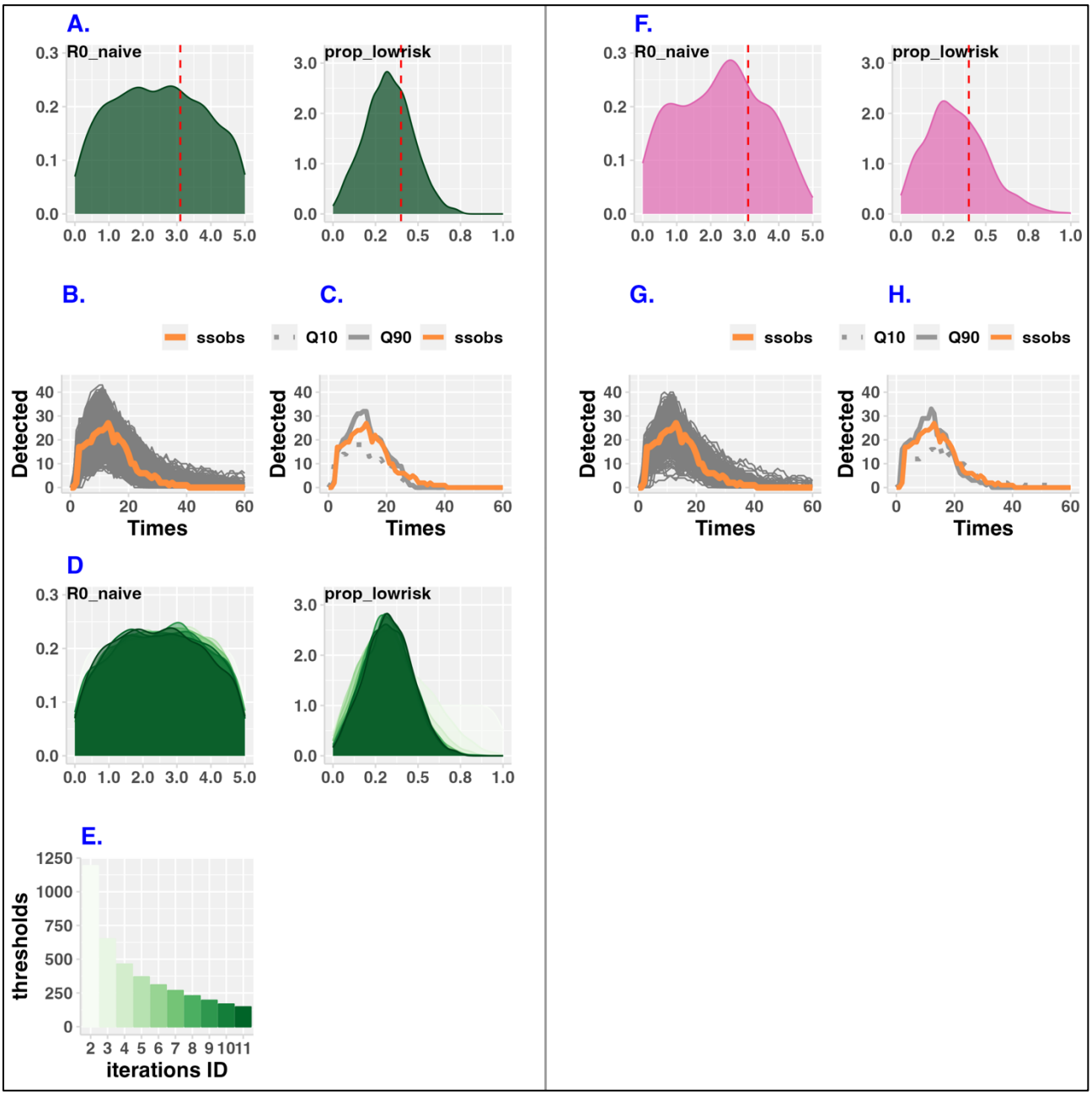

Figure 5. **Scenario One Batch - Detected - BRSV**: Results using the "abc" and "BRREWABC" Packages shown on the right and left, respectively. D. and A. (resp. F.) represent respectively the density of particles accepted at each iteration and the density of the last 3000 (resp. 500) particles accepted, which form the posterior of the parameters R0_naive and prop_lowrisk, the vertical line corresponds to the target parameter value; E. Reducing the tolerance margin between each iteration; B. (Resp. G.) Trajectories obtained from the 3000 (resp. 500) accepted particles in gray and the observed trajectory in orange; C. (Resp. H.) in gray the quantile trajectories (Q10 in dashed line and Q90 in solid line) obtained from the accepted particles and in orange the observed trajectory

EXAMPLE 3: Influence of biological stochasticity (5 batches of 10 animals each)

This example is provided to highlight that inference quality might decrease as the biological system (and thus the associated model) becomes more stochastic. We still considered here 50 animals, but distributed into 5 batches of 10 animals each. As population becomes smaller, even stochasticity increases. Summary statistics for both observed and simulated data were the number of infected animals (as in Example 1) every 12 hours. Batches were sorted in ascending order of their number of infected animals, so that we compared observed and simulated batches having a similar ranking of prevalence. As in Example 1, the observed trajectory was the closest one to the median of 1000 trajectories with identical parameter values (Figure 6). There is a slight divergence between the median and the closest trajectory on the left side of Figure 6 due to result aggregation. This trend is not apparent when the same curves are examined per batch (right side of Figure 6).

