BICI v0.80: Bayesian Individual-based Compartmental Inference

Christopher M. Pooley[1], Andrea B. Doeschl-Wilson[2], Grant Henderson[1] and Glenn Marion[1]

[1] Biomathematics and Statistics Scotland, James Clerk Maxwell Building, The King's Buildings, Peter Guthrie Tait Road, Edinburgh, EH9 3FD, UK.

[2] The Roslin Institute, The University of Edinburgh, Midlothian, EH25 9RG, UK.

Table of Contents

1) Introduction

Compartmental models have long been used as a means of understanding the collective dynamics of interacting agents, with notable applications in epidemiology, chemistry and ecology. BICI allows for arbitrary compartmental model specification and performs simulation, inference and posterior simulation[1].

Models can contain multiple interacting species, which can each be treated at a population or individual level. BICI can be run entirely using a point-and-click interface. Alternatively, a scripted language, referred to as "BICI-script", can be used to construct, store, and export complex models and allow for them to be run on HPC (high performance computing).

For inference, BICI accepts a variety of individual and/or population-level data, and priors can be specified from a large range of possibilities. Posterior parameter outputs include trace plots, distributions, correlations, and summary statistics (means and 95% credible intervals) as well as diagnostic information (*e.g.*, measuring MCMC convergence and helping to identify potential model misspecification). State outputs include various visualisations for populations, transitions and individuals.

A detailed description of the statistical model underlying BICI, along with the Bayesian inference methodologies, is given in an accompanying paper [1]. The focus of this manual is on the practicalities of running the BICI software tool.

1.1 Downloading

BICI is freely available under the GNU General Public License, and can be downloaded from the GitHub repository github.com/theITEAM/BICI.

Depending on your platform, the following instructions explain how BICI is run:

- **Windows** – Download the file "BICI_v0.5_windows.zip" and unzip. BICI is run by clicking on the "BICI.exe" icon (if the error message "Windows protected your PC" appears, click on "More info" and "Run").
- **Linux** – Download the file "BICI_v0.5_linux.tar.gz". This can be extracted by using the terminal command "tar -zxvf BICI_v0.5_linux.tar.gz". The code is executed using "./BICI".
- **Macintosh** – Download the file "BICI_v0.5_Mac.zip" and unzip. BICI is run by clicking on the "BICI.app" icon (if the error message "BICI can't be opened because it is from an unidentified developer…" appears, right click on "BICI.app" and select "Open" to give the option to run).

1.2 The user

Users of BICI may come from diverse backgrounds and various disciplines. We imagine that it will be used by different people in different ways. Here are some examples:

- **The casual user** – They might work entirely within the interface. They could, for example, create and run simple SIR-type models using the point-and-click interface. Inference could be done using time-series population infection data in '.csv' format. BICI could also be useful as a teaching aid to introduce concepts in epidemiological modelling.

[1] A simulation samples system dynamics, otherwise referred to as a "state", from the model given a set of model parameters. Inference generates estimates for model parameters and state based on data. Posterior simulation simulates from the model using model parameters estimated from inference.

- **The serious user** – Complex models can be loaded using input files or through BICI-script. The interface can generate executable files that can be run in parallel on a Linux cluster. The outputs from this can then be read back into the interface for visualisation.
- **The pipeline user** – BICI-script can be created by another program, run by the core code and the output files used for some other purpose, *i.e.* a seamless pipeline which doesn't require the visual interface at all.

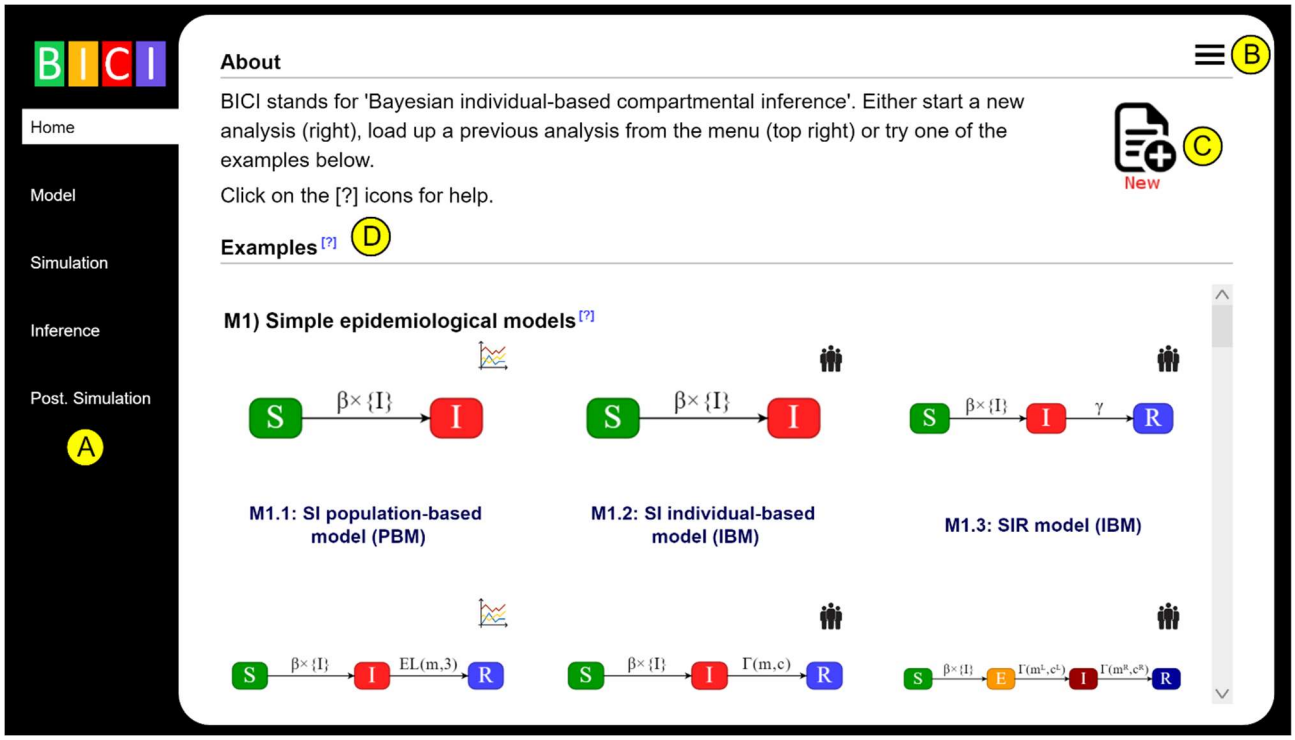


Figure 1 – *The home page*. A: The main menu (used for navigating between pages), B: The drop-down menu (for loading saving and exporting, see §6.2), C: Start a new model, D: Examples (see §8).

1.3 Getting started

Figure 1 shows the home page displayed when BICI is first loaded. The main menu on the left (Fig. 1A) is used to navigate from page to page. Opening one page may open up further submenus with different options. The structure of this manual broadly follows the tree structure of this menu.

Section 2 describes setting up a compartmental model, §3 deals with simulation from the model, §4 incorporates data to allow for Bayesian inference and, finally, §5 looks at posterior simulation (which can be used for future prediction and counterfactual analysis). Section 6 discusses input and outputs, §7 looks at BICI-script (with further details in Appendices A, B and C), and, finally, an extensive set of examples is provided in §8.

To start using BICI three options are available: a previous model/analysis can be loaded from the drop-down menu (Fig. 1B, see §6.2), a new model can be started (Fig. 1C) or one of the example applications can be investigated (Fig. 1D). These examples demonstrate a wide variety of different models and data scenarios that illustrate the capabilities of BICI in a variety of different situations (see §8). They can be

modified in any way and their default settings will be restored when reloaded from the home page (Fig. 1D).

New users are encouraged to try the examples first and spend some minutes exploring the software to get a feel for how BICI's interface works. Alternatively, they can have a go at creating their own model by following the tutorial in the next section.

Whilst exploring the software, make use of the many [?] help buttons that provide much of the information outlined in this manual.

1.4 Tutorial

In this tutorial the user will learn how to:

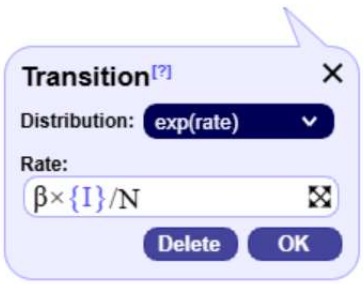
- Construct a simple SIR model.
- Simulate from that model.
- Generate a simulated dataset.
- Do inference on that data to generate parameter and state estimates.
- Perform a counterfactual analysis.

1.4.1 Setting up a simple SIR model

Follow these steps:

- A new model can be started by navigating to the "Home" page (on the main menu on the left-hand side) and clicking the "New" button in the top right-hand corner.
- BICI allows for multiple interacting species to be added (for example, predator-prey models, or disease vector models). The example here focuses on just a single species. Let's add one named "People" and select it to be population-based.
- Next, we add a classification. BICI allows for individuals to have multiple classifications. For example, they may be classified by their disease status, or their age, or their locations. For now, we will set up a single classification called "DS" standing for "disease status" (don't worry about all the other options).
- The next job is to add the compartments. This can be done by clicking "⊕ Compartment" on the lower menu bar. Let's set up three compartments called 'S', 'I' and 'R' and give them different colours. The workspace should look something like this:

- Now let's add some transitions. These allow individuals to change the compartment they are in. Select "⊕ Transition" from the lower menu bar and click first on 'S' and then on 'I'. An arrow going from one to the other should appear. Attached to this is a panel with gives details of the transition. The "distribution" option determines the type of probability distribution that describes the time the individual resides in 'S' before moving to 'I'. The "exp(rate)" option means that there is a certain defined rate with which the transition occurs (*i.e.* a Poisson process).



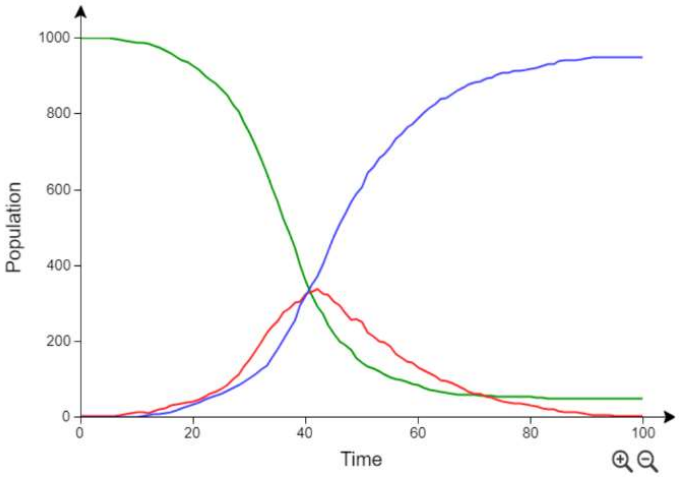
- Here we set that rate to 'β×{I}/N' (note, Greek characters can be added by typing using Latex notation, *e.g.* '\beta' automatically becomes 'β', and the '*' character automatically becomes '×').

In BICI notation the curly brackets denote "the population of", so '$\{I\}$' represents the total number of infected individuals.

- Notice that there is no 'S' in this expression. This is because BICI uses individual rates of transition, not population-wide rates (as are sometimes shown).
- Add in a recovery rate γ and the model should look something like this:

1.4.2 Simulation

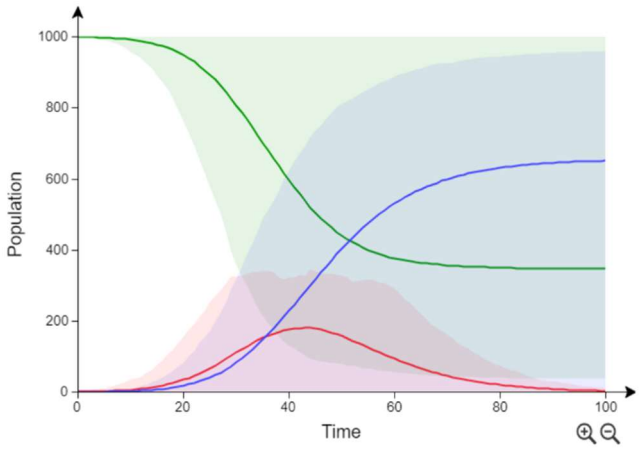
- Next, we add some individuals to the model. Click on the "Simulation→Initial Conditions" page and then "⊕ Init. Pop.". Click "Next" to indicate that we are setting up a fixed initial population with the graphical interface.
- Click on the different compartments to select how many individuals start in each of them. Set them such that $S=1000$, $I=1$ and $R=0$ (*i.e.,* we are going to look at the effect of a single infected individual entering an otherwise susceptible population) and click "Done".
- On the "Simulation→Parameters" page set $N=1001$ (*i.e.,* the total population size), $\beta=0.3$ and $\gamma=0.1$ (these correspond to an individual taking on average 10 days to recover from infection and a basic reproduction ratio of $R_0=\beta/\gamma=3$).
- We now set up the simulation on the "Simulation→Run" page. Select a start time of 0, an end time of 100 days and a time-step of 1 (note, longer time-steps speed up simulation/inference but can lead to discretisation errors[2]).
- Click on "Start" to see what happens…



- The output should be something like this, which shows the population variation over time in the three compartments.
- Try looking at different visualisations on the "Simulation→Results" page.

[2] As a rough guide, provided the time-step is less than ~20% of the shortest mean transition time, the discretisation error should be relatively minor.

- Go back to "Simulation→Run" and set the simulation number to 100. This produces 100 simulations. The solid lines give the average behaviour, and the shading indicates the region that encompasses 95% of simulations.

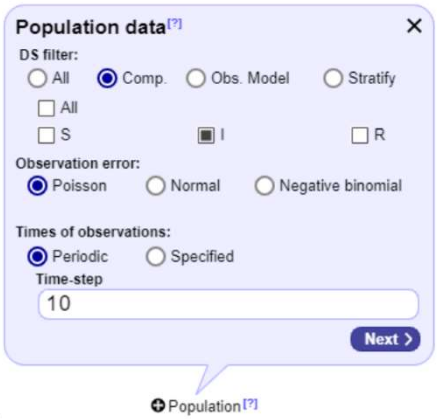


- We find huge so-called "stochastic" variability[3]. Sometimes "epidemic extinction" occurs when, despite the fact the basic reproduction number[4] is greater than one R_0>1, the epidemic dies out for stochastic reasons.

1.4.3 Generating simulated data

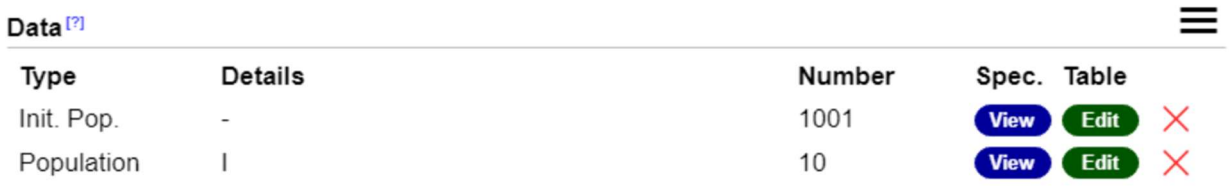
We now generate some hypothetical data from the simulation results in the previous section:

- Go to the "Simulation→Generate Data" page and select "Init. Cond." from the drop-down list in the bottom left-hand corner.
- Click on "⊕ Init. Pop.". This shows data representing the initial population. Click "Done" twice to add to the list of generated data sources.
- Select "Population" from the drop-down list in the bottom left-hand corner and click on "⊕ Population".
- We now tell BICI that we want to generate time series data that estimates the infected population at periodic time intervals. Click on "Comp." and then select the 'I' checkbox (see right). The observation error is set to 'Poisson', which means that there is some inherent error in the observed values compared to the real values. We set a time-step of 10 days and click "Next" to generate the data.
- Click "Done" to add this data to our list.
- Finally, click on "Copy" to copy the simulated data into the "Inference→Data" page so we can perform inference on it.



1.4.4 Inference

- The data sources should look like this:



- Now the data has been added, let's consider the prior. Go to the "Inference→Prior" page.

[3] Stochasticity is defined as having a random probability distribution or pattern that may be analysed statistically but not predicted precisely. Epidemics are stochastic because they result from random processes, such as random contacts within a large group of individuals.

[4] R_0 is defined as the expected number of cases directly generated by one case in a population where all individuals are susceptible to infection.

- The prior captures previous knowledge about the parameter values before the data is considered. For example, if we are very certain about a quantity, we could fix its value. If, on the other hand, we are very uncertain, we could use a wide distribution to represent a plausible range of values.
- Supposing we don't know much about the disease, here the prior is chosen to be wide and relatively uninformative: a uniform prior between zero and one for β and γ.

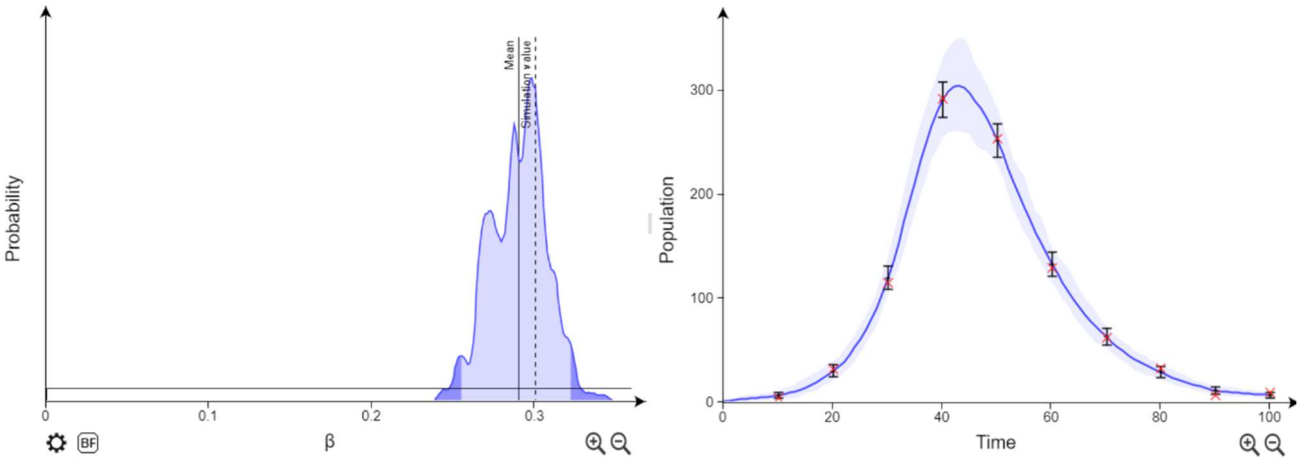
Parameters [?]