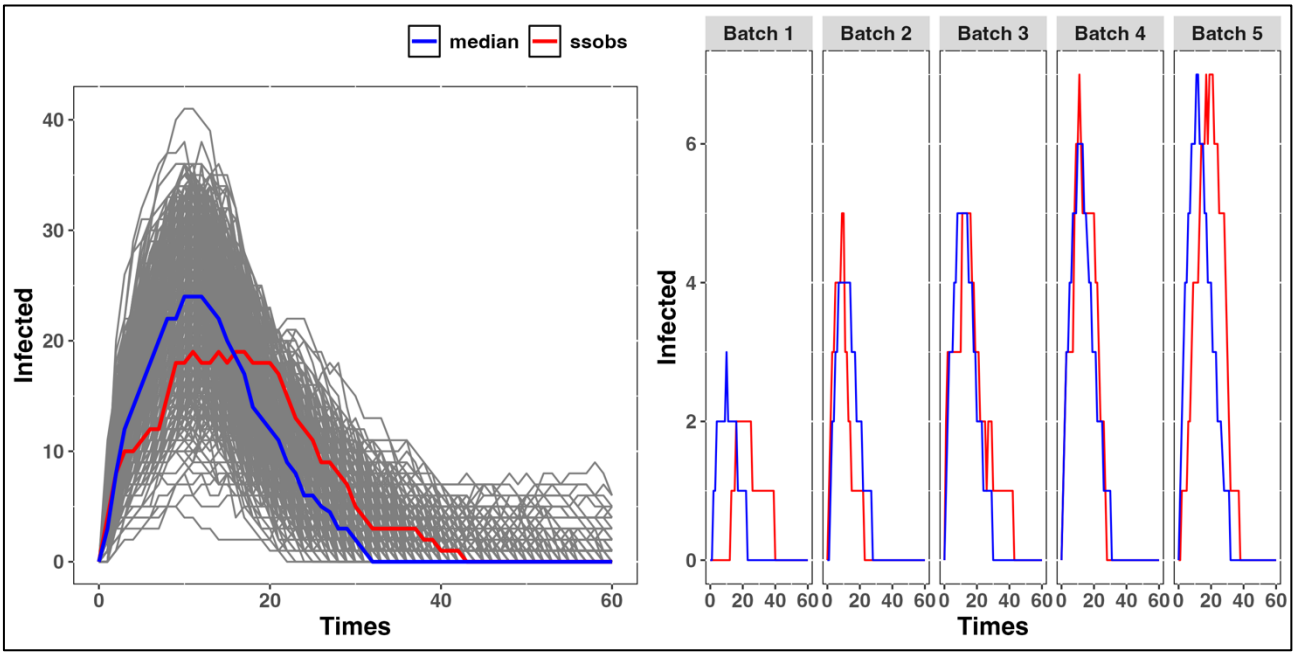

Figure 6. **Scenario Five Batches - All Infected - BRSV**: Visualization (left) of the 1,000 simulated trajectories (gray) with the target parameters (R_o_naive = 3.1 and prop_lowrisk = 0.4); of the median derived from the ensemble of these 1,000 trajectories (red); of the synthetic trajectory approximating the median as the observed summary statistic (blue). Visualization (right) of the median (blue) and the closest path to the median (red), considering the batches.

The package "BRREWABC" was applied with the same parameters as before, e.g. distance, acceptance rate, number of particles accepted, prior etc. Results are presented Figure 7 on the left.

The posterior density distributions of the parameter estimates are concentrated around certain values, indicating convergence of the algorithm to precise parameter values. Well-defined peaks in the marginal distributions indicate accurate estimation of these parameters. However, compared to the parameter values initially used to simulate the observed trajectory (`R0_naive = 3.1` and `prop_lowrisk = 0.4`), the posterior densities show an underestimation for `R0_naive` and an overestimation for `prop_lowrisk` over the iterations, as shown in Figure 7D. This trend is confirmed by the modes of the posterior densities of the last particles accepted (Figure 7A), which are 0.529 for `R0_naive` and 0.533 for `prop_lowrisk`, respectively. This observation suggests that the observed trajectory is probably not the most representative for this set of parameters.

Furthermore, the 90% credible intervals remain very wide for both parameters, especially for R0_naive (Table 2). This enormous range of values indicates a large uncertainty in the parameters needed to reproduce the observed trajectory, due to the high stochasticity of the model. However, considering the posterior density modes as representative candidates, their high intensity suggests that they may better represent the observed trajectory.

Observing the iteration tolerance across iterations (Figure 7E), a significant reduction in tolerances between the first and last iterations is observed, indicating an improvement in the fit of the simulations to the observed data over iterations. The fact that the tolerances of the last iterations are small demonstrates the ability of the algorithm to generate simulations that are very close to the observed data, indicating good convergence of the algorithm. It should be noted, however, that adjusting the stopping criteria to increase the number of iterations to have a majority of small tolerances would improve the accuracy of the results, but would require a significant computational cost.

Despite the underestimation and overestimation of the target values, the fact that the posterior trajectories near the 10th and 90th quantiles satisfactorily frame the observed trajectory (Figure 7B, C) reinforces the idea that the estimated parameters are adequate to capture the dynamics represented by this trajectory. The simulations around the quantiles show that the majority of the simulated trajectories falls within the quantile bands, illustrating a good fit to the observed data.

In conclusion, the parameter estimation results obtained for R_0_naive and prop_lowrisk using the "BRREW-ABC" package show precise convergence and close agreement with the observed data. Reduced tolerances and well-defined distributions around the modes suggest that the procedure is effective for parameter estimation of an epidemiological model. The simulations generated from the accepted particles capture the dynamics of the observed trajectory well. However, the peaks of the distributions (modes) are far from the parameters originally used to generate the data, indicating that in this example the observed trajectory is not representative of the parameter values used. This shows that we add more uncertainty to the parameters that best represent the trajectories as we add more stochasticity in the model.

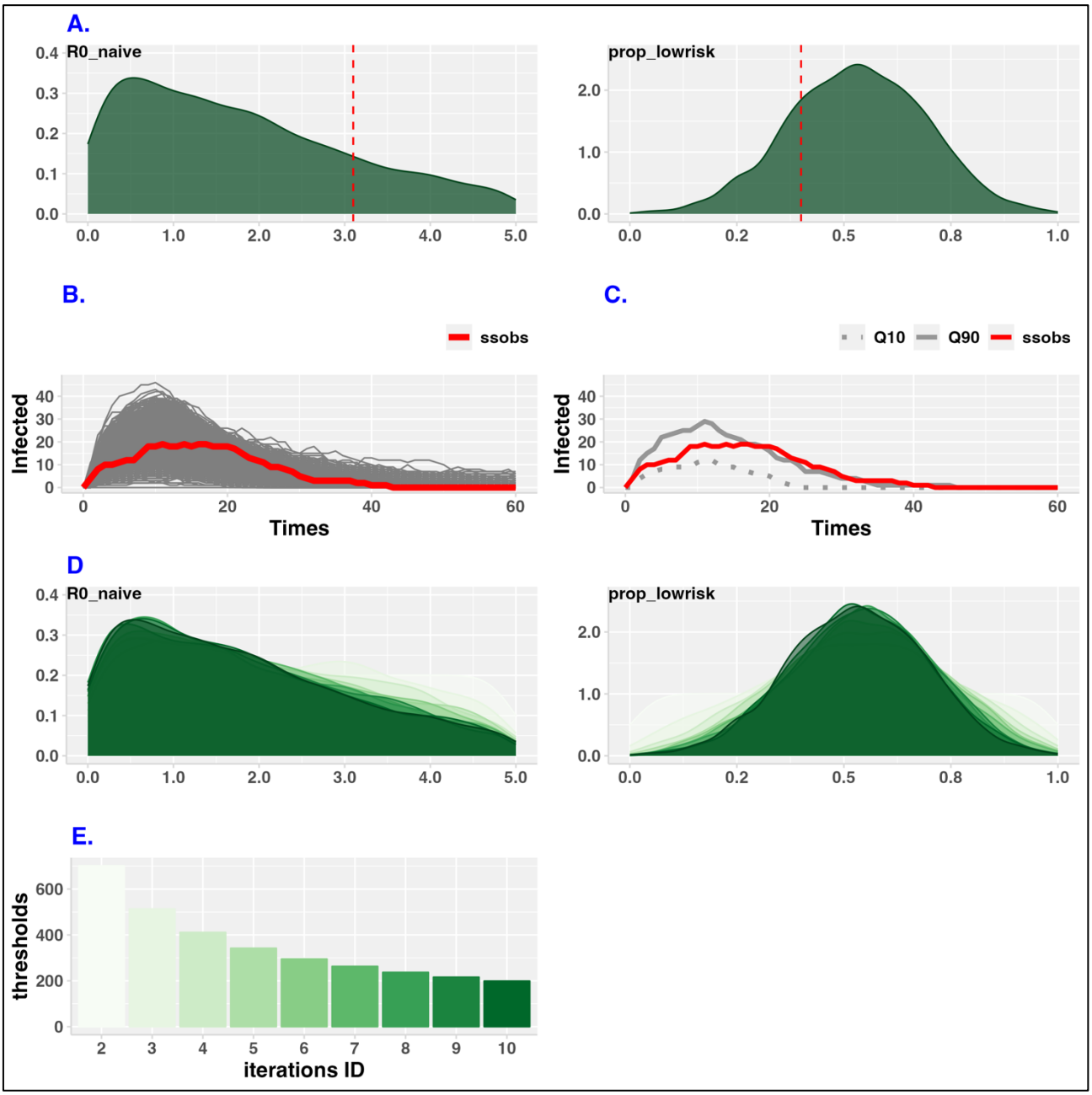

Figure 7. **Scenario Five Batch - All Infected - BRSV**: A. Density of the 3,000 last particles accepted, which forms the posterior of parameters $R0\_naive$ and $prop\_lowrisk$; Vertical lines correspond to target parameter values; B. trajectories obtained from the 3,000 accepted particles (gray) and observed trajectory (red); C. quantile trajectories (gray; Q10 in dotted line; Q90 in solid line) obtained from accepted particles and observed trajectory (red); D. Visualization of the 3,000 particles accepted at each iteration; E. Reduction in the tolerance margin among iterations, making it easier to check the algorithm convergence using visualization methods.