$N \sim$ fix(1001)

$\beta \sim$ uniform(0,1)

$\gamma \sim$ uniform(0,1)

- N, the population number, is fixed to 1001.
- We start inference by going to the "Inference→Run" page and filling out a start time of 0, an end time of 100 and a timestep 1.
- BICI gives some different possibilities for the algorithm used to perform inference. Here we select the "DA-MCMC" option.
- The number of updates informs how long the algorithm takes to run. Running for longer may produce a better approximation to the posterior, but it can also be computationally wasteful. *A priori* it is difficult to know what this number should be, but the default value of 5000 is usually a good first guess. MCMC diagnostics can be used to determine if inference needs to be rerun using more updates (see §4.4.7 for details).
- MCMC "chains" run in parallel, so helping to speed up computation. Each chain runs on a separate CPU core. Since modern computers typically have 4-8 cores, so the default number of chains is set to 3 (to avoid the computer slowing down too much). An additional advantage of running parallel chains is that it tells us something about the reliability of the results (see §4.4.7).
- Click on the "Start" button to begin the inference.
- This example should take just a few seconds to run…
- A variety of different visualisations of the posterior can be found on the "Inference→Results" page (see Appendices H-K).
- Here are some results that show: (1) the posterior distribution for the parameter β on the left (note, this distribution contains the value used to simulate the data, denoted by the vertical dashed line) and (2) the infected population as a function of time on the right (where the red crosses represent the observed data and the error bars provide uncertainty in those measurements):



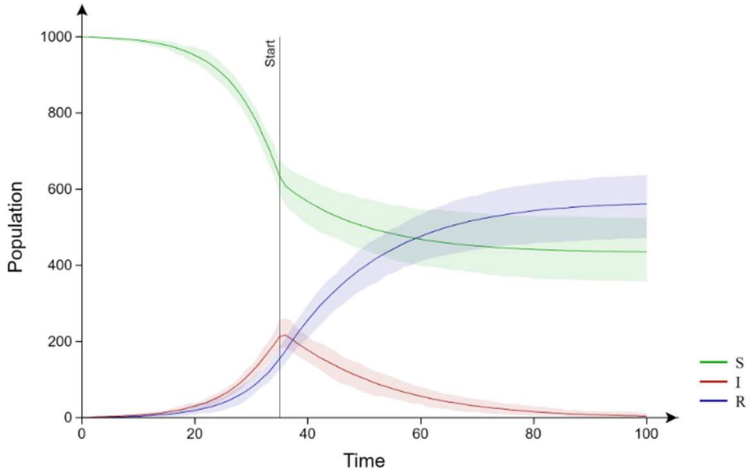
1.4.5 Posterior simulation

Posterior simulation can be used for future prediction and counterfactual analysis. Here we consider the effect of an intervention to reduce disease transmission mid-way through the epidemic.

- Go to the "Post. Simulation→Parameter" page and click on "⊕ Multiplier". Select the parameter β.
- "Multipliers" act by multiplying the corresponding parameter by a potentially time varying factor.

Set the parameter multiplier as follows:

- This defines a spline that starts with a constant value of one until day t_{inter}=35, at which point it sharply falls to 0.3 for the remaining time (this can be visualised by clicking on the "View" button). This is equivalent to a disease intervention introduced at t_{inter} that reduces the transmission rate β to 30% its previous value (for example, this could be due to the introduction of mask wearing during the Covid-19 epidemic).
- Go to the "Post. Simulation→Run" page and set the start time to 35 and the end time to 100. This corresponds to the epidemic progressing as before until t_{inter} and subsequently predicting the effect of the intervention.
- Click "Start".
- The graph on the right shows that the intervention drastically reduces the impact of the epidemic.



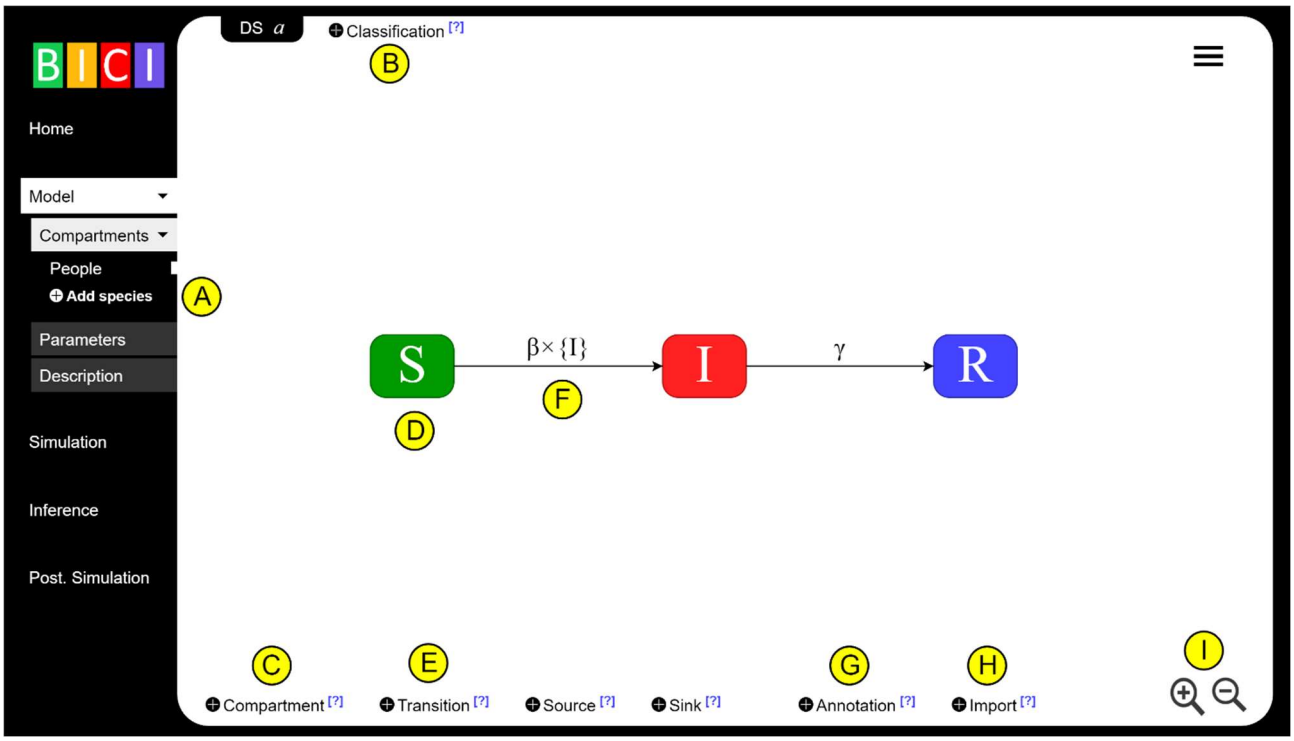


Figure 2 – *The compartmental model*. This shows an SIR compartmental model (the compartments represent 'S' susceptible, 'I' infected and 'R' recovered from disease). A: Add a species, B: Add a classification, C: Add a compartment, D: A compartment, E: Add a transition, F: A transition, G: Add an annotation, H: Import compartments and transitions, I: Zoom in and out of the model.

2) Model

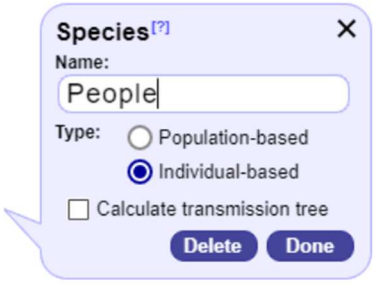
This section describes model specification, which encompasses both compartmental structure and parameter definitions.

2.1 Compartmental model

The "Model→Compartments" page (Fig. 2) shows how a compartmental model is defined. Three layers of division are used to differentiate individuals within the system. These are discussed in the next three sections:

2.1.1 Species

Species group together individuals of the same type. In many cases only one species will be present, *e.g.* people. In other cases, however, different species can interact, *e.g.* predator-prey models, disease models with vectors, or models that account for the accumulation of pathogen in the environment (here the pathogen, itself, becomes a species).



A species can be added by clicking on "⊕ Add species" (Fig. 2A). Two types of species can be created:

- **Population-based** – The total numbers of individuals in different compartments are tracked as a function of time. Only population-level data can be used and transitions are restricted to have either exponential or Erlang distributed waiting times.

- **Individual-based** – Here each individual in the system has its own timeline. This allows for individual-level data (as well as population-level data) to be utilised in inference, and enables the greatest flexibility to define transition distributions. Furthermore, the dynamics of individuals can vary (beyond their compartmental classification) thanks to so-called "individual" or "fixed" effects. Individual-based models, however, can become computationally slow when they contain many individuals.

Transmission tree – If the model is individual-based, a further checkbox allows the user to select if the transmission tree is calculated (this keeps track of who is infecting whom). This is of primary use in cases in which genetic information on the pathogen is available (otherwise it is generally turned off to help speed up computations). If selected, it is necessary to choose which classification relates to the infection process and then go on to select "infected" compartments in that classification (see §2.1.3).

See §2.1.12 for restrictions on the naming of species (along with other properties below).

2.1.2 Classifications

A classification is a discrete means of differentiating individuals. Examples of typical classifications include disease status, location or sex.

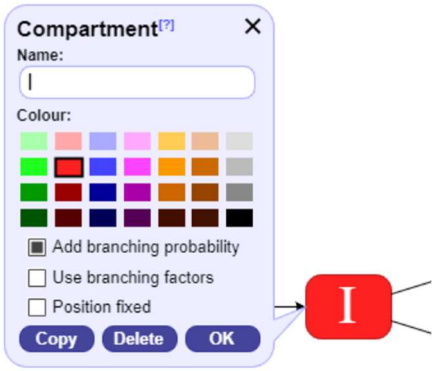
A classification can be added by clicking "⊕ Classification" (Fig. 2B) at the top of the page. It is usually given a short but meaningful name (such as 'DS' for "disease status") and an index. This index is a single letter that can be used in mathematical equations to represent all the possible values the classification can take.

A coordinate system is associated with the classification. Cartesian coordinates are used to represent the usual block style of compartmental models, as shown in Fig. 2 (*e.g.* the SIR model). In spatial models, an alternative is to use geographical coordinates (that use longitude and latitude information). These can represent a map-based distribution of points (*e.g.* farms) or regions (*e.g.* local authorities).

Cloning – "Cloning" a classification is used to copy compartments from one species to another (note, once cloned the two classifications must always share the same compartments, but may have different transitions). Cloning is useful when two or more species share the same classification (*e.g.* in disease transmission experiments the individuals and pathogen share the same set of closed contact groups).

2.1.3 Compartments

Compartments are possible states a classification can take. They are added by clicking "⊕ Compartment" (Fig. 2C), which allows the user to place them onto the workspace (Fig. 2D). A compartment name is usually chosen to be short (conventionally a single upper-case letter), *e.g.* 'S' to represent susceptible individuals. The colour of the compartment can be selected (by convention this is green for susceptible 'S' and red for infected 'I'). These colours are used for plotting population graphs when results are generated. The position of the compartment can be fixed by selecting the checkbox, otherwise it can be dragged around the screen with the cursor.



When multiple exponential transitions leave a compartment, the "Add branching probability" checkbox allows the user to select whether or not branching probabilities are implemented. If branching does occur,

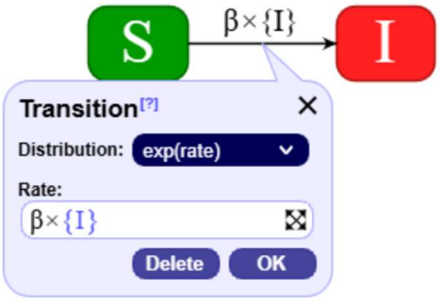
the “Use branching factors” checkbox specifies whether or not so-called “branching factors” are used (see §2.1.6).

In cases in which the transmission tree is turned on (see §2.1.1), a further checkbox allows the user to select if the compartment corresponds to an “infected” state.

See §2.1.9 for importing compartments into the model from a file (useful if there are many).

2.1.4 Transitions

Transitions allow for individuals to move from one compartment (initial) to another (final). Clicking “⊕ Transition” (Fig. 2E) and then selecting the initial and final compartments adds a transition to the model. For example, when an individual becomes infected it moves from an initial susceptible ‘S’ compartment to a final infected ‘I’ compartment (Fig. 2F).



Note, intermediary points can also be added to map out a path instead of just a line, *e.g.* this could be used to add a non-straight arrow going from ‘R’ back to ‘S’ again in Fig. 2 to incorporate waning immunity.

Different possible probability distributions can be selected for the transition time:

- **exp(rate)** – The exponential distribution is specified by a rate. Such transitions are known as “Markovian”, because they occur independently of when the individual entered the initial compartment. Such an approach is valid for random processes, for example contacts between different members of a population leading to the spread of infection.
- **exp(mean)** – As above, but specified through a mean.
- **gamma** – The gamma distribution is specified by a mean and coefficient of variation[5]. An example application of the gamma distribution is to model the incubation period (time duration between becoming infected and then becoming infectious) as used in an SEIR model. In reality such a transition is expected to be non-Markovian, because the time an individual becomes infectious (after becoming infected) depends on complex processes underlying immune system dynamics.
- **erlang** – The Erlang distribution is specified by a mean and shape (where the shape must take a positive fixed integer value). This is a special case of the gamma distribution.
- **log-normal** – The log-normal distribution is specified by a mean and coefficient of variation.
- **weibull** – The Weibull distribution is specified by a scale and shape. It is somewhat similar to the gamma distribution, but is used in preference under certain circumstances.
- **period** – Here the event happens after a certain prescribed time period. For example, this could be used to represent the predictable increase in age of an individual. The period must be specified as a positive number.

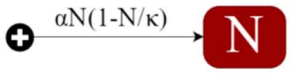
The defining quantities (mean, shape etc…) for these distributions are set via user-defined equations (see §2.1.10). Note, if a species is set to “population-based”, only the “exp(rate)”, “exp(mean)” or “erlang” distributions can be selected from.

[5] The coefficient of variation is equal to the standard deviation divided by the mean.

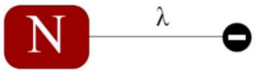
See §2.1.9 for importing transitions into the model from a file (useful if there are many)

2.1.5 Sources and sinks

A source allows for individuals to enter the system (*e.g.* as a result of births). The probability per unit time (or rate) at which this occurs is determined by a user-defined equation that can contain model parameters, populations and be dependent on other classifications. Unlike other transitions (which define individual-based transition probabilities), the equation for a source provides the overall rate at which individuals enter the system. Note, if the equation is stratified by another classification, the rates entering each subpopulation are specified separately[6].

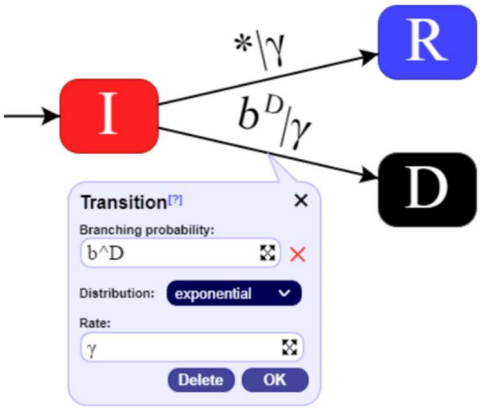


A sink allows for individuals to leave the system (*e.g.* as a result of death). Sinks have the same options as for other transitions.

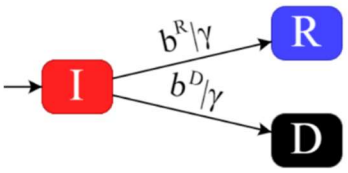


2.1.6 Branching

In cases in which multiple transitions leave an initial compartment, there are three possibilities:



- **No branching probabilities** – In this case, all the transitions must be exponentially distributed (note, even here branching probabilities can explicitly be added by clicking on the "Add branching probability" checkbox when the initial compartment is selected).
- **Branching probabilities** – An equation on the transition is used to determine the probability an individual selects that particular branch when they enter the initial compartment. Note, one of the transitions has no probability associated with it because, by definition, the probabilities must add up to one (this is denoted by "*" in the workspace).
- **Branching factors** – This option is turned on by clicking on the initial compartment and selecting the "Use branching factors" checkbox. The branching factor equation is defined down each transition to denote the *relative* probability of going down that branch. Consequently, the branching probability is given by the branching factor divided by the sum of all branching factors leaving the initial compartment. In the example on the right the probability of dying is given by $b^D/(b^D + b^R)$. Such a definition is useful because, unlike branching probabilities, branching factors are defined for all positive numbers making them suitable to incorporate fixed and/or individual effects.

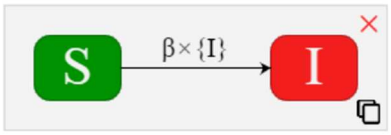


[6] To give an example, suppose a model has a source entering the S compartment and a further classification "Sex". If the source rate is defined as "α", this means α individuals enter the system per unit time on average, with approximately 50% male and 50% female. It the source rate is defined as "α_s", then a rate of α_M enter as male and a rate of α_F enter as female, *i.e.* the different possibilities add up.

2.1.7 Building up a compartmental model

Numerous compartments and transitions can be added to a classification. Compartments can be dragged across the workspace to move their location, or edited to change colour or name.

Multiple compartments can be selected by dragging a selection box with the right mouse button pressed down. Selected compartments can be moved, copied or deleted.

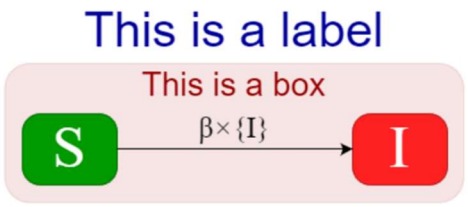


For large compartmental models (*e.g.* spatial models with many farms), it may be necessary to zoom in and out of the workspace. This can be done by clicking on the zoom buttons (Fig. 2I). Zooming in can, alternatively, be done by double clicking on the workspace.

2.1.8 Annotations

Although annotations play no part in how the model works, they can be used to clarify the meaning behind different elements of the model. They can be added by clicking "⊕ Annotation" (Fig. 2G).

Different types of annotations can be added:



- **Label** — These are user-defined text that can be positioned anywhere on the workspace.
- **Bounding box** — These are labelled boxes which bound selected compartments.
- **Map** – A map can be loaded in GeoJSON format (note, this is only available when geographical coordinates are used), *e.g.* maps could be added as a visual reference for farm locations.

2.1.9 Importing

Creating large numbers of compartments and transitions would become laborious using the point-and-click interface. For this reason, BICI allows them to be imported using "⊕ Import" (Fig. 2H).

The following import options are available:

- **Compartments** – Loads multiple compartments from a file.
  REQUIRES – A data table containing the column "Name", which gives the compartment names to be added. Optionally, position data can be also included by columns "x" and "y" giving the x and y position in the workspace (in Cartesian coordinates) or "Lat" latitude and "Long" longitude (in geographical coordinates). Also a column "Color" can be used to allocate colours to compartments (either in hexadecimal format, *e.g.* "#ff0000", or RGB format, *e.g.* "rgb(255,0,0)").
  EXAMPLE – This shows an example of an input .csv file (viewed in Excel) which loads up 'S', 'I' and 'R' compartments at specified locations and with specified colours:

	A	B	C	D	E	F	G	H
1	Name	x	y	Color				
2	S	-10	0	#00ff00				
3	I	0	0	#ff0000				
4	R	10	0	#0000ff				

- **Comp. Map** – Loads multiple compartments represented by geographical boundary data (in geographical coordinates only).