Table 2. Modes and credibility intervals of the two estimated parameters for Examples 1, 2, and 3, and the two packages considered.

		"BRREWABC" package	**"abc" package**
Example 1	Mode of R_O_naive	3.073	3.174
	Mode of Prop_low	0.400	0.410
	90% credible intervals of R_O_naive	[0.417; 4.689]	[0.551; 4.678]
	90% credible intervals of Prop_low	[0.137; 0.589]	[0.102; 0.665]
Example 2	Mode of R_O_naive	2.817	2.576
	Mode of Prop_low	0.318	0.253
	90% credible intervals of R_O_naive	[0.370; 4.659]	[0.272; 4.335]
	90% credible intervals of Prop_low	[0.097; 0.559]	[0.071; 0.655]
Example 3	Mode of R_O_naive	0.529	
	Mode of Prop_low	0.533	
	90% credible intervals of R_O_naive	[0.145; 4.283]	
	90% credible intervals of Prop_low	[0.259; 0.789]	

EXAMPLE 4: Use of methods on another pathogen parameterization and influence of observation particularities

Our aim with this example is to show that the inference results vary depending on the observed trajectory, despite using a similar model. As in Example 3, we considered 5 batches of 10 animals each. Unlike the previous examples, the circulating pathogen was *M. haemolytica.* Parameters to be estimated were transmission rates (`transmission_low = 0.005, transmission_high = 0.01125`) and the initial proportion of non-carrier animals (`prop_lowrisk = 0.1`). Three synthetic observed data were considered in this example, obtained by simulating 60 trajectories, of which were chosen the closest ones to the first quartile (Q1) (Figure 8A), the median (Figure 8B), and the third quartile (Q3) (Figure 8C), respectively.

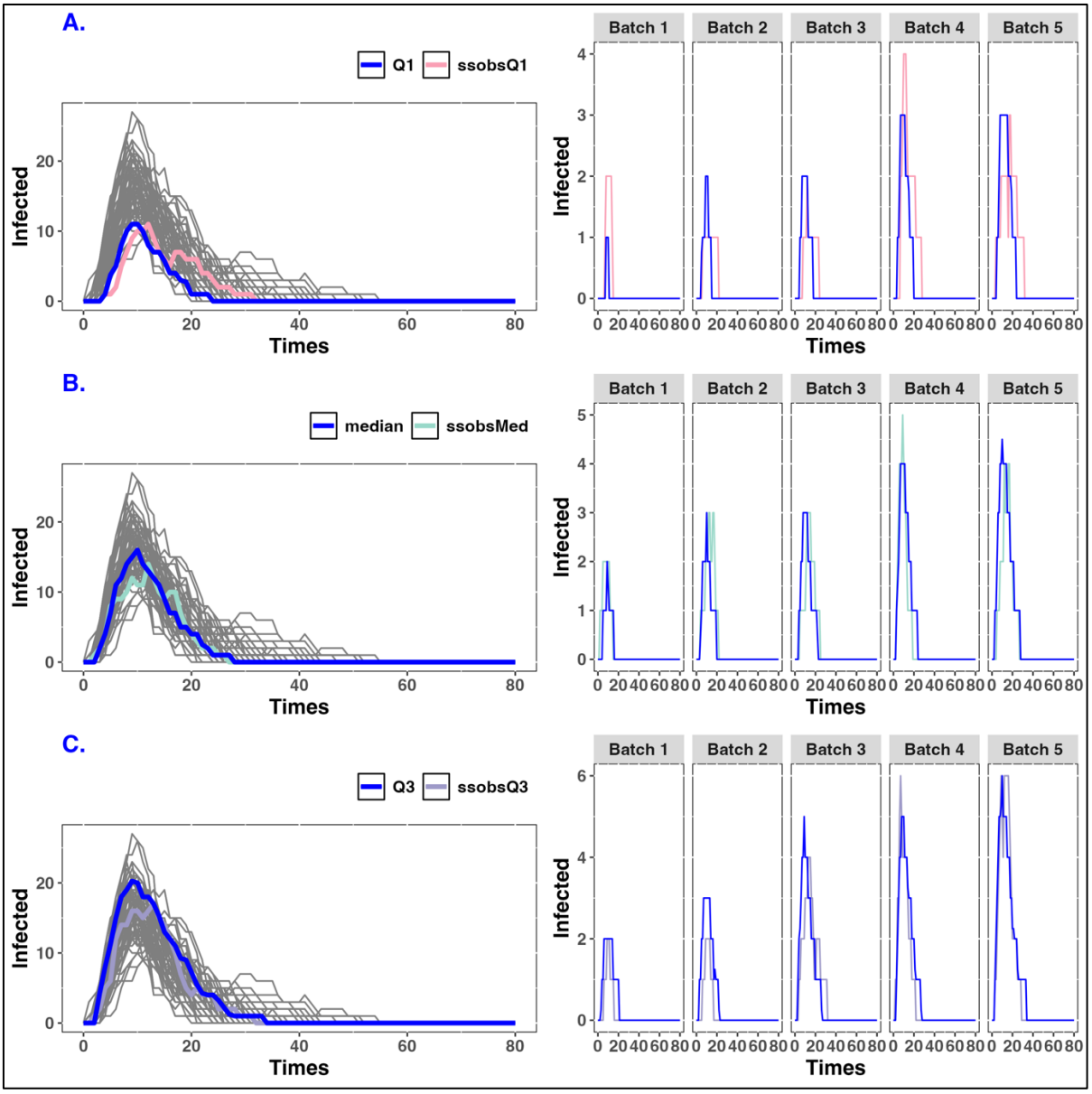

Figure 8. **Scenario Five Batch - All Infected - M. haemolytica**: Visualization of the 60 simulated trajectories (gray) with the target parameters ($transmition\_low = 0.005$, $transmition\_high$ = 0.01125, $prop\_lowrisk = 0.1$). A. First quantile; B. Median; C. Third quantile derived from the ensemble of these 60 trajectories (blue); and associated synthetic trajectories approximating them (pink, light green, and purple respectively).

The package "BRREWABC" was applied with:

It should be noted that in this example we modified (reduced or increased) certain inputs of the procedure to allow faster execution on a local computer system. However, that increasing these inputs would have been beneficial. In general, to determine the optimal input values, start with small values and gradually increase them, evaluating both the computation time and the quality of the estimates until the best compromise is found.

- A number of particles to be accepted equal to 200;
- A squared Euclidean distance corresponding to the sum of the squares of the differences between the simulated and observed summary statistics;
- A minimum acceptance rate equal to 0.01;
- A maximum number of simulations per iteration equal to 15;
- A maximum number of simulations per iteration of 100,000
- a uniform prior distributions, between [0, 0.5] for `transmission_low` and `transmission_high`, and between [0, 1] for `prop_lowrisk`. We changed the parameters because we ran this experiment locally instead of using a compute server.

The results are presented in Figure 9A, B, and C.

In all the estimated results, there is a tendency to over- or underestimate parameter values (`transmission_low = 0.005`, `transmission_high = 0.01125`, `prop_lowrisk = 0.1`) used to generate the observed trajectories, regardless of which trajectory (Q1, median, Q3) is considered as observed (Cf. Figure 9). However, these differences in degree vary from case to case, as shown in Figure 9 and Table 3 For example, for the trajectory close to Q3 (Figure 9C), the modes of the particle distributions accepted for `transmission_high` and `prop_lowrisk` parameters (0.008350843 and 0.1093644) are close to the values initially used. On the other hand, for the trajectory near the median (Figure 9B), only the mode of the accepted particle distribution for `prop_lowrisk` parameter is relatively close to the initial values, while for the trajectory near Q1 (Figure 9A) only the mode value for transmission_low parameter is slightly closer to the initial value, with a mode of 0.02404285. Overall, the credibility intervals remain wide, with the exception of the transmission_high interval for the median trajectory, which is rather narrow (Table 3). These enormous ranges of values once again confirm the large uncertainty in the parameters needed to reproduce the observed trajectory, due to the high stochasticity of the model.

Figure 9A', B', and C' show that the infection trajectories simulated from the latest particles accepted by the "BRREWABC" algorithm capture well the variability and uncertainty of the observed data. The Q10 and Q90 quantiles provide a clear view of this uncertainty, framing the observed trajectories and demonstrating the robustness of the inference despite the inherent differences between quartiles.

In conclusion, the parameter estimation results obtained for `transmission_low`, `transmission_high`, and `prop_lowrisk` using the "BRREWABC" package show precise convergence and close agreement with the observed data. This underscores the necessity of adjusting the inputs (e.g., increasing the number of particles to be accepted, the maximum number of simulations per iteration; decreasing the minimum acceptance rate) to improve this limitation. However, this will require substantial computation time and significant local computing power. They also show that inference estimates, based on these observed trajectories, largely vary according to the trajectory used as observed, despite the use of a similar model and process. This variability is observed in all parameters, but also in the population dynamics obtained from the accepted particles. These results underscore the importance of incorporating the variability of observed trajectories into epidemiological analyses to make reliable predictions about the spread of infection.