REQUIRES – A GeoJSON file from which regions can be selected. This is a special geographical file format which can be used, *e.g.*, to store geographical boundary data.

- **Transitions** – Loads multiple transitions from a file.
  REQUIRES – A data table containing at least the following columns: "From" gives the compartment from which the transition originates ('+' is used to represent a source), "To" gives the compartment to which the transition ends up ('-' is used to represent a sink) and "Value" specifies the transition. The format for this value depends on the transition distribution:
  - *exp(rate:[eq])* – An exponential distribution, where "[eq]" is replaced with an equation that determines the rate at which transitions occurs, *e.g.* 'r' or 'β×{I}'.
  - *exp(mean:[eq])* – An exponential distribution with an equation for the mean.
  - *gamma(mean:[eq1],cv:[eq2])* – A gamma distribution with equations for mean and coefficient of variation.
  - *erlang(mean:[eq],shape:[integer])* – An Erlang distribution with equation for mean and a fixed positive integer shape.
  - *log-normal(mean:[eq1],cv:[eq2])* – A log-normal distribution with equations for mean and coefficient of variation.
  - *weibull(scale:[eq1],shape:[eq2])* – A Weibull distribution with equations for scale and shape.
  - period(time:[eq]) – For when the event happens after a certain prescribed time period.

  EXAMPLE – This shows an example of an input .csv file which loads up transitions between 'S', 'I' and 'R' compartments:

	A	B	C	D	E	F	G	H
1	From	To	Value					
2	S	I	exp(rate:\beta*{I})					
3	I	R	gamma(mean:m,cv:c)					

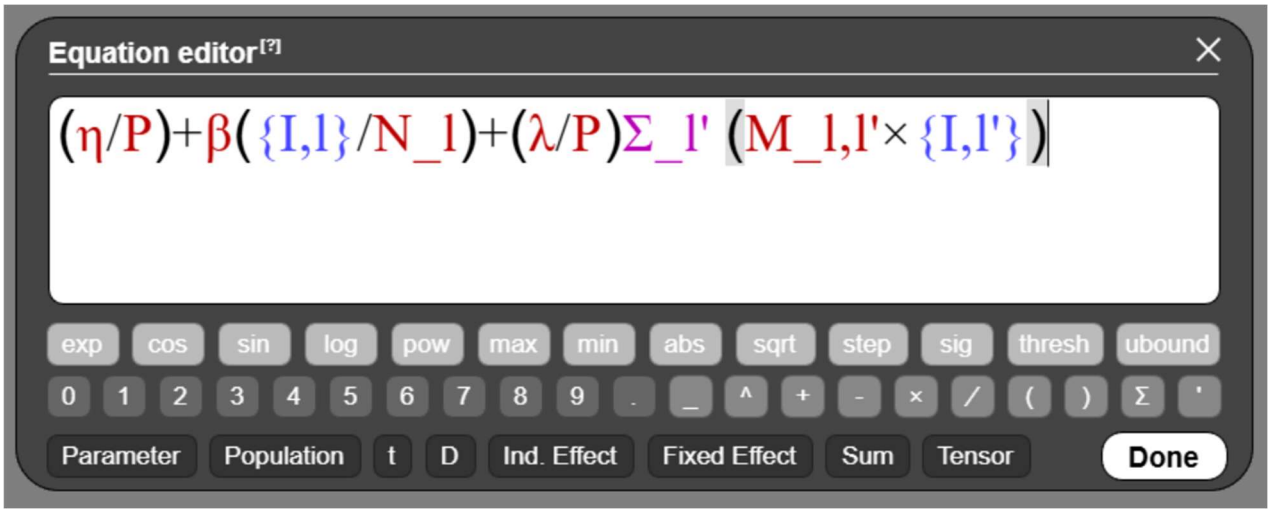


Figure 3 – *The equation editor*. Different options in this editor are described in §2.1.10. Parameters are shown in red, populations in blue, sums in purple, and grey shading indicates matching brackets.

2.1.10 Equations

Equations are used to represent many different quantities in BICI. Examples include the rate at which individuals pass down a transition, coefficients of variation for distributions, branching probabilities, and parameters related to the observation model.

A textbox is provided to enter equations, and for simple expressions this is more than adequate. Sometimes, however, equations become complicated, and for this reason an editor is incorporated into the software. This can be opened by clicking on the '⚄' icon.

Equations are potentially made up of the following elements:

- **Numbers** – Fixed numerical quantities. These can either be written as decimals (*e.g.* "10" or "2.34") or using scientific notation (*e.g.* "1.24e3" for 1240).
- **Parameters** – These are usually represented by Greek letters (*e.g.* β) or a single letter (*e.g.* a), but can, in principle, be any text[7]. Greek can be typed in Latex format and it automatically gets converted to the equivalent character (*e.g.* \alpha' changes to 'α').

  Parameters can depend upon classification indices by using an underscore (which makes a subscript when formatted). For example, 'β_a' becomes "β_a" when formatted, which means that **β** is a vector that depends on a compartment indexed by a (where a could, *e.g.*, represent different locations).

  Superscripts can also be added, *e.g.* 'β^Label' gets converted to β^{Label}. Note that superscript do not imply the parameter is being raised to a given power. They are, however, a useful way to differentiate parameters, *e.g.* such that β^{A} is treated differently to β^{B}. Indices and superscripts can both be used, but the superscript must always go first, *e.g.* 'β^Label_a' is fine but 'β_a^Label' is incorrect.

  The values for parameters must be set under model simulation, but can be estimated from data under inference (in which case they require a prior definition, see §4.2).
- **Populations** – These are denoted by curly brackets and represent the number of individuals within a sub-population (*e.g.* $\{\mathrm{I}\}$ may represent the total number of infected individuals or $\{\mathrm{S},\mathrm{M}\}$ may represent the number of susceptible males). Indexes can also be used in populations, *e.g.* $\{\mathrm{I},\mathrm{a}\}$ could represent the local infected population in location indexed by a. The total number of individuals can be either represented by {} or {all}.

  Individual or fixed effects can also be incorporated into populations as a way to incorporate individual variation (see §2.1.11).
- **Time** – The quantity 't' is reserved to represent time. This can either be explicitly used within an equation (*e.g.* 'a+b×cos(c×t)', which is a trigonometric function involving time t), or to indicate a time-varying parameter represented by a spline (*e.g.* $\eta(\mathrm{t})$).
- **Populations at specified times** – These represent a population at a specified time point (for reparameterised splines and derived quantities only). For example, $\{\mathrm{I};\mathrm{t}{=}20\}$ would be the number of infected individuals at time 20.
- **Operators** – Standard operators are used to combine together quantities: '*' for multiply (which gets converted to '×'), '/' for divide, '+' for addition and '-' for subtraction.
- **Sums** – These can be used to add up over compartments within a classification. For this purpose, a "primed" index is used. They have the format "For example the text input "\sum_a' (M_a,a'*{I,a'})" is interpreted as the mathematical expression $\sum_{a'}\left(M_{a,a'} \times \{I, a'\}\right)$. Here the matrix $M_{a,a'}$ mediates interactions between different areas and the primed location index a' is summed over.

[7] Note, they cannot contain a space or the following characters: "+-*×/0123456789.{<>⟨⟩}()|Σ∫'".

Sums have the format "\sum_index1',index2' (…)", *i.e.* they can be over multiple indices, but the content must always be encapsulated by round brackets. If the indices are the same, double primes can be used to differentiate, *e.g.* "\sum_index',index'' (…)".

To facilitate computation on spatial models, it is possible to restrict the range of sums. For example, "\sum_a'[a,50] (M_a,a'*{I,a'})", which gets converted to $\sum_{a'}^{\mathrm{d}(a',a)<50} M_{a,a'} \times \{I, a'\}$, only sums over terms where location a' is less than 50km from location a. This example assumes geographical coordinates. In Cartesian coordinates the Euclidean distance between compartments is restricted.

- **Time integral** – These can be used to integrate an expression over time (for derived quantities only). They have the format "\int dt (…)", where the content goes in the round brackets. For example, the expression "\int dt ({I})" is mathematically represented by $\int \{I\} dt$. By default, time is integrated over the entire system period, but limits can be placed (*e.g.* "\int[10,50] dt ({I})" becomes $\int_{10}^{50} \{I\} dt$.
  Time integration can be used to estimate the relative contribution to different routes of transmission. For example, if the force of infection is given by $\beta\{I\} + \phi$ the derived quantity "\int dt (\phi) / \int dt (\beta{I}+\phi)" (mathematically $\int \phi dt / \int (\beta\{I\} + \phi) dt$) gives the relative contribution of external infections).
- **Individual effects** – These are denoted by square brackets (*e.g.* $[g]$) and allow individual-based variation. Individual effects are log-normally distributed with a mean of one and a variance matrix that can be set or estimated. Additionally, they can be correlated between individuals (through some loadable $\mathbf{A}$ matrix), which allows for quantitative genetics models to be incorporated, or correlated with other individual effects. See §2.2.2 for details.
- **Fixed effects** – These are denoted by triangular brackets (*e.g.* $\langle w \rangle$) and allow for an individual property (*e.g.* weight) to be related to model transitions (*e.g.* it could be that lighter individuals become infected more quickly). To implement a fixed effect the covariate vector $\mathbf{X}$, which gives the property values for each individual in the system, must be loaded from a file. Fixed effects have a population mean of one and are strictly positive. A parameter ('μ' with superscript given by the fixed effect name, so μ^w in this case) controls the strength of the fixed effect, which is either set or inferred. See §2.2.3 for details.
- **Functions** – A selection of different functions can be added into equations:
  - *exp(x)* – The exponential function.
  - *log(x)* – The logarithm function.
  - *sin(x)* – The sine function.
  - *cos(x)* – The cosine function.
  - *step(x)* – The step function (if $x>0$ returns 1, else 0).
  - *sig(x)* – The sigmoidal function.
  - *pow(x|y)* – The power function x^y.
  - *max(x|y)* – The maximum function (if $x>y$ returns x, else returns y).
  - *min(x|y)* – The minimum function (if $x<y$ returns x, else returns y).
  - *abs(x)* – The absolute function (if $x>0$ returns x, else returns $-x$).
  - *sqrt(x)* – The square root function.
  - *thresh(x|y)* – The threshold function (if $x<y$ returns 0, else returns x).
  - *ubound(x,y)* – An upper bounded function (if $x>y$ returns ∞, else returns x).
- **Reserved parameters** – There are a number of special reserved parameters:

- *Distance matrix* $D_{a.a'}$ – Defined as D_a,a', this is set to the distance between compartments a and a'. In geographical coordinates this is given in kms and in Cartesian coordinates it represent the Euclidean distance between the coordinates of the two compartments.
- *Identity matrix* $\delta_{a,a'}$ – Defined as δ_a,a', this takes the value 1 when $a = a'$, otherwise zero. This matrix is often used such that a given rate only applies to individuals in a given state. For example, 'β×δ_s,M×{I}' would only apply when an individual's sex (indexed by s) is male (M).
- *Density vector* DEN_a^d – Defined as DEN^d_a, this gives the compartmental density, as measured using Kernel Density Estimation (KDE). This relies on a normally distributed spatial kernel with a specified radius d. In Cartesian coordinates, this radius is given by the Euclidean distance and in geographical coordinates (latitude and longitude) it is measured in kilometres.
- *Relative density vector* $RDEN_a^d$ – This is the same as DEN_a^d but scaled such that the average across all compartments is one.

- **Derived functions** – BICI provides a number of standard functions for useful derived quantities. For example, in the epidemiological setting these can be used to estimate the reproduction number and generation time using a number of approaches (see Appendix D for details).

2.1.11 Individual variation in populations

Usually, populations in equations represent the number of individuals in a given set of specified compartments. For example {I} represents the total number of individuals in the 'I' compartment. In some instances, however, it is useful for individuals to differ in their contribution to this 'population'. Supposing that the compartment 'I' represents infected individuals, it could be that some individuals are more infectious than others. Here we want the 'population' to represent the overall level of infectivity rather than just the number of infected individuals.

To incorporate this individual-level variation, it is possible to add individual and/or fixed effects into the population (note, this is possible for individual-based models only). The following syntax is used '{s ; v}' where:

Compartmental specification s –Determines the group of individuals under study. It follows the same rules as for an ordinary population, for example it can take values 'I', 'E|I' or 'I,g' (where g is an index).

Individual variation v – Accounts for individual variation in contributions to the population. This consists of one or more fixed or individual effects multiplied together.

An example would be {I ; <f>}. This 'population' is mathematically equivalent to $\sum_{i \in I} \langle f \rangle_i$, where the sum i goes over all individuals in 'I' compartment.

Note, because both individual and fixed effects are designed to have a population average of one, so the overall individual variation term v is expected to be one on average[8]. This mean that the sum would be expected to add to something similar to the raw population size. In other words, when variation across individuals is small so {s ; v} ≈ {s}.

2.1.12 Naming restrictions

To avoid misunderstanding the following naming conventions must be followed:

[8] At least ignoring potential correlations.

Species and classification names – Cannot be more than 40 characters. Cannot contain the following characters “|"*×{}<>⟨⟩}[]()=~’→;$&#\Σʃ_^” or any spaces. Furthermore, the following reserved words cannot be used: “compartment”, “population”, “alpha”, “distribution” or “file”.

Compartment names – As above, but superscripts are allowed, *e.g.* ‘I^cow’ becomes I^{cow}, to differentiate between different compartments.

Parameters – As above, but superscripts and subscripts (for any dependent indices) are allowed, *e.g.* ‘\beta^cow_a’ becomes β_a^{cow}.

Classification index name – Can only be a single character from the lowercase alphabet (cannot be ‘t’, which is reserved to represent time variation or ‘z’, which is for indices on covariance matrices).

Individual and fixed effect names – As above, with no superscript or subscript allowed.

Individual names – There are no restriction on individual name, apart from the fact that they cannot contain any spaces.

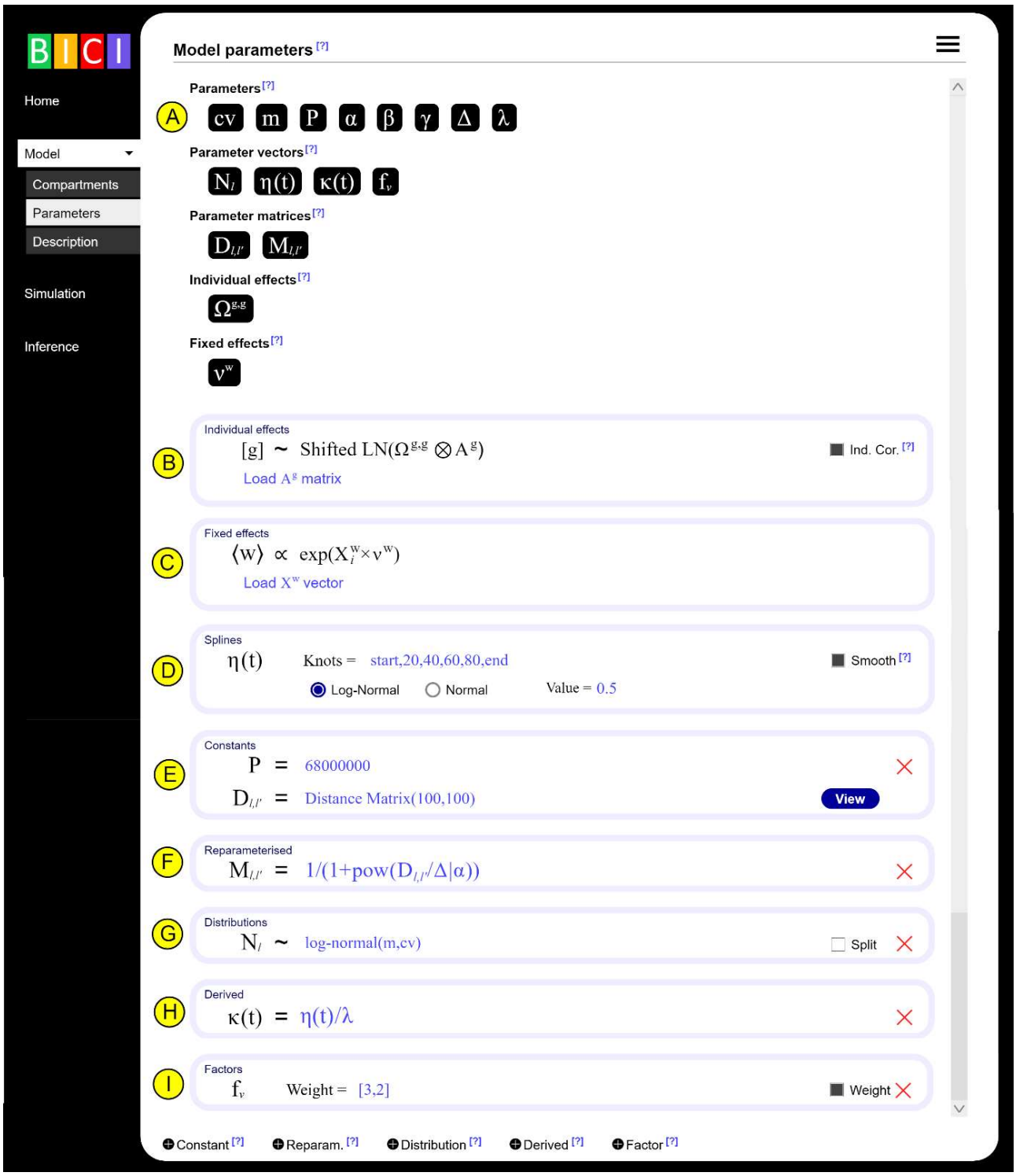


Figure 4 – *Model parameters*. This shows how different parameters are treated in the model. A: All model parameters, separated into categories. B: Individual effects, C: Fixed effects, D: Splines, E: Constants, F: Reparameterisations, G: Distributions, H: Derived, I: Factors.

2.2 Parameters

The "Model→Parameters" page in Fig. 4 summarises how parameters are treated in the model. The following sections describe different features of this page:

2.2.1 Parameter summary

Figure 4A shows a list of all model parameters, ordered by category. First there are univariate parameters, followed by vectors and matrices (and more generally tensors). Following this are parameters associated with variances and correlations for any individual effects in the model. Finally, parameters for fixed effects are shown. Clicking on these buttons gives a brief description of the parameter and allows for the user to jump to the part of the model in which the parameter appears.

2.2.2 Individual effects

In Fig. 4B we see that an individual effect [$\boldsymbol{g}$] has been added to the model. This represents a vector with a value for each individual in the system. Individual effects are assumed to be a sampled from the following shifted log-normal distribution:

$$[\mathbf{g}] = \text{Shifted LN}(\boldsymbol{\Omega}^{\text{gen}} \otimes \mathbf{A}^{\text{gen}}) \qquad (1)$$

Here the user-defined label "gen" refers to the fact that this individual effect comes from a genetic effect. The "shifted" notation corresponds to the mean on the log-scale deviating from zero to ensure that the population average of the individual effect is one (this makes sense because individual effects are used to investigate variation across individuals, not changes in this mean).