Table 3. Mode and credibility interval values for all parameters

	Mode			90% credible intervals		
	transmis-sion_low	**transmis-sion_high**	**prop_lowrisk**	**transmission_low**	**transmission_high**	**prop_lowrisk**
Q1	0.024	0.011	0.911	[0.003; 0.389]	[0.001; 0.324]	[0.071; 0.986]
Median	0.091	0.002	0.060	[0.007; 0.479]	[0.000; 0.010]	[0.014; 0.357]
Q3	0.074	0.008	0.109	[0.021 0.431]	[0.002; 0.051]	[0.021; 0.477]

Scenario Five Batch - All Infected - *M. haemolytica*: Left: density of the 200 last particles accepted, which forms the posterior of parameters $transmition\_low$, $transmition\_high$ and $prop\_lowrisk$; Vertical lines correspond to the target parameter values, the observed trajectory being the near Q1 (A), the near median (B) and the near third quantile (C), respectively. Right: the first column contains the trajectories obtained from the 200 accepted particles (gray) and the observed trajectories for the first quantile (A', pink), for the median (B', light green), and for the third quantile (C', purple); the second column contains the quantile trajectories (Q10: dotted line; Q90: solid line) obtained from the 200 accepted particles (gray), the observed trajectories being colored.

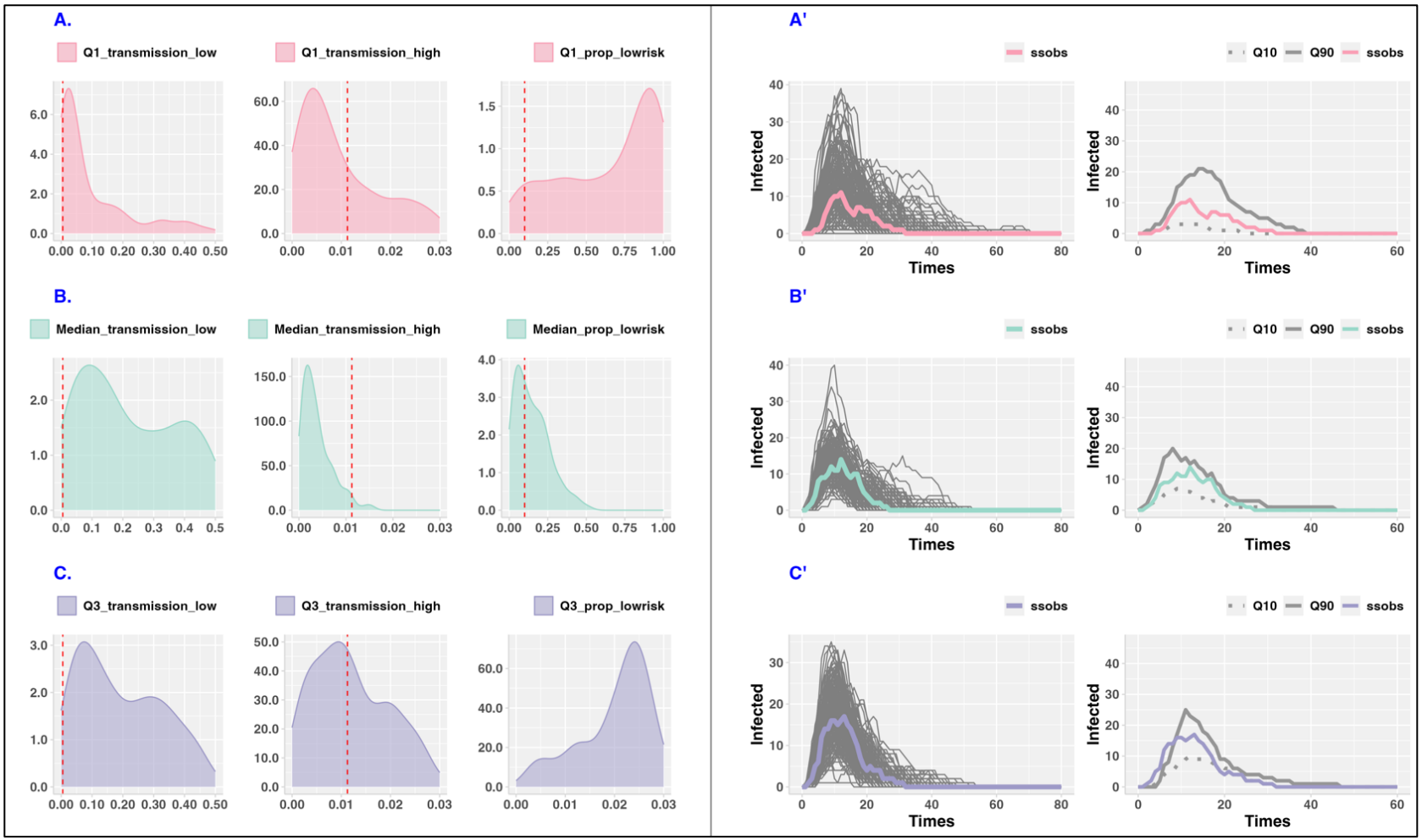

Figure 9. **Scenario Five Batch - All Infected - *M. haemolytica***: Left: density of the 200 last particles accepted, which forms the posterior of parameters $transmition\_low$, $transmition\_high$ and $prop\_lowrisk$; Vertical lines correspond to the target parameter values, the observed trajectory being respectively the near Q1 (A), the near median (B) and the near third quantile (C). Right: the first column contains the trajectories obtained from the 200 accepted particles (gray) and the observed trajectories for the first quantile (A', pink), for the median (B', light green), and for the third quantile (C', purple); the second column contains the quantile trajectories (Q10: dotted line; Q90: solid line) obtained from the 200 accepted particles (gray), the observed trajectories being colored.

SECTION 3 - CONCLUSION AND DISSCUSION

We have described and illustrated in this report two efficient ABC procedures for estimating model parameters while providing a posterior distribution of the most plausible values: the "abc" package for ABC regression, and the "BRREWABC" package for ABC-SMC.

To illustrate our approach, we applied the two processes to a stochastic epidemiological model of bovine respiratory disease. An initial parameter set was used to generate synthetic data, which were then considered as observed data. In some cases, the densities of the posterior distributions obtained corresponded well to parameter values used to generate the observed data (parameter values of synthetic data), while in other cases they did not. However, for all results: In each case, the parameters used to simulate the synthetic observations are included in the posteriors. Furthermore, the results obtained when the estimated values did not match the parameter values of synthetic data do not call into question the efficiency and reliability of the inference procedures. Indeed, the posterior trajectories simulated with the latest accepted particles always closely matched the observed trajectory, thus providing a distribution of the best parameter sets to achieve the observed trajectory. This phenomenon can be explained by the inherently stochastic nature of the model. In a stochastic model, random processes can cause significant variation between simulations, even when parameters are fixed. Therefore, estimated values can sometimes deviate from the initial values without indicating an error or inefficiency of the model. This highlights the robustness of the model which, despite these variations, manages to produce trajectories that closely match the observed data. This observation allows us to conclude that the observed trajectory was probably not representative of the initial parameters due to the highly random nature of the model. It also shows that despite the inherent variability of stochastic processes, the model is able to adapt and make reliable predictions.

The "abc" package offers several notable advantages that have made it popular with users wishing to perform Approximate Bayesian Computation (ABC) analyses. First, its ease of use and comprehensive documentation make it accessible to less experienced users. Once the initial simulations of summary statistics have been performed, parameters estimation becomes very fast, allowing results to be obtained without significant additional delay. In addition, simulated data can be reused for other observed data sets that reflect the same information, facilitating the creation of a useful data library for future studies.

However, the "abc" package has its drawbacks. The initial process of simulating the many summary statistics can be very time consuming and expensive in terms of computational resources. This cumbersome step can limit the effectiveness of the package, especially for users with limited resources. In addition, once simulations have been run, the model may lack flexibility to adapt to changes in model structure or to incorporate new information. Finally, "abc" may be less effective for high-dimensional models or problems with many variables, as the increased complexity of initial simulations can make analyses more difficult and time-consuming.

The "BRREWABC" package for ABC-SMC provides robustness and accuracy, making it a valuable tool for complex epidemiological analyses. One of the main advantages of this package is its ability to generate posterior estimates that progressively approximate the true posterior distribution over the course of iterations, thus improving the robustness and accuracy of estimates. In addition, BRREWABC" incorporates a parallelization process that speeds up computations and allows more complex models to be handled in less time. This package also offers remarkable flexibility, allowing tuning parameters of the algorithm to be changed during processing and the analysis to be resumed from the point at which it was interrupted. For example, if at the tenth iteration it becomes clear that the number of particles is insufficient, it is possible to pause the process, adjust the number of particles, and resume the analysis at the same iteration. This feature allows the process to be dynamically adapted to specific needs and observations, ensuring more accurate and efficient estimates.

Despite its many advantages, BRREWABC does also have its challenges. The package can be computationally intensive, especially for very complex models or large particle populations, requiring a larger investment in hardware. Although the process is parallelized to reduce computation time, the initial computation time can still be significant, especially for very complex models, which can slow down initial analyses.

The choice between the "abc" and "BRREWABC" packages depends on the specific needs of the study and the resources available. The "abc" package is recommended for users looking for a quick and easy solution after an initial simulation phase, especially when resources are limited, and the simulated data can be reused for different analyses. On the other hand, "BRREWABC" is better suited for studies that require high accuracy, robust uncertainty management, and the ability to handle complex models, with parallelization capabilities to speed up computations. Users should carefully evaluate the characteristics of their model, the computational resources available, and the need for flexibility and accuracy before choosing the best package for their application.

Bibliography

[1] K. Csillery, M. G. B. Blum, O. E. Gaggiotti and O. Francois, "Approximate Bayesian Computation (ABC) in practice," *Trends in Ecology and Evolution,* vol. 25, no. 7, pp. 410-418, 2010.

[2] M. A. Beamont, "Approximate Bayesian computation in evolution and ecology," *Annual Review of Ecology, Evolution, and Systematics,* vol. 41, pp. 379-406, 2010.

[3] J. K. Pritchard, M. T. Seielstad, A. Perez-Lezaun and M. W. Feldman, "Population growth of human Y chromosomes: a study of Y chromosome microsatellites," *Molecular Biology and Evolution,* vol. 16, no. 12, pp. 1791-1798, 1999.

[4] S. Tavaré, D. J. Balding, R. C. Griffiths and P. Donnelly, "Inferring coalescence times from DNA sequence data," *Genetics,* vol. 145, no. 2, pp. 505-518.