The covariance in Eq.(1) comes from the tensor product of two parts: The first part relates to the covariance of the trait itself. In this particular case it is a scalar Ω^{gen}, but when more than one individual effect is included, it becomes a matrix[9]. BICI uses the convention whereby diagonal elements of $\boldsymbol{\Omega}$ represent variances and the upper right-hand corner of the matrix stores correlations between individual effects[10].

The second part in Eq.(1) relates to correlations in individual effects between individuals. These are optional, and included by selected the "Ind. Cor." checkbox (Fig. 4B). Such correlations are a useful way to incorporate quantitative genetics into models (such that related individuals are more likely to share similar values for their individual effect[11]).

If selected, it is necessary to load up an $\mathbf{A}$ matrix. This can either be done directly, or from a pedigree or by loading the inverse of the $\mathbf{A}$ matrix:

A MATRIX REQUIRES – A data table with headings given by the individual IDs and a square matrix of elements that defines the relationships between these individuals (note, $\mathbf{A}$ can also include IDs of individuals who do not appear in the compartmental model itself, and individual effect estimates can be made for these). Such a matrix is commonly used in quantitative genetic analysis and can be derived either from pedigree information or genetic data. Typically, the $\mathbf{A}$ matrix will be 1 along its diagonal, 0.5 for a

[9] *E.g.,* if we imagine two individual effects [$\boldsymbol{g}$] and [$\boldsymbol{f}$], a 2×2 covariance matrix $\boldsymbol{\Omega}^{f,g}$ would be defined.
[10] Correlations, which go between -1 and 1, tend to be easier to interpret than covariances.
[11] *E.g.*, if a parent is more susceptible than average, it might be likely that its offspring would also be more susceptible.

parent/sibling relationship, 0.5 for full-sibs, 0.25 for half-sibs, and become increasingly small for more and more distant relatives.

EXAMPLE – This shows an example of an input .csv file (viewed in Excel) which loads up the relationship matrix for 4 individuals (here 'Ind-1' is the parent of 'Ind-2' and 'Ind-3' and 'Ind-4' are half-sibs):

	A	B	C	D	E	F	G	H
1	Ind-1	Ind-2	Ind-3	Ind-4				
2	1	0.5	0	0				
3	0.5	1	0	0				
4	0	0	1	0.25				
5	0	0	0.25	1				

PEDIGREE REQUIRES – A data table containing at least the following columns: "ID" a unique identifier for individuals, and "sire" and "dam" which give the names of the male and female parents (".” is used if a parent is not in the data-set). As above, the pedigree can include IDs of individuals who do not appear in the compartmental model itself.

EXAMPLE – This shows a pedigree with 4 individuals ('Ind-1' and 'Ind-2' are parents of 'Ind-3' and 'Ind-4'):

	A	B	C	D	E	F	G	H
1	ID	sire	dam					
2	Ind-1	.	.					
3	Ind-2	.	.					
4	Ind-3	Ind-1	Ind-2					
5	Ind-4	Ind-1	Ind-2					

INVERSE **A** MATRIX REQUIRES – A data table with headings given by the individual IDs and a square matrix of elements that defines the inverse relationship matrix.

2.2.3 Fixed effects

Fixed effects allow for individual quantities to be incorporated into the model and act as covariates that can modify transition rates, means *etc...* Fixed effects are indicated in equations by triangular brackets, *e.g.* $\langle \mathbf{w} \rangle$. This represents a vector with a value for each individual in the system calculated according to:

$$\langle \mathbf{w} \rangle = ce^{\mathbf{X}^w \mu^w} \qquad (2)$$

The constant of proportionality c is chosen such that $\langle \mathbf{w} \rangle$ has a population-wide average of one. The vector $\mathbf{X}^w$ stores individual information (*e.g.* weight), which acts as a covariate. The strength of the interaction is governed by fixed effect parameter $v\mu^w$. If this is zero, it means that $\langle \mathbf{w} \rangle$ is one across all individuals. If it is non-zero, in means that $\langle \mathbf{w} \rangle$ varies across individuals and so relates whatever equation $\langle \mathbf{w} \rangle$ is in with the covariate.

Incorporating a fixed effect means that we must load the covariate vector $\mathbf{X}^w$. This is done by clicking on the "Load $\mathbf{X}^w$vector" in Fig. 4C.

REQUIRES – A data table containing at least the following columns: "ID" a unique identifier for individuals and "Value" stores the individual covariate.

EXAMPLE – This example input .csv file contains the log of the weights of 4 individuals:

	A	B	C	D	E	F	G	H
1	ID	Value						
2	Ind-1	0.5						
3	Ind-2	2.6						
4	Ind-3	-0.4						
5	Ind-4	0.9						

Note, logs of quantities (rather than the quantity itself) are often used for fixed effects. This is for three reasons: (1) They transform something which is positive into something that can potentially be positive or negative (and often has a closer resemblance to a normal distribution), (2) logs can tackle quantities which vary over a large range (*i.e.* many orders of magnitude), and (3) because of the log relationship $\mathbf{X}^w = \log(\boldsymbol{Q})$, so Eq.(2) become $\langle \mathbf{w} \rangle = c\mathbf{Q}^{\mu^w}$, *i.e.* the fixed effect has a simple power-law relationship with the quantity.

2.2.4 Splines

Splines are used to account for time variation in model parameters, and are indicated by placing (t) after the parameter definition (Fig. 4D). The values of the spline are specified at a number of key time points, otherwise known as "knots". Values at intermediate time points are interpolated (using a number of different options, as discussed below).

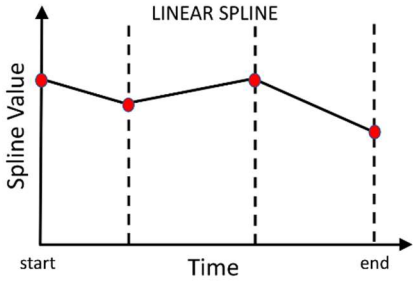
Splines can be used in three different contexts:

(1) Time variation in a dynamic process (*e.g.*, a time-varying transmission rate $\beta(t)$).
(2) Incorporating time-varying covariates (*e.g.*, to investigate how disease transmission depends on monthly temperature records $T(t)$). In this case the spline would comprise of constant data that would be loaded from a file.
(3) Time variation in an underlying process which is defined through reparameterisation (see below). In this case the spline would comprise of equations for each knot time that would be loaded from a file.

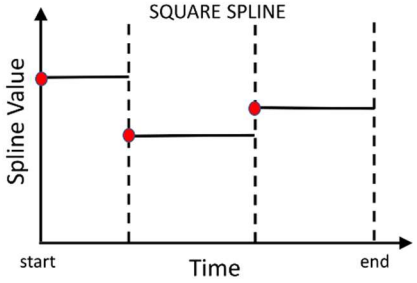
Different spline types

Splines can be defined to have one of four types:

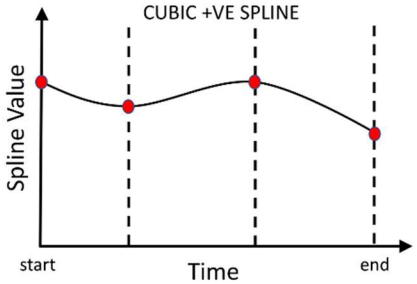
- **Linear** –All intermediate points are linearly extrapolated. This is otherwise known as "piecewise linear". The diagram on the right shows four knots (red circles), with knot times indicated by the vertical dashed lines (one at the start, one at the end, with two intermediate points). The solid black line shows the spline itself, which passes through the knot points.



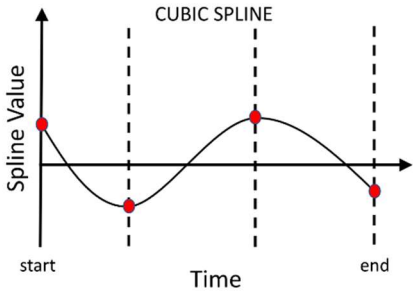
- **Square** – This is flat between the knot times. The value is specified using the knot time at the start of each interval. Note, one fewer spline value is used to specify this type of spline.



- **Cubic +ve** – This fits a cubic spline to the log of the parameter value (hence this spline is guaranteed to be positive, provided the values at the knots times are positive).



- **Cubic** – This fits a cubic spline to the parameter values at the knot times (note, this spline is not guaranteed to be positive).



The two types of cubic spline fits piecewise polynomials (up to a cubic term) that interpolate between the knot times. They are defined to be continuous up to second derivative, so they have a much smoother appearance than the linear spline. However, because a change in a single knot value changes the entire spline, so performing inference using cubic splines is generally slower than for the linear or square counterparts.

The knot times can be arbitrarily set, although they must be monotonically increasing (sharp increases or decreases for a linear spline value can be accommodated by having a very short time period, *e.g.* a single time-step, between successive knots).

Clicking on the "Knots" value (Fig. 4D) opens a text box used to set the knot times (comma separated). Optionally, the special key words 'start' and 'end' can be used to represent the overall time range for simulation or inference. Intermediate points can be added, *e.g.* in the example in Fig. 4D the knots are set

to ‘start,20,40,60,80,end’ which means that values for the parameter are defined at the start, at times 20, 40, 60, 80 and then at the end.

Large number of spline values can be loaded from a file (see §2.2.10 for details). Furthermore, vector, matrix or tensor elements can also be splines (so $T_l(t)$ could store temperature variation at different locations l, or $M_{a,a'}(t)$ could describe a time-varying age-contact matrix).

Spline smoothing

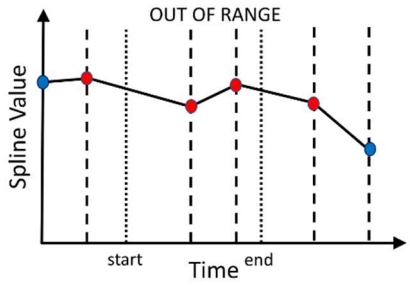
Smoothing can be added in cases in which the spline is being directly inferred (which is often done with linear splines because they don’t inherently incorporate any smoothing). This is used to represent the fact that parameter values might not be expected to jump drastically between consecutive knots, but rather undergo some smooth change. Smoothing can be applied in two possible ways:

- **Normal** – The expected distribution of the parameter value at one knot on the spline is expected to be normally distributed with respect to the value at the previous knot. This approach is appropriate when the parameter can be *negative as well as positive*.
- **log-normal** – The expected distribution in the log of the parameter value at one knot on the spline is expected to be normally distributed with respect to the log of the value at the previous knot. This approach is appropriate when the parameter is *strictly positive*.

In both cases, the “Value” can be set, which specifies the standard deviation in the normal (a smaller value implies more stringent smoothing).

Spline time range

Splines can be defined outside of the time range specified by the simulation or inference. In the example on the right, the knot times span from before the simulation start time to after the simulation end time. In these instances, the spline knot immediately preceding the start time and those after the end time, along with any intermediate knots, are used to defined the spline values (these knots are shown in red). Other knots are ignored (shown in blue).



Reparameterised splines

In some circumstances the time variation in a quantity is functionally dependant on other system properties. For example, one way to model a farm-based trading network is to introduce a spline that represents the probability a farm becomes infected on a daily basis. This probability is not directly known, but it is functionally related to movements of animals between farms (which are known) and the infection status of other farms (e.g., see Ex. F5). Such a model is achieved by reparametrizing the spline. This involves loading up equations that determine the spline values at the different time points from a file. If equations involve time varying quantities, such as populations or time-varying parameters, the spline must be defined to be square. For reparameterised square splines, populations at previous times can also be used in equations. For example the equation ‘α×{I;t=40}’ can be used to represent the spline value between times 100 and 200 (note, the population time t=40 lies before the 100).

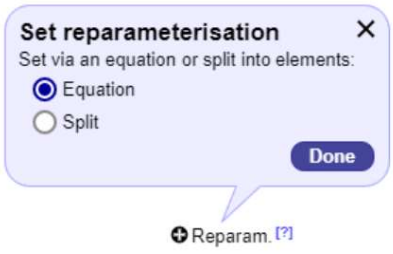
2.2.5 Constants

Parameters (including splines) can be set as constant by clicking “⊕ Constant”. A constant takes that value regardless of whether simulation or inference is performed. If a parameter is only set for the purposes of simulation, this should be done under the “Simulation→Parameter” page. In the example in Fig. 4E, the parameter P (which represents a population) is set to a fixed value. The distance $D_{l,l'}$ and identity $\delta_{l,l'}$ matrices are, by definition, constant.

Constants with a large number of elements can be loaded from a file (see §2.2.10 for details).

2.2.6 Reparameterisations

Reparameterisation expresses one parameter (vector/matrix/tensor) in terms other parameters. It is typically used as a way of reducing the overall number of parameters to be estimated[12].



Setting a parameter to be reparameterised can be done by clicking "⊕ Reparam". Two options are available:

- **Equation** – Here the parameter is written in terms of an equation. In the example in Fig. 4F, the contact matrix across locations is reparameterised in terms of a power-law spatial kernel. If the parameter contains any dependencies, these can also be reflected in the equation (in the example the indices l and l' appear in both the left and right-hand-sides of the equation).
- **Split** – In this case each element of the parameter is defined by a separate univariate equation. Such equations can be loaded from a file:
  REQUIRES – A data table containing columns with headings given by the indices of the tensor and a final "Equation" column (see §2.1.10 for details on formatting equations and §2.2.10 for file formatting).

Note, although equation reparameterisation doesn't typically increase code execution time[13], it does increase the memory usage. If memory is a limiting factor, substituting the reparameterised equation may be advisable[14].

Furthermore, in the current version of BICI reparameterisation cannot involve time-varying quantities, including splines, populations or explicit use of the time variable t.

2.2.7 Distributions

Setting elements of a parameter vector / matrix / tensor to be sampled from a distribution is similar to specifying a prior (and uses the same set of possibilities). Here, however, new parameters are introduced that characterise the distribution (*e.g.* a mean, a standard deviation etc...). These go on to have their own prior, incorporating a so-called "hierarchical" model. A classic example is that of a random effect, where for different groups a parameter value is assumed to be sampled from a normal distribution (whose standard deviations may be estimated from the data).

Distributions for parameters can be set by clicking "⊕ Distribution". The example in Fig. 4G shows that parameter vector N_l is sampled from a log-normal distribution with mean m and coefficient of variation cv.

A parameter could have one sampling distribution for all elements, or separate distributions for each element. In the latter case, the "Split" checkbox is selected (Fig. 4G). If there are many elements to set then it may be convenient to load them from a file (see §2.2.10 for details). This is done using the same format

[12] An example of this is when a contact matrix $M_{l,l'}$ of interactions, which contains L^2 elements where there are L geographical locations, is reparameterised as a power-law spatial kernel, which only contains three parameters.
[13] In many cases they make BICI run faster.
[14] *E.g.*, in example M3.3 the spatial kernel matrix "$1/(1+\mathrm{pow}(\mathrm{D}_{l,l'}/\Delta|\alpha))$" can be placed in the equation for the transition rate (instead of matrix $\mathrm{M}_{l,l'}$).

as for priors in §4.2.1, except that here the distribution quantities (means, standard deviations, *etc…*) can be set as parameters rather than numerical quantities.

2.2.8 Derived

A derived quantity is functionally related to other model parameters through an equation (Fig. 4H). Such a quantity usually has some physical meaning (*e.g.* the reproduction number R_0 in an epidemic model, which is related to the transmission and recovery rates).

The dependency of the derived quantity must be the same as that of the equation. For example, "f=M_a×{I}" would not be valid because M_a is a vector that depends on classification with index a and the population {I} is time dependent. In this case the correct definition would be "f_a(t)=M_a×{I}".

Derived quantities can involve time integration, and this can, for example, be a useful way to estimate the relative size of different routes of infection (see §2.1.10). Derived quantities can also involve populations at a specified time point, *e.g.* "g=2×{I;t=50}" would give twice the value of the infected population at time t=50.

2.2.9 Factors

A 'factor' is a parameter that depend on a classification, but is constrained to have a mean value of exactly one.

For example, the expression βf_s could be used to represent the infections rate for two sexes, indexed by s. The normal transmission rate parameter β governs the overall rate of transmission and the factor f_s allow for the two sexes to have a different susceptibility (*e.g.* $f_M = 1.1$ for males an $f_F = 0.9$ for females, such that the average is zero).

In some circumstances the accuracy with which different elements of a factor can be estimated may be different. Here a weight w_i can be assigned to each element i such that the weighted average for the factor is one:

$$\sum_i (f_i w_i) \Big/ \sum_i w_i = 1. \qquad (3)$$

2.2.10 Loading from files

If a parameter requires large number of elements to be specified this can be done using a file. This is usually for constant values from data (*e.g.* time series covariate data), but can also be done for the purposes of setting large numbers of distributions, priors or reparameterisation equations.

To load the elements first click on the blue value (*e.g.* see Fig. 4) to open up a parameter editor and then the "⊕ Load" button in the bottom left-hand corner.

REQUIRES – A data table containing columns with headings given by the indices of the parameter (and 't' if time-varying). When loading constants (or parameter values for simulation) the final column has heading "Value". This column gives the numerical value of the vector element. Different rows relate to different elements in the parameter (and they can be provided in any order). Elements are assumed to have a value of zero if not specified in the table.

The final column has heading “Prior”, “Distribution” or “Equation” in the cases in which the prior, distribution or reparameterisation, respectively, are being specified separately for each parameter element.

VECTOR EXAMPLE – The values of a constant parameter x_a could be loaded with the following example (where the classification with index a contains the compartments ‘A’, ‘B’ and ‘C’):

	A	B	C	D	E	F	G	H
1	a	Value						
2	A	1.2						
3	B	0.2						
4	C	-2.3						

SPLINE EXAMPLE – The values of a constant spline $x(t)$ could be loaded with the following (where the values of ‘t’ must correspond to the knot times specified for the parameter spline):

	A	B	C	D	E	F	G	H
1	t	Value						
2	start	4.5						
3	40	6.1						
4	end	-9.8						

TIME-VARYING VECTOR EXAMPLE – The values of a constant vector spline $x_a(t)$ could be loaded with the following (where the classification with index a contains compartments ‘A’ and ‘B’ and the parameter has knot times ‘start’, ‘40’ and ‘end’):

	A	B	C	D	E	F	G	H
1	a	t	Value					
2	A	start	1.4					
3	A	40	2.6					
4	A	end	5.5					
5	B	start	0.5					
6	B	40	1.2					
7	B	end	1.5					

TENSOR EXAMPLE – The values of a constant tensor $x_{a,a',b}$ could be loaded with the following (where the classification with index a contains compartments ‘A’ and ‘B’ and the classification with index b contains compartments ‘X’ and ‘Y’):

	A	B	C	D	E	F	G	H
1	a	a'	b	Value				
2	A	A	X	1.1				
3	A	A	Y	-0.4				
4	A	B	X	2.4				
5	A	B	Y	2.1				
6	B	A	X	2.2				
7	B	A	Y	1.3				
8	B	B	X	0.4				
9	B	B	Y	0.5				

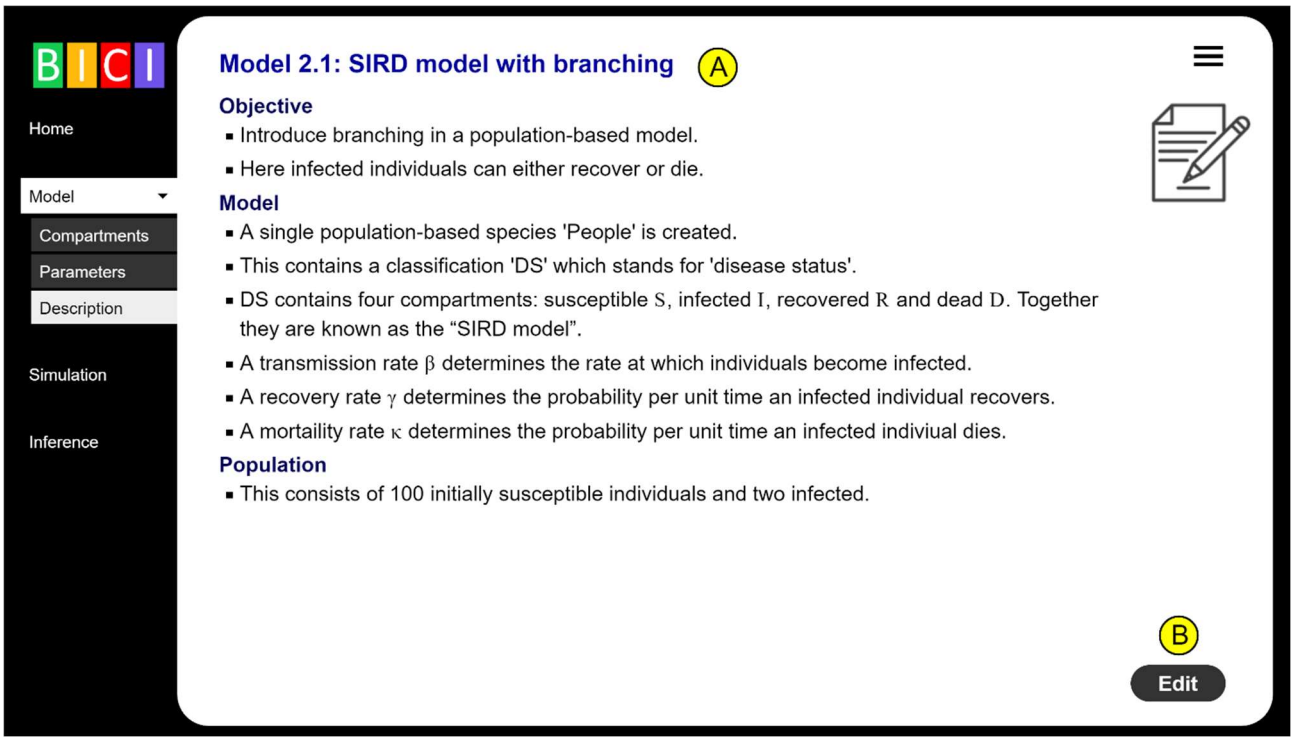


Figure 5 – *Description*. A: Text panel giving a description of the data, model, and analysis (this example taken from M2.1 in §8), B: Edit this description.

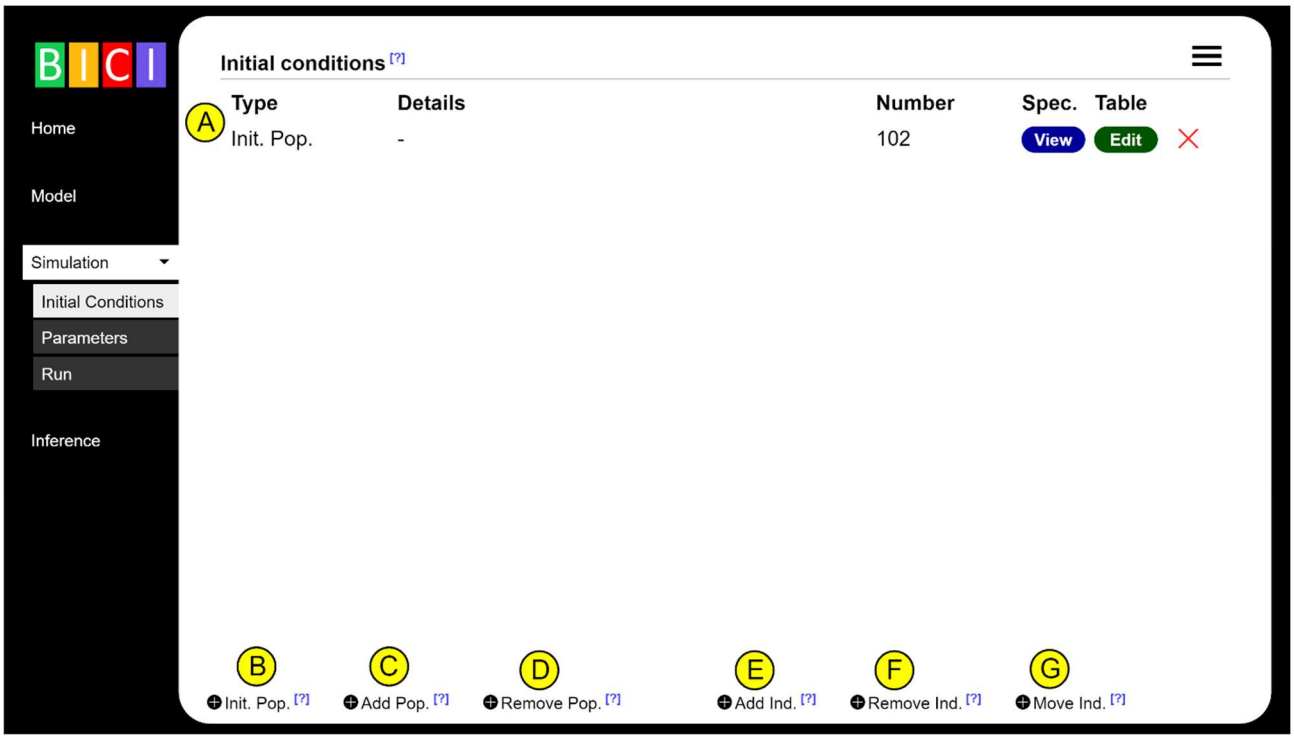


Figure 6 – *Initial conditions*. This page is used to load initial conditions (as well as any addition or removal of individuals after the start time). A: Table showing loaded data, B: Add information about the initial population, C: Add populations during the simulation, D: Remove populations during the simulation, E: Add individuals to the system, F: Remove individuals from the system, G: Move individuals within the system. Note, C and D are only available for population-based models, and E-G are only available for individual-based models.

2.3 Description

Figure 5 shows the "Model→Description" page that allows users to provide a brief description of the model and any analysis performed. This is not only useful to keep track for personal use, but also makes it easier and more transparent for others to understand what has been done. The description can be edited by clicking on the "Edit" button. A markdown format is used that allows for the following concise formatting options: "# Title", "## Subtitle", "- Bullet point", "*italic text*" and "**bold text**". Parameters can be added by enclosing in dollar symbols, *e.g.* "a" or "b^super_sub".

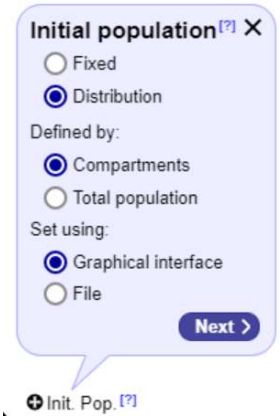
3) Simulation

Simulation relies on specification of model parameters, along with the initial conditions, and stochastically predicts the dynamics of the compartmental model over a period of time.

3.1 Initial conditions

Figure 6 shows the page where initial conditions are loaded (as well as any individuals or populations externally added, moved or removed after the start time). The table at Fig. 6A shows all loaded information, which can be viewed, edited or deleted. The lower menu bar allows the user to add new data. Below we list the various possibilities:

3.1.1 The initial population



The initial population defines the number of individuals in different compartments at the start time. For population-based models, this must be specified (or if omitted it implies the initial population is zero). For individual-based models, an alternative is to leave the initial population unspecified and add known individuals at the start time (see §3.1.4).

Clicking "⊕ Init. Pop." (Fig. 6B) allows for the initial population to be specified. This can be defined in one of two different ways:

(1) Fixed – The populations in the initial state are defined and fixed.

When there is only one classification, each of the compartmental populations is simply specified. However, when there is more than one classification, two possibilities exist:

- *Focal* – Here a "focal" classification is selected and populations in different compartments are defined for this. For other classifications the percentages of individuals in different compartments are set (note, one compartment is left unspecified because it is automatically calculated such that the total adds up to 100%). This method is approximate in the sense that it relies on uncorrelated distributions across different classifications.
- *All* – The populations are specified for all possible combinations of compartments across all classifications. This method exactly specifies the initial population.

The population specification is set using:

- *Graphical interface* – Here the point-and-click interface is used to specify populations and/or percentages in the text boxes provided.
- *File* – A file specifying the initial populations:
  FOCAL REQUIRES – A data table containing columns "Compartment" and "Population". The "Compartment" column must go through all compartments in the focal classification and all but

one for the other classifications (the remaining is automatically calculated such that percentages/fractions add up to 100%). In the case of the focal classification, the "Population" column gives the number of individuals in each compartment. Other classifications are represented by percentages.

EXAMPLE – For a model containing disease status (here the focal classification) and sex classifications, this file loads up a population of 100 individuals into the 'S' compartment, 1 into the 'I' compartment and none into the 'R' compartment. 49% of individuals are taken to be male (and 51% female, which is automatically calculated).

	A	B	C	D	E	F	G	H
1	Compartment	Population						
2	S	100						
3	I	1						
4	R	0						
5	M	49%						

ALL REQUIRES – A data table containing columns with headings given by the classification names along with "Population", which gives the initial number of individuals within each compartmental combination.

EXAMPLE – For the model above, this sets populations individually:

	A	B	C	D	E	F	G	H
2	S	M	49					
3	S	F	51					
4	I	M	1					
5	I	F	0					
6	R	M	0					
7	R	F	0					

(2) Distribution – The initial population is assumed to be sampled from a distribution. This is set in the following way:

- *Focal* – A population distribution is specified for each compartment in the "focal" classification. For other classifications a Dirichlet distribution is set to give the probability of being in different compartments (this is defined by alpha values α_c for each compartment c such that the probability of being in c is $\alpha_c / \sum_d \alpha_d$).
- *All* – A population distribution is placed on the overall initial population N^{total}, and a Dirichlet distribution is specified for all possible combinations of compartments across all classifications.

The distribution specification is set using:

- *Graphical interface* – The point-and-click interface is used to specify distributions and α values in the text boxes provided.
- *File* – A file specifying the initial distribution:

  FOCAL REQUIRES – A data table containing columns "Compartment" and "Distribution". The "Compartment" column must go through all compartment in the model. In the case of the focal classification, the "Distribution" column gives the distribution the population number is sampled

from, *e.g.* "uniform(10,20)" (see below for formatting). Other classifications are represented by α values used to defined the Dirichlet distributions.

EXAMPLE – Sex-based SIR model with disease status being the focal classification (note, the Dirichlet prior on sex implies that there are twice as many males as females):

	A	B	C	D	E	F	G
1	Compartment	Distribution					
2	S	uniform(100,200)					
3	I	exp(10)					
4	R	fix(0)					
5	M	2					
6	F	1					

ALL REQUIRES – A data table containing columns for each classification and a final column with "Alpha". This column defines a Dirichlet distribution across all possible compartment combinations. The distribution for the overall population number must be set once the file is loaded.

EXAMPLE – Sex-based SIR model with a specified Dirichlet distribution. Here the probability of being a susceptible male is $10/(10+15+1+1+0.1+0.1) = 37\%$. This is used in conjunction with an overall distribution in the total number of individuals, *e.g.* $N^{total} \sim \text{uniform}(100,300)$:

	A	B	C	D	E	F	G	H
1	DS	Sex	Alpha					
2	S	M	10					
3	S	F	15					
4	I	M	1					
5	I	F	1					
6	R	M	0.1					
7	R	F	0.1					

Since population distributions must be a strictly positive, they can be selected from the following four possibilities:

- *"uniform(min,max)"* – A flat prior distribution between a minimum and maximum value.
- *"exp(mean)"* – An exponential distribution with a specified mean.
- *"gamma(mean,cv)"* – A gamma distribution with a specified mean and coefficient of variation.
- *"log-normal(mean,cv)"* – A log-normal distribution with a specified mean and coefficient of variation.

3.1.2 Adding populations to the system

Clicking "⊕ Add Pop." (Fig. 6C) allows for data on any additional populations that are added to the system after the start time. Note, this is for population-based models only.

REQUIRES – A data table containing at least the following columns: "t" the time the individuals are added, columns for each classification to denote into which compartment the individuals enter the system and "Population", which gives the number of individuals added.

EXAMPLE – In this example 10 susceptible males are added at time 10, 20 susceptible females at time 20 and infected individuals are added at time 30:

	A	B	C	D	E	F	G	H
1	t	DS	Sex	Population				
2	10	S	M	10				
3	20	S	F	20				
4	30	I	M	2				
5	30	I	F	2				

3.1.3 Removing populations from the system

Clicking "⊕ Remove Pop." (Fig. 6D) allows for data on any populations that are removed from the system after the start time. Note, this is for population-based models only.

REQUIRES – A data table containing at least the following columns: "t" the time the individuals are removed, columns for each classification to denote from which compartment the individuals leave the system and "Population", which gives the number of individuals removed.

EXAMPLE – Here individuals are removed at times 40 and 100:

	A	B	C	D	E	F	G	H
1	t	DS	Sex	Population				
2	40	S	M	15				
3	40	S	F	15				
4	100	I	M	10				
5	100	I	F	2				

3.1.4 Adding individuals

Clicking "⊕ Add Ind." (Fig. 6E) allows for data on any individuals added to the system. These individual can either be added at the start or at any later time.

REQUIRES – A data table containing at least the following columns: "ID" a unique identifier for individuals, "t" the time individuals are added, and columns for each classification to denote into which compartment the individuals enter the system. These compartmental specifications can be exact, *e.g.* 'S', or reflect any potential uncertainty (see §4.1.2.1 for details on how this is implemented).

EXAMPLE – This adds five individuals to the system at different time points ('Ind-3' has an equal probability of being in the 'I' or 'R' compartments, 'Ind-4' has a 40%/60% probability of being in 'I' or 'R', respectively, and 'Ind-5' has compartmental probabilities parameterised by a):

	A	B	C	D	E	F	G	H
1	ID	t	DS	Sex				
2	Ind-1	0	S	M				
3	Ind-2	0	S	M				
4	Ind-3	10	I|R	F				
5	Ind-4	100	I:0.4|R:0.6	F				
6	Ind-5	100	I:a | R:1-a	F				

The table above illustrates a case when there is uncertainty in the initial state of individuals. An alternative for an individual-based model is to specify a population distribution for the initial conditions and leave the initial state of the individuals unspecified (denoted by '.'). This is illustrated in example A5 in §8.

3.1.5 Removing individuals

Clicking "⊕ Remove Ind." (Fig. 6F) allows for data on any individuals removed from the system.

REQUIRES – A data table containing at least the following columns: "ID" a unique identifier for individuals and "t" the times individuals are removed.

EXAMPLE – This removes five individuals from system at different time points:

	A	B	C	D	E	F	G	H
1	ID	t						
2	Ind-1	200						
3	Ind-2	250						
4	Ind-3	150						
5	Ind-4	200						
6	Ind-5	200						

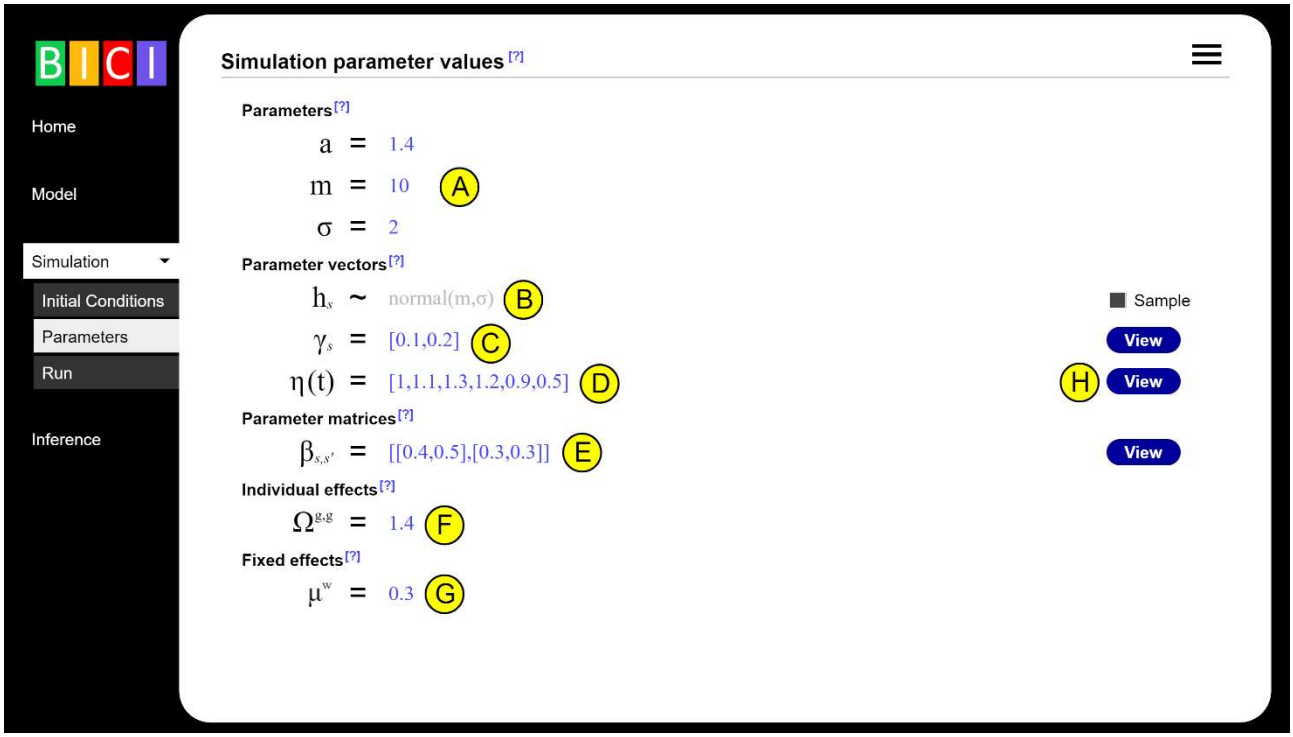


Figure 7 – *Simulation parameter values*. This page is used to specify parameter values before simulation can be performed. A: Univariate parameters, B: Distributions, C: Vectors, D: Splines, E: Matrices, F: Individual effect parameters, G: Fixed effect parameters, H: View the parameter definition.