[5] M. A. Beaumont, W. Zhang and D. J. Balding, "Approximate Bayesian computation in population genetics," *Genetics,* vol. 162, no. 4, pp. 2025-2035, 2002.

[6] P. Del Moral, A. Doucet and A. Jasra, "Sequential Monte Carlo samplers," *Journal of the Royal Statistical Society: Series B (Statistical Methodology),* vol. 68, pp. 411-436, 2006.

[7] M. G. B. Blum and O. François, "Non-linear regression models for Approximate Bayesian Computation," *Statistics and Computing,* vol. 20, no. 1, pp. 63-73, 2010.

[8] F. Liang, C. Liu and R. J. Carroll, "ABC-regression: A general approach for Bayesian regression models," *Journal of the American Statistical Association,* vol. 112, no. 518, pp. 1330-1341, 2017.

[9] P. Marjoram, J. Molitor, V. Plagnol and S. Tavare, "Markov chain Monte Carlo without likelihoods," *Proceedings of the National Academy of Sciences,* vol. 100, no. 26, pp. 15324-15328, 2003.

[10] K. Csilléry, O. François and M. G. B. Blum, "abc: an R package for approximate Bayesian computation (ABC)," *Methods in Ecology and Evolution,* vol. 3, no. 3, pp. 475-479, 2012.

[11] G. Beaunée, "BRREWABC: Batched Resilient and Rapid Estimation Workflow through Approximate Bayesian Computation.," 2024. [Online]. Available: https://github.com/GaelBn/BRREWABC. [Accessed 30 Mai 2024].

[12] B. Sorin-Dupont, S. Picault, B. Pardon, P. Ezanno et S. Assié, «Modeling the effects of farming practices on bovine respiratory disease in a multi-batch cattle fattening farm,» *Preventive Veterinary Medicine,* vol. 106009, 2023.

[13] S. Picault, Y.-L. Huang, V. Sicard, S. Arnoux, G. Beaunée and P. Ezanno, "EMULSION: transparent and flexible multiscale stochastic models in human, animal and plant epidemiology," *PLoS Computational Biology,* vol. 15, no. 9, p. e1007342, 2019.

APPENDIX

This appendix extracted in full and without modification from B. Sorin-Dupont et al. (2023) as they describe processes and transitions of the model.

Processes of the model

In the model, five processes drive the states of individuals: infection, hyperthermia, clinical signs, detection and treatment. They are represented by a formalism broadly used in computer science, finite state machines, which is close to flow diagrams used by epidemiologists, with a higher expressiveness.

Hyperthermia

Hyperthermia was composed of two states: hyperthermic (H) and non-hyperthermic (NH). NH animals could transition to H with a probability of p_H attributed to non-infectious factors. Once in the H state, they remained there for a period τ_H sampled from a Beta distribution adjusted using observed data, before reverting to NH. Additionally, the transitions from NH to H and back could result from the infection process.

Infection and clinical signs status

Four health statuses were considered: susceptible (S), asymptomatic carrier (E), infectious (I) and resistant (R) animals. Asymptomatic carriers could spontaneously turn I with probability p_E and could also be infected by surrounding infectious individuals (I). Three actions were triggered when entering the I state: (1) the individual exhibited mild clinical signs for a duration τ_M drawn from a Beta distribution calibrated from observed data, (2) the animal changed from NH to H state, (3) a random sort with probability p_C drove whether the individual would display severe clinical signs at the end of its mild clinical signs. If displaying severe clinical signs, a boolean deciding on the survival of the individual was drawn from a binomial law of probability p_d. Death occurred at the end of the severe clinical signs duration (τ_C). If the individual did not die from infection, it then recovered and became resistant (R). Recovery occurred after duration τ_I drawn from a gamma distribution according to the given bacterial pathogen. Theoretically, τ_I is longer than τ_M . However, $\tau_M + \tau_C$ could exceed τ_I. In that case, the infectious period was $\tau_M + \tau_C$. When transitioning to R state, animals changed from H to NH.

Detection status

Two detection methods were used. The first detection relied on visual on-farm appraisal of clinical signs, assuming lethargy was the most significant sign to calibrate the delay (τ_M) between infection and severe sign occurrence. Severe clinical signs were detected with a sensitivity of 1, while the sensitivity for mild clinical signs detection was assumed to be 0.5. The model assumed a clinical check-up at every time step (12 hours). Following the detection of the first case through visual appraisal, all hyperthermic animals were identified using rectal temperature measured at the next feeding time, 12 hours later.

Treatment status

Each animal detected as diseased or hyperthermic, or in the context of collective treatment could transition from not treated (NT) to treated (T). Treated animals received one antibiotic dose, assumed to be effective after a certain duration τ_T . If animals still exhibited clinical signs after this duration, they would be treated again for the same duration, but the number of treatments per individual per episode was limited (max_T). Transitions from T to NT occurred in three cases: (1) recovery after τ_T due to successful treatment with probability P_T, (2) the end of the infectious period occurred while under treatment but was not caused by it, (3) treatment failure after max_T doses. A treatment success triggered the transition from I to R, consequently trigerring transitions driven by the end of the infectious state i.e. the transitions to non-hyperthermic and to asymptomatic states.