3.1.6 Moving individuals

This moves individuals to specified compartments in a specified classification at specified times (note, this movement is enforced, and not as a result of an induced change, as would be modelled by a transition). For

example, this could be used to represent the movement of animals within a disease transmission experiment.

REQUIRES – A data table containing at least the following columns: "ID" a unique identifier for individuals, "t" the time individuals are moved and the name of the classification in which the individuals are moved (this column gives the destination compartment).

EXAMPLE – In this example the movement is made in the classification 'Location'. At time 100 three individuals are move to location 'B' and then at time 200 they are move to location 'A'.

	A	B	C	D	E	F	G	H
1	ID	t	Location					
2	Ind-1	100	B					
3	Ind-2	100	B					
4	Ind-3	100	B					
5	Ind-1	200	A					
6	Ind-2	200	A					
7	Ind-3	200	A					

Currently individual move data cannot be used on classifications that contain non-Markovian transitions.

3.2 Parameters

Figure 7 illustrates the "Simulation→Parameters" page. This page shows all the parameters that need to be specified before the model can be simulated. They fall into a variety of different categories:

- **Univariate** (Fig. 7A) – Clicking on the blue values opens up a text box that can be edited.
- **Distributions** (Fig. 7B) – If a parameter has been defined as coming from a distribution (on the "Model→Parameters" page), the "Sample" checkbox allows the user to select whether or not to directly set the parameter values, or sample as part of the simulation procedure.
- **Vectors** (Fig. 7C) – Elements of a vector depend on different compartments within a classification (the subscript on the vector corresponds to the classification index). Clicking on the blue value opens up an editor. Values for large vectors can be loaded from a file:

  REQUIRES – A data table containing a column heading given by the index of the vector and "Value", which givens the numerical value of the vector element (see §2.2.10 for details).
- **Splines** (Fig. 7D) – These are treated like vectors, but here elements correspond to the knot times specified on the "Model→Parameters" page. Values for splines with many knots can be loaded from a file:

  REQUIRES – A data table containing column headings "t" and "Value". The "t" column contains a list of the knot time, *e.g.* 'start', 20, …, 'end' (see §2.2.10 for details).
- **Matrices** (Fig. 7E) – These are quantities that depend on compartments in two classifications (which could be the same, as here, or different). Values for large matrices can be loaded from a file:

  REQUIRES – A data table containing column headings given by the indexes of the matrix and "Value" (see §2.2.10 for details).

- **Individual effect covariance matrices** (Fig. 7F) – Here variances and correlations are specified that determine the variation in individual effect across the population (see §2.2.2 for details). The matrix which specifies these is given by "Ω" with superscript defined in the "Model→Parameters" section. When there is just a single individual effect, Ω contains a single element for the variance. For more than one individual effect, the diagonals give the variances, the upper right-hand corner of the matrix gives the correlations between the individual effects and the lower left-hand corner has default elements containing '.'.
- **Fixed effect parameters** (Fig. 7G) – The strength of a fixed effect is determined by a parameter with name μ and superscript given by the fixed effect name, w in this case (see §2.2.3 for details).

Once parameters have been specified, a number of different visualisations can be made by clicking on the "View" buttons (Fig. 7H).

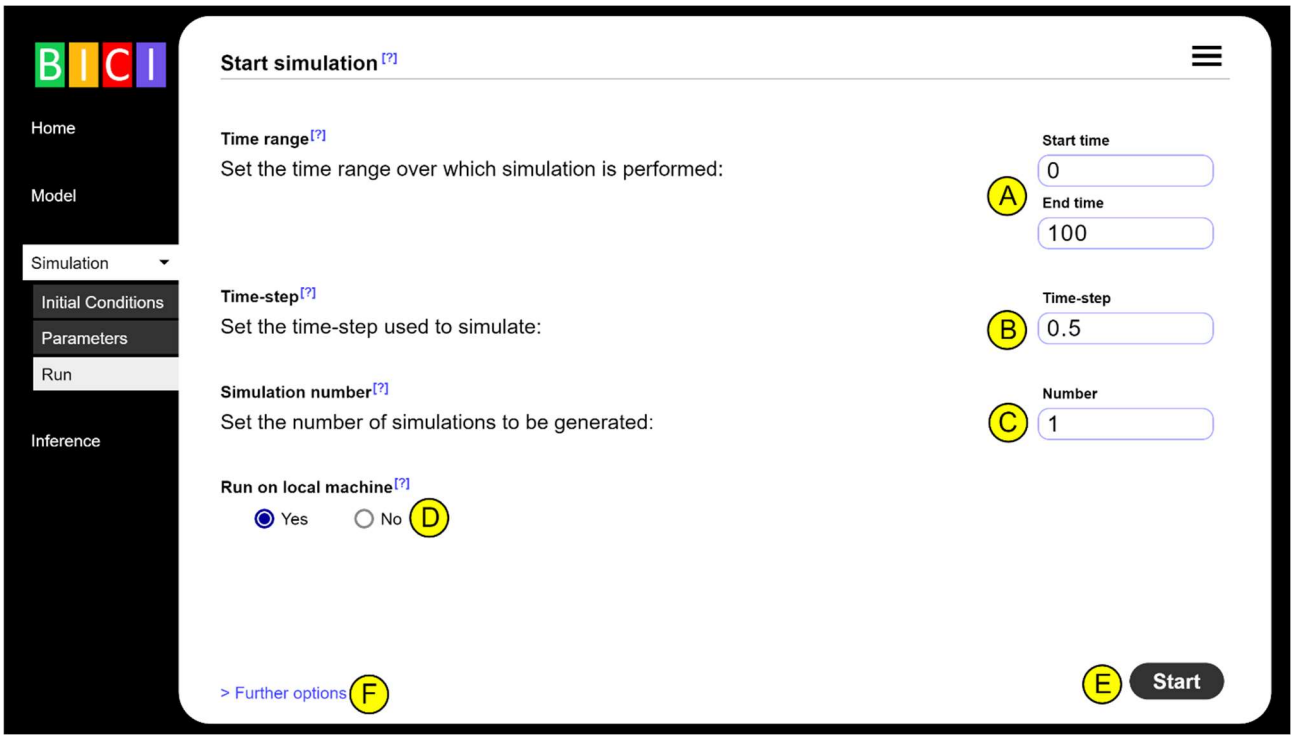


Figure 8 – *Running a simulation*. This page is used to set up a simulation. A: Time range, B: Time-step, C: Number of simulations to be performed, D: Where the simulation is done, E: Click to start, F: Further options available.

3.3 Run

Figure 8 shows the "Simulation→Run" page, which allows the user to set various options before starting a simulation:

- **Time range** (Fig. 8A) – Sets the time period over which the simulation is performed.
- **Time-step** (Fig. 8B) – A finite time-step is used to approximate the governing equations (see next section).
- **Number** (Fig. 8C) – This sets how many simulations are performed. Because simulations are stochastic, so each will provide a different potential realisation from the system. Visualising the overall envelope of many simulations can be a useful way to predict probability distributions for likely future behaviour.

- **Run on local machine** (Fig. 8D) – BICI can either be run on the local machine (for small jobs) or on a Linux cluster (for big jobs, see §6.4). Because simulation is generally much faster than inference, most jobs can be run locally.

Clicking on the "Start" button (Fig. 8E) gets a simulation under way. Once complete, the user is directed to the "Results" page (see §3.4).

3.3.1 How to set the time-step

The time-step Δt should be chosen to be sufficiently long to allow the algorithm to be fast, but sufficiently short for it to be accurate. Assessing this balance depends on the model type:

Population-based – Here populations are updated with the approximate τ-leaping algorithm using Δt. This approximation starts to break down when transition rates exceed around $1/(5\Delta t)$.

Individual-based – In this case the time-step determines how often population numbers are updated. Consequently, transitions which do not contain populations are entirely unaffected by Δt. Transitions which do, however, are affected when rates exceed around $1/(5\Delta t)$ (or equivalently when transition means are less than around $5\Delta t$).

A run-time warning is generated if the time-step is either chosen to be too large (leading to discretisation errors) or too small (becoming computationally inefficient). Note this is solely a user guide and can be ignored (see Appendix G).

3.3.2 Further options

Additional features can be specified by clicking on "Further options" (Fig. 8F):

- **Individual max** - When one or more species use an individual-based model, this sets the maximum number of individuals allowed (exceeding this limit results in an error message). Such a threshold is introduced to stop BICI running out of memory when the individual number diverges. By default, this is set to 20,000 individuals. The text box can be used to make this limit higher, if required.
- **Parameter output** – If the number of elements in a tensor exceeds this value, results are not output. This is introduced to stop very large output files (for example it might avoid the distance matrix being output every MCMC iteration). By default, this is set to a maximum of 1000 elements, but can be changed using the text box.
- **Set random seed** – Simulation and inference rely on pseudorandom number generators. These start with an initial value called a "seed". If the seed option is turned off (as it is by default), this seed is randomly generated by the BICI interface (in which case BICI will produced different results each time it is run). If the seed is set, however, BICI will consistently give the same results, despite being stochastic. The seed can take any value between 0 and 10,000.

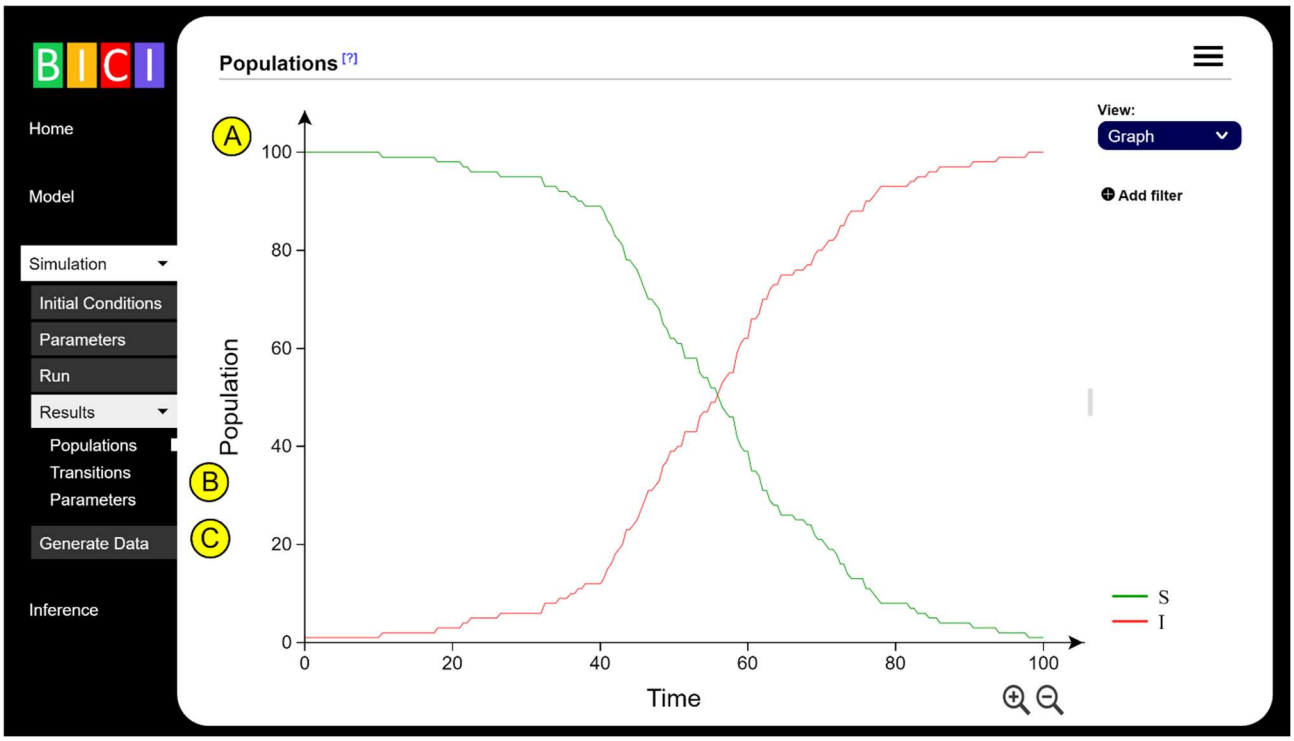


Figure 9 – *Results from a simulation*. A: This shows the time variation in the populations within the S and I compartments (taken from M1.1 in §8), B: Click on different menu options for visualisations of populations, transition or parameters, C: Click to generate simulated data.

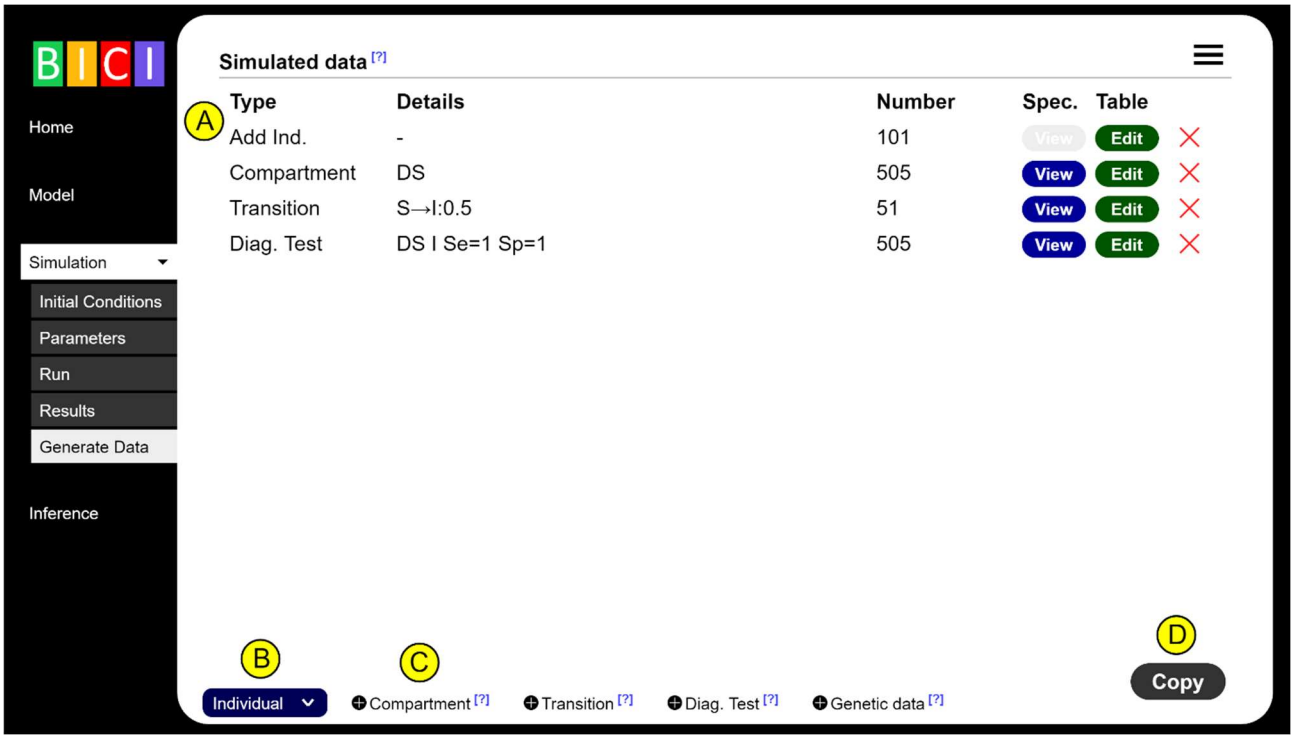


Figure 10 – *Generating simulated data*. A: A table of simulated data sources, B: Choose data type, C: Select data source to be simulated, D: Click to copy simulated data to the inference page.

3.4 Results

The "Simulation→Results" page shows outputs from a simulation. The illustrative example in Fig. 9A gives a plot of the populations in different model compartments as a function of time. Here there are initially 100 susceptible individuals (S compartment, denoted by the green line) and just a single infected (I compartment, denoted by the red line). Due to disease transmission, the susceptible individuals increasingly make the transition to becoming infected.

Different sections on the main menu allow for a variety of visualisations (Fig. 9B) focusing on "Populations", "Transitions" and "Parameters" (and also "Individuals" in the case of individual-based models). Further details for these are provided in §4.4 and Appendices H-K.

A description of various potential warning messages provided after code execution is given in Appendix G.

3.5 Generating data

The "Simulation→Generate Data" page on Fig. 10 can be used to create data based on simulated results. These hypothetical datasets can then be used as inputs for performing inference. This not only provides a good test to show that inference is able to successfully recover model parameters, but it is also an invaluable tool in experimental design, because it allows the user to estimate the likely power of a given setup[15].

The drop-down list (Fig. 10B) allows for different data types to be selected: "Init. Cond.", "Individual", "Population" and "Additional". These are examined in the following sections:

3.5.1 Simulated initial conditions data

Clicking these options essentially recreates any data that was entered on the "Simulation→Initial Conditions" page.

3.5.1.1 Init. Pop

Click "⊕ Init. Pop." to generate data that represents the initial population from the simulation. Selecting "Done" adds this data to the list in Fig. 10A (as it does with all the other options below).

3.5.1.2 Adding populations to the system

Click "⊕ Add Pop." to generate data for any populations added after the start time.

3.5.1.3 Removing populations from the system

Click "⊕ Remove Pop." to generate data for any populations removed after the start time.

3.5.1.4 Adding individuals

Click "⊕ Add Ind." to generate data for individuals added in the simulation.

3.5.1.5 Removing individuals

Click "⊕ Remove Ind." to generate data for individuals removed in the simulation.

3.5.1.6 Moving individuals

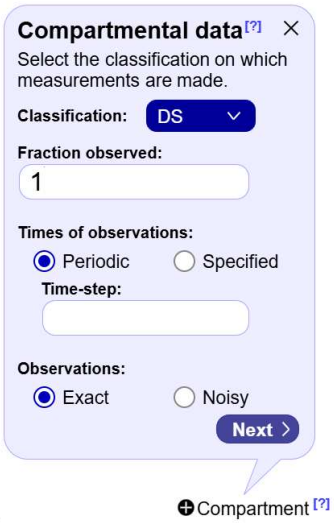
Click "⊕ Move Ind." to generate data for individuals moved in the simulation.

[15] In particular, it allows for an estimation of model parameter uncertainties, which are often key to understanding whether a given experiment would be sufficiently informative to answer the scientific question under study.

3.5.2 Simulated individual-level data

3.5.2.1 Compartment data

Click "⊕ Compartment" (Fig. 10C) to generate compartmental data.



The classification on which compartments are measured is selected from the drop-down list (see right). The fraction observed can be set (by default 1, but a lower value, *e.g.* 0.5, would result in only a fraction of observation being made). The timings of the compartmental observations can either be made at periodic intervals, or at specified time points.

The observations can be chosen to be exact (which will generate results like 'S' or 'I') or noisy. In the latter case, a specified value represents the probability p of observing the correct compartment. The remaining probability 1-p is equally distributed across all other compartments. For example, in an SIR model if p=0.9 and the true compartment is I, the simulated output would be 'S:0.05|I:0.9|R:0.05' (see §4.1.2.1 for formatting compartmental uncertainties.

3.5.2.2 Transition data

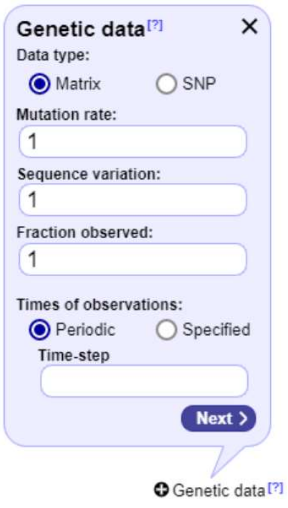
Click "⊕ Transition" to generate transition data. Please refer to §4.1.2.2 for further details.

3.5.2.3 Diagnostic test data

Click "⊕ Diag. Test" to generate disease diagnostic test data. The fraction observed can be set (by default 1, but a lower value, *e.g.* 0.5, would result in only a fraction of observation being made). The option to either measure periodically or at specified time points is given. Please refer to §4.1.2.3 for further details.

3.5.2.4 Genetic data

Click "⊕ Genetic data" to generate genetic data.



The nature of this data depends on its type:

- **Matrix** – For a given set of pathogen genetic measurements, this generates a matrix of SNP differences between those measurements.
- **SNP** – For a given set of pathogen genetic measurements, this option outputs simulated SNP data. If selected, it is necessary to specify a text "root" used for SNP names, along with the number of SNP base pairs to be simulated.

Other options are as follows:

- **Mutation rate** – This sets the rate at which the pathogen genome mutates (in base-pair mutations across the whole genome per unit time).
- **Sequence variation** – This allows for variation in sequences for infections entering the system. The number of changes compared to a consensus sequence is drawn from a Poisson distribution with this specified mean.
- **Fraction observed** – Gives the proportion of infected individuals observed at a given time point (in real dataset this fraction is often quite low).

- **Times of observations** – The times at which individuals are observed. This can either be set as periodic or at specified time points.

3.5.3 Simulated population-level data

3.5.3.1 Population data

Click "⊕ Population" to generate population data. Measurements can either be made periodically, or at specified time points. Please refer to §4.1.3.1 for further details.

3.5.3.2 Population-level transition data

Click "⊕ Pop. Transition" to generate population-level transition data. Measurements can either be made between periodically defined intervals (*e.g.* weekly), or at specified time points (note, here the observed numbers of transitions is measured between these time points). Please refer to §4.1.3.2 for further details.

3.5.4 Simulated additional data

3.5.4.1 Individual effect data

Click "⊕ Ind. Effect" to generate individual effect data. Select the individual effect required from the drop-down list, and the data source generated will contain the simulated values for the individual effect. When performing inference this is useful because it allows for the prediction accuracy to be estimated (a metric that determines how well inference is able to recover true individual effect values).

3.5.4.2 Individual group data

Click "⊕ Ind. Group" to specify groups of individuals in the simulated population. This can be useful when estimating prediction accuracies for specified groups. Please refer to §4.1.4.2 for further details.

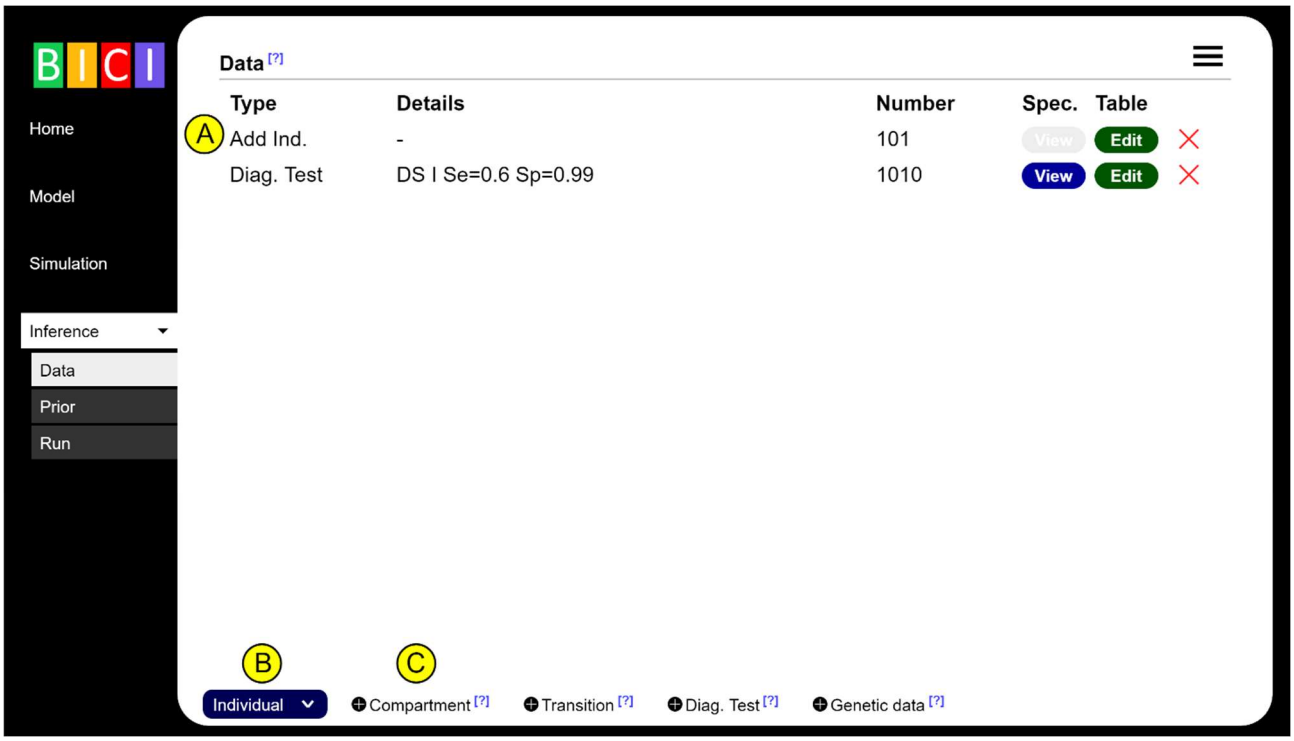


Figure 11 – *Loading data*. A: Table of data sources, B: Choose data type, C: Select data source to be added.

4) Inference

Bayesian inference is the process of combining data (which can take a variety of forms and may be inherently noisy) with previous knowledge regarding model parameters (the prior) to generate the best available estimates for model parameters and system dynamics (the posterior).

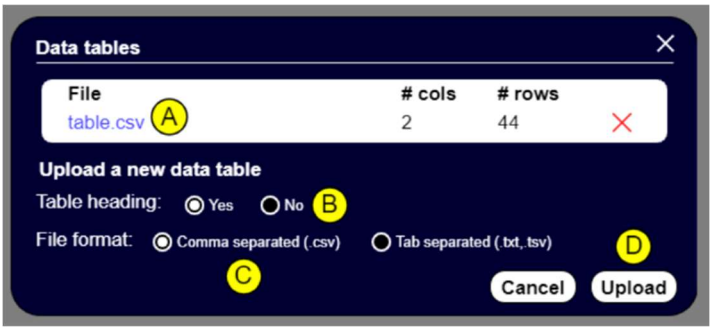
4.1 Data

Rather than loading the data all at once, the user loads different sources of data one at a time (in any order). As illustrated in Fig. 11A, the "Inference→Data" page shows the data sources currently loaded[16].

The data comes in different types, as selected using the drop-down list in the bottom left-hand corner (Fig. 11B):

- **Init. Cond.** (§4.1.1) – Provides information about the initial conditions, as well as any individuals that are added, removed or enforced to move compartment.
- **Individual** (§4.1.2) – Individual-level data, such as the times at which individuals undergo a given transition, when they are in a given compartment or disease diagnostic test results. Genetic sequence data for the pathogen can also be added, if available.
- **Population** (§4.1.3) – Population-level data, such as time-series compartmental population measurements, or time-series population-level transition numbers.
- **Additional** (§4.1.4) – This provides addition information, not used for inference. Here true values for individual effects can be loaded (such that prediction accuracies can be calculated) or individual groups can be specified (*e.g.* sires and progeny in a quantitative genetics model).

Data is loaded into a BICI using a dialogue box (see right). A: If a data table has previously be loaded (*e.g.* for another data source) it can be directly selected from the list, B: This determines if the table has a heading row at the top (recommended), C: Two formats for data tables are supported, either columns are comma separated ('.csv' files) or tab separated ('.txt'/'.tsv' files), D: Clicking "Upload" allows the user to upload a new file.



4.1.1 Initial conditions data

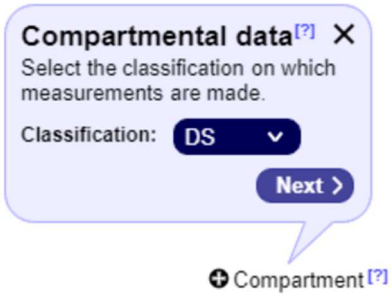
This uses exactly the same set of possibilities at for simulation, so the reader is referred to §3.1.

[16] This example is taken from example C4 in §8 and shows two sources related to adding individuals at the start time and disease diagnostic test results.

4.1.2 Individual-level data

4.1.2.1 Compartment data

Click "⊕ Compartment" (Fig. 11C) to add compartmental data. This provides information about the compartment individuals are in at specified time points, *e.g.* it could be used to denote the fact that individual "Ind-1" is observed to be in the infected I compartment at time t=3.45 and "Ind.2" is in the recovered R compartment at time t=9.21.



REQUIRES – A data table containing at least the following columns: "ID" a unique identifier for individuals, "t" the measurement times and the classification on which the measurements are made, *e.g.* "DS".

The last column contains compartmental specifications that can be made in a number of different ways to reflect any potential uncertainty:

- **Precisely** – The known compartment is exactly specified, *e.g.* 'S' would mean that the individual is in the S compartment at the measurement time.
- **Multiple choices** – Different possibilities are given, separated by the '|' character, *e.g.* 'S|I' would mean an equal probability of being in S or I.
- **Probabilities** – The probabilities of being in any potential compartments are explicitly specified, *e.g.* 'S:0.7|I:0.3' would mean a 70% probability of being in S and a 30% probability of being in I. Such an expression could also contain parameters, *e.g.* 'S:p|I:1-p', that could be estimated during inference.
- **Unspecified** – This is denoted by '.' and indicates missing data.

If an individual is not in the system when a compartmental observation is made, this can be indicated by the '!' character.

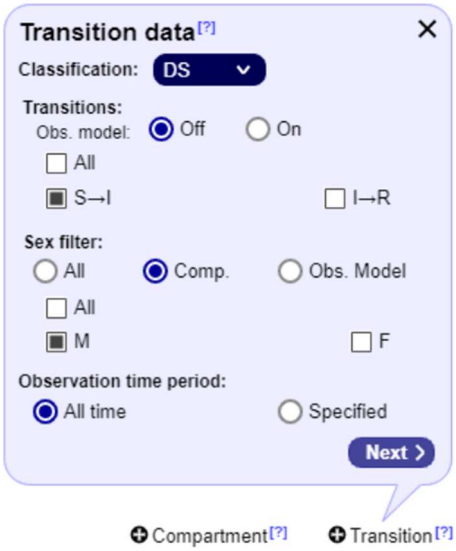
EXAMPLE – Here compartmental observations on disease status classification 'DS' are made on three individuals at different times: 'Ind-1' is in compartment 'S' at time 20 and 'I' at time 30, 'Ind-3' is in 'I' or 'R' at time 10 but with a 60%/40% probability at time 100, and for 'Ind-4' the observation probability is parameterised by b:

	A	B	C	D	E	F	G	H
1	ID	t	DS					
2	Ind-1	20	S					
3	Ind-1	30	I					
4	Ind-3	10	I|R					
5	Ind-3	100	I:0.6|R:0.4					
6	Ind-4	100	I:b | R:1-b					

4.1.2.2 Transition data

Click "⊕ Transition" to add transition data. This provides information about the timings of individual transitions, *e.g.* it could be used to store the precise infection times for a group of individuals.

REQUIRES – A data table containing at least the following columns: "ID" a unique identifier for individuals[17] and "t" the times individuals undergo the transition.



When loading transition data, the following options are available:

- **Classification** – Specifies the classification within which the transition observations are made.
- **Transitions** – Checkboxes can be used to select the transition (or transitions) under observation. Alternatively, an observation model can be specified that gives the probability of observing a given transition, *e.g.* '$P(S\to I)=0.3$' would mean a 30% probability of observing an infection transition. Note, parameters can be placed into the observation model (*e.g.* '$P(S\to I)=p$' or '$P(S\to I)=\eta(t)$' to incorporate time variation) whose parameters can be estimated during inference.
- **Compartmental filter** – When there is more than one classification, the compartments in the other classifications can be used to filter which individuals are actually observed. Three possibilities exist:
  - *All* – All compartments are observed.
  - *Comp.* – Uses checkboxes to indicate observed compartments.
  - *Obs. Model* – Allows for specification of an observation model (again, this can involve parameters with potential time variation).
- **Observation time period** – Two options exist:
  - *All time* – Goes over the entire inference period.
  - *Specified* – Allows for a time window to be set.

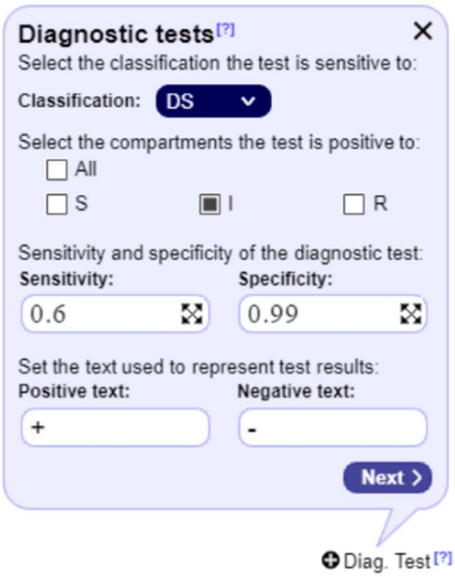
EXAMPLE – This shows data for infection times (note 'Ind-4' did not become infected, so it's not included in this data table):

	A	B	C	D	E	F	G	H
1	ID	t						
2	Ind-1	13.32						
3	Ind-2	14.32						
4	Ind-3	1.23						
5	Ind-5	54.34						

[17] Note, if no ID is associated with the observations, one can be arbitrarily be made up, *e.g.* if only recovery times are known from an epidemic then the first recovered individual can be called "Ind-1", the second "Ind-2", and so on and so forth.

4.1.2.3 Diagnostic test data

Click "⊕ Diag. Test" to add disease diagnostic test data. This provides a (noisy) way to assess if individuals are infected or not, *e.g.* such data could be used to denote the fact that individual "Ind-1" was tested at time t=34.5 and gave a positive result, whereas "Ind-2" was tested at time t=14.0 and was found to be negative.



REQUIRES – A data table containing at least the following columns: "ID" a unique identifier for individuals, "t" the times the individuals are tested, and "Result", which provides the binary results of the tests (see below for what values this can take).

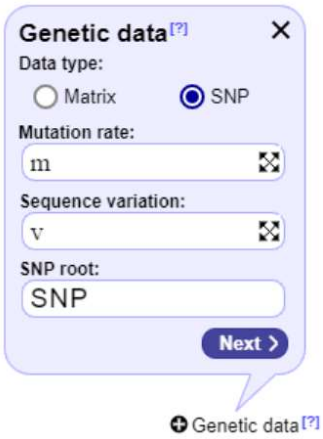
When loading diagnostic test data, the following options are available:

- **Classification** – Specifies the classification within which the observations are made.
- **Infected compartment(s)** – Checkboxes are used to denote which compartments the test aims to give a positive result for.
- **Diagnostics** – Values for the test sensitivity and specificity are set (note, these can be parameters estimated under inference).
- **Text value** – The text used to represent positive and negative test results in the data file (*e.g.* '0'/'1' or '+ve'/'-ve').

EXAMPLE – This shows diagnostic test data for five individuals at various time points:

	A	B	C	D	E	F	G	H
1	ID	t	Result					
2	Ind-1	10	-					
3	Ind-2	10	+					
4	Ind-3	15	+					
5	Ind-4	20	-					
6	Ind-5	20	-					

4.1.2.4 Genetic data



Click "Genetic data" to incorporates pathogen genetic sequence data. Such data can help inform how a pathogen moves between individuals, *e.g.* if a genetic sequence for the pathogen is observed in individual "Ind-1" at t=10.5 and a closely related (or identical) genetic sequence is observed in "Ind.2" at time t=15.6, it is evidence that the pathogen may have been transmitted from "Ind-1" to "Ind-2". Pathogen genetic data can be particularly good at estimating rates of transmission between groups (*e.g.* different species) or geographical spread.

Two types of data can loaded:

- **Matrix** – A matrix of SNP differences between a set of pathogen genetic measurements.
- **SNP** – Raw SNP data for a set of pathogen genetic measurements.

The following options must be set:

- **Mutation rate** – A parameter giving the rate at which the pathogen genome mutates (in base-pair changes across the whole genome per unit time).
- **Sequence variation** – A parameter that allows for variation in sequences for infections entering the system. The number of changes compared to a consensus sequence is drawn from a Poisson distribution with this specified mean.
- **SNP root** – Gives the root name for SNP columns in the file (for SNP data).

REQUIRES – A data table containing at least the following columns: "ID" gives a unique name for each individual, "t" giving the time of the observation. The following columns depend on the type of observation:

- *Matrix* – A column "Obs" provides a unique name for each genetic observation. Elements down this column must also be column headings in the table that represent a square matrix giving base-pair differences between the observations.
  EXAMPLE – Here pathogen genetic observations are made on five individuals at different time points (note the matrix of base-pair differences must be symmetric):

	A	B	C	D	E	F	G	H
1	ID	t	Obs	Obs-1	Obs-2	Obs-3	Obs-4	Obs-5
2	Ind-1	43	Obs-1	0	12	34	23	56
3	Ind-2	45	Obs-2	12	0	34	23	99
4	Ind-3	23	Obs-3	34	34	0	34	67
5	Ind-4	10	Obs-4	23	23	34	0	34
6	Ind-5	34	Obs-5	56	99	67	34	0

- *SNP* – The data table has a series of columns giving sequence data which have 'SNP root' at the beginning of their column name. Elements of these columns are expected to contain two characters denoting the version of the SNP, *e.g.* 'AA', 'AB', 'BA', or 'BB'.

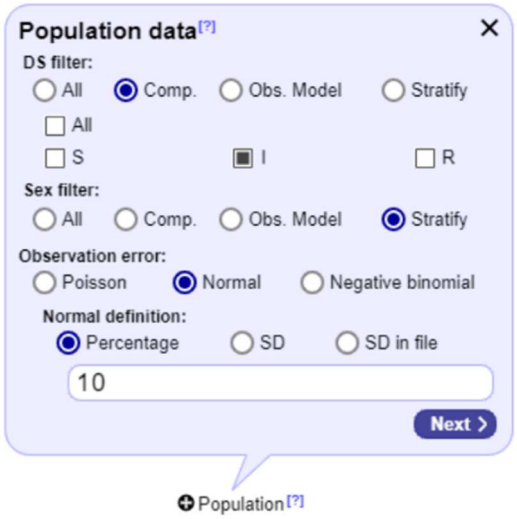
EXAMPLE – Here pathogen genetic SNP data[18] are defined for five individuals at different time points (in this case 'SNP root' is 'SNP').:

	A	B	C	D	E	F	G	H
1	ID	t	SNP1	SNP2	SNP3	SNP4	SNP5	
2	Ind-1	43	AA	AB	BB	AA	AA	
3	Ind-2	45	AB	AB	AB	AA	BB	
4	Ind-3	23	BB	BB	BB	AA	BA	
5	Ind-4	10	AA	AB	AA	AB	AB	
6	Ind-5	34	AA	AB	BB	AA	AA	

4.1.3 Population-level data

4.1.3.1 Population data

Click "⊕ Population" to add population data. Population data informs how many individuals are in specified compartments at different times, *e.g.* it could be measurements for the number of infected individuals at a number of different time points (so-called time series data).



REQUIRES – A data table containing at least the following columns: "t" the time at which population measurements are made and "Population", which gives the estimated number of individuals.

The population under study can be filtered for each of the classifications using the following options:

- **All** – All individuals within the classification are counted.
- **Comp.** – Selected checkboxes are used to choose certain compartments (*e.g.* only infected individuals in compartment 'I').
- **Obs. Model** – An observation model can be specified that gives the fraction of individuals observed in one or more compartments, *e.g.* 'P(S)=0.3' would mean that only a 30% fraction of individuals are observed in the S compartment (note, fractions can be larger than one, corresponding to cases in which the observed data is proportional to the population, *e.g.* a measurement of pathogen level would be expected to be proportional to the number of infected individuals). Parameters can be placed into the observation model (*e.g.* 'P(S)=p' or 'P(S)=η(t)' to incorporate time variation) that can be estimated during inference.
- **Stratify** – The data file contains an additional column (with heading given by the classification name) that specifies the compartment corresponding to that particular observation. Multiple possible compartments are separated by '|' (*e.g.* 'E|I' would include exposed and infected individuals). Furthermore, an observation model can be incorporated (*e.g.* 'S:0.3|I:0.3' or 'S:p|I:q').

[18] Note here only 5 SNPs are shown but in reality this number would typically be many thousand. Interestingly the speed of the algorithm doesn't depend strongly on the number of SNPs.

The observation error captures noise in the measurement process itself. This assumes that the observed value is drawn from a distribution with mean given by the underlying value.

Three possible distributions can be selected from:

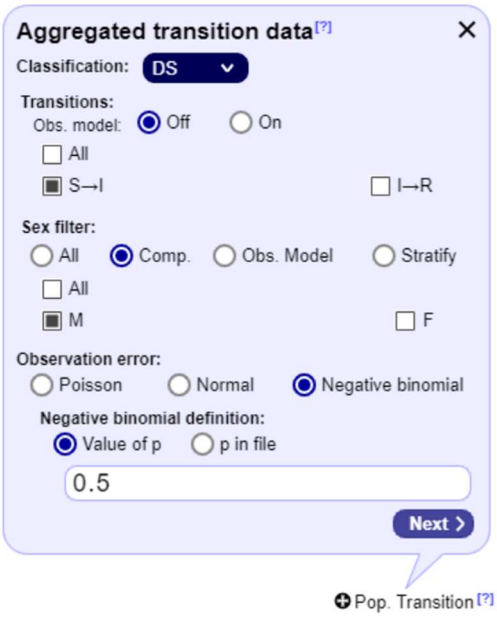
- **Poisson** – This has a variance equal to the mean.
- **Normal** – This can be chosen to have a variance less than the Poisson distribution. The standard deviation can be defined in one of three different ways:
    - *Percentage* – The standard deviation is set to a percentage of the mean.
    - *SD* – The standard deviation is explicitly fixed.
    - *SD in file* – The data table contains an additional column "SD" that specifies the standard deviation.
- **Negative binomial** – This distribution is overdispersed with respect to the Poisson distribution. Specifically, the variance is given by the mean divided by a probability p. This probability can be defined to one of two ways:
    - *Value of* p – The probability p is explicitly fixed.
    - p *in file* – The data table contains an additional column "p" which specifies its value.

EXAMPLE – This shows the estimated number of infected individuals at different time points. A normally distributed observation model has been selected with the standard deviation specified in the table (reflecting the uncertainty in the observations):

	A	B	C	D	E	F	G	H
1	t	Population	SD					
2	10	67	15					
3	20	102	20					
4	30	203	40					
5	40	502	102					
6	50	1023	302					

4.1.3.2 Population-level transition data

Click "⊕ Pop. Transition" to add aggregated population-level transition data. Here the observation period is sliced into divisions (usually of equal size, although this is not a requirement) and the overall number of transitions of a specified type is counted within these divisions, *e.g.* such data could represent the number of population-wide cases per week during an epidemic. The size of divisions would usually be chosen to be sufficiently large such that a reasonable number of events would be expected to occur[19], but sufficiently small to retain temporal resolution.



Note, if data on individual timings of transitions exist, it is usually best to incorporate these as individual transition data, as described in §4.1.2.2[20].

REQUIRES – A data table containing at least the following columns: "Start" and "End" that give the start and end time of the divisions over which transitions are counted (note, the end time of one division would normally be the same as the start time for the next) and "Number" which gives the estimated number of transitions.

Either one or multiple transitions can be selected within a given classification. Alternatively, an observation model can be specified that denotes the fraction of transitions observed (*e.g.* '$P(S\rightarrow I)=0.5$' would correspond to only 50% of infection transitions being observed).

If, rather than time series data, only the overall number of transitions is known, this can be incorporated by setting the start and end times to encompass the entire time period, *e.g.* this could be used when only the total number of recoveries is known at the end of an epidemic (as estimated from a serological test).

See §4.1.2 for further details on classification filters and the observation error.

EXAMPLE – This shows the number infections during weekly intervals (here a Poisson distributed observation error is used):

	A	B	C	D	E	F	G	H
1	Start	End	Number					
2	0	7	20					
3	7	14	34					
4	14	21	64					
5	21	28	234					
6	28	35	345					

[19] Say more than five during the peak of an epidemic outbreak, such that stochastic noise from the Poisson process does not dominate.

[20] Note, if no ID is associated with the observations, one can be arbitrarily be made up, *e.g.* if only recovery times are known from an epidemic then the first recovered individual can be called "Ind-1", the second "Ind-2", and so on and so forth.

4.1.4 Additional data

4.1.4.1 Individual effect data

Click "⊕ Ind. Effect" to load up true values for individual effects (to compare against estimated values from inference). Note, these true values are usually generated from simulated data (see §3.5.4.1). This data type has no influence on inference.

REQUIRES – A data table containing the column "ID" a unique identifier for individuals and "Value", which gives the true individual effect values.

EXAMPLE – This example shows individual effect values loaded for five individuals:

	A	B	C	D	E	F	G	H
1	ID	Value						
2	Ind-1	1.243						
3	Ind-2	0.834						
4	Ind-3	2.345						
5	Ind-4	0.034						
6	Ind-5	0.934						

4.1.4.2 Individual group data

Click "⊕ Ind. Group" to load up groupings of individuals. Note, this is used for visualisation (*e.g.* to view individual effects for parents) and has no bearing on inference. The group must be given a name (*e.g.* "sires") and individuals within the group can either be specified in the interface (using checkboxes) or from a file:

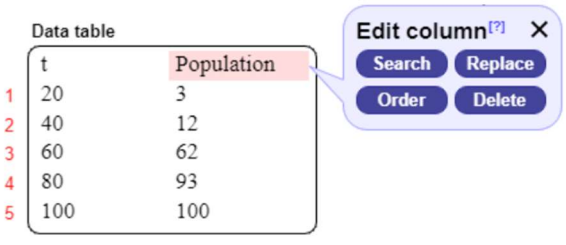
REQUIRES – A data table containing the column "ID" a unique identifier for individuals which list this in a group.

EXAMPLE – This loads up a group of five individuals:

	A	B	C	D	E	F	G	H
1	ID							
2	Ind-1							
3	Ind-2							
4	Ind-3							
5	Ind-4							
6	Ind-5							

4.1.5 Table editing

When loading or editing data tables, BICI provides some simple tools that can be accessed by either clicking on the table headings or row numbers:



- **Search** – Find rows containing a specified string. If multiple results are returned, these can be iterated through. A wildcard '*' can be used to

represent any ignored text (*e.g.* "S*" would find all entries beginning with 'S', or "*ind*" would return all entries containing 'ind').

- **Replace** – Replace one text string with another along a column. Wildcards '*' can be used in this replacement. For example, searching for "S*" and replacing it with "R*" would replace all entries beginning with 'S' to corresponding entries beginning with 'R'. Note, the same number of wildcards must be used in both the search and replace text boxes.
- **Delete** – Delete rows for which a given column value corresponds to a specified string. A wildcard '*' can be used to represent text that is ignored (*e.g.* "S*" would delete all entries beginning with 'S', or "*ind*" would delete all entries containing 'ind')
- **Order** - Order rows based on elements along a column. This ordering can either be performed numerically or alphabetically.

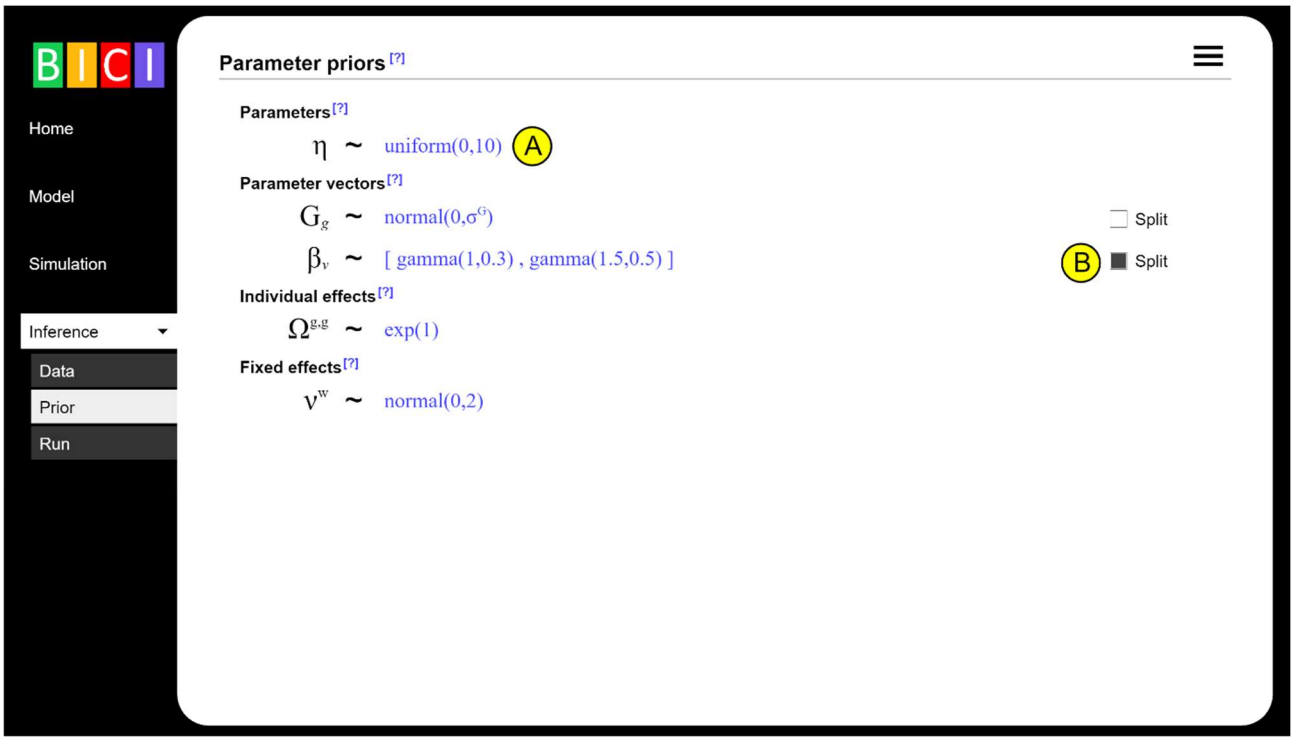
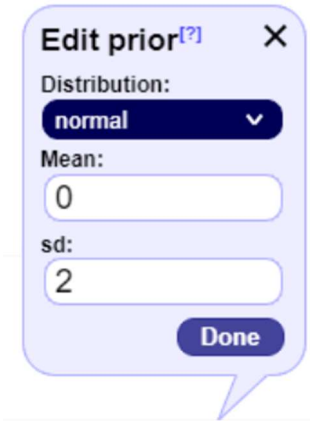


Figure 12 – *The prior*. A: A list of all parameters requiring priors, B: Selecting this checkbox allows for the prior to be specified for different elements of the parameter.

4.2 Prior

The "Inference→Prior" page on Fig. 12 allows for prior distributions to be assigned to model parameters (note, this list doesn't include constants, distributions or derived parameters).

When Bayesian inference is performed, priors are used to encapsulate any previous knowledge about model parameters. One common approach is to identify a range of plausible values a given parameter is likely to take (based on expert judgment), and use this to inform a uniform distribution.



A commonly used alternative approach for providing an uninformative prior is the Jeffreys prior (see Appendix F). Unlike the uniform prior, this has the advantage of remaining the same under reparameterisation. This option is recommended in most cases.

Clicking on the prior definitions (Fig. 12A) allows for them to be edited. For most parameters, prior distributions can be selected from the following nine different possibilities:

- **inverse** – An inverse distribution truncated to lie between a specified minimum and maximum. Note, this only applies to positive quantities. It is a commonly used distribution because it represents the Jeffreys prior for means and rates.
- **uniform** – A flat prior distribution between a minimum and maximum value.
- **power** – A polynomial distribution truncated to lie between a specified minimum and maximum and with a specified power. This can be used for non-inverse Jeffreys priors (*e.g.* the coefficient of variation has a Jeffreys prior cv^{-2} for the gamma distribution).
- **exp** – An exponential distribution with a specified mean (only appropriate if the parameter is strictly positive).
- **normal** – A normal distribution with a specified mean and standard deviation.
- **gamma** – A gamma distribution with a specified mean and coefficient of variation (for positive parameters only).
- **log-normal** – A log-normal distribution with a specified mean and coefficient of variation (for positive parameters only).
- **beta** – A beta distribution with specified α and β values (for parameters that go between 0 and 1 only). Note, the distribution mean is given by $\alpha/(\alpha+\beta)$.
- **bernoulli** – A Bernoulli distribution with a specified mean (for binary parameters that can take the values 0 or 1).
- **fix** – Fixes a parameter to a particular value.

For parameter factors, a special type of prior is used:

- **mdir** – The modified Dirichlet distribution with a specified sigma (which determines the fractional variation in parameter factor elements). See Appendix F for details.

For covariance matrices associated with individual effects, two special types of prior can be selected from:

- **mvn-jeffreys** – This is a Jeffreys prior for a multivariate normal distribution. Here the minimum specifies a limit on the determinant of the covariance matrix and the maximum sets a hard limit on the variance (see Appendix F for details).
- **mvn-uniform** – This specifies a uniform minimum and maximum variance (see Appendix F for details).

4.2.1 Split

If the "Split" checkbox is selected (Fig. 12B), it means that a prior specification is made separately for each element of the parameter. If many such specifications are required, these can be loaded from a data file:

REQUIRES – A data table containing columns with headings given by the indices of the tensor and a final "Prior" column. See below for the text format used for this last column (§2.2.10 provides details on file formatting).

4.2.2 Text representation for a prior or distribution

When written in text format, priors (or distributions) are expressed in the following way, dependent on distribution:

- **"inverse(min,max)"** – An inverse distribution between a specified minimum and maximum, *e.g.* "inverse(0.01,1)".
- **"uniform(min,max)"** – A uniform distribution with specified minimum and maximum, *e.g.* "uniform(4,5)".
- **"exp(mean)"** – An exponential distribution with specified mean (only appropriate if the parameter is strictly positive).
- **"normal(mean,sd)"** – A normal distribution with a specified mean and standard deviation.
- **"gamma(mean,cv)"** – A gamma distribution with a specified mean and coefficient of variation (for positive parameters only).
- **"log-normal(mean,cv)"** – A log-normal distribution with a specified mean and coefficient of variation (for positive parameters only).
- **"beta(α,β)"** – A beta distribution with specified alpha and beta values.
- **"bernoulli(mean)"** – A Bernoulli distribution with a specified mean (for binary parameters that can take the values 0 or 1).
- **"fix(value)"** – Fixes a parameter to a particular value.
- **"mdir(σ)"** – The modified Dirichlet distribution, where σ controls the fractional variation in parameter factor elements.
- **"mvn-jeffreys(min,max)"** – The uninformative Jeffreys prior used for multivariate normal covariance matrices applied to individual effects. Here the minimum specifies a limit on the determinant of the covariance matrix and the maximum sets a hard limit on the variance (see Appendix F for details).
- **"mvn-uniform(min,max)"** – An uninformative uniform distribution used for multivariate normal covariance matrices applied to individual effects. The minimum and maximum limits determine a range on the typical diagonal elements of the covariance matrix.

4.2.3 Guide to choosing a prior

In situations in which nothing is apparently known about parameters, it is often tempting for users to provide very uninformative priors (*e.g.* uniform across a wide range). This can result in very long execution times. Methods like MCMC rely on generating an initial state by sampling from the prior and simulating from the system. If this state is very different from the posterior it can take a long time for the algorithm to get from one to the other (a time period known as burn-in). Worst still, it may be that entirely unphysical solutions exist within parameter space into which the system can become stuck. Consequently, a careful examination of the prior is recommended in cases in which the burn-in phase is taking a long time.

Example – Suppose we consider an SIR model with an infection rate[21] of $(\beta I/N) + \phi$ (where β is the transmission rate, I is the number of infected individuals, N is the total number of individuals in the system and ϕ is an external rate of infection) and recovery rate γ. Since we don't know anything about the disease or the external infections entering the system, it is tempting to put large priors on the all the parameters (say $\beta \sim \text{uniform}(0,100)$, $\gamma \sim \text{uniform}(0,100)$, $\phi \sim \text{uniform}(0,100)$). However just based on some simple reasoning this can be considerably restricted:

- Biologically in might be implausible to expect that an individual would recover in less than one day or be infected for more than 100 days on average. This would yield the prior $\gamma \sim \text{uniform}(0.01,1)$.
- How about β? The basic reproduction number R_0 is the expected number of cases directly generated by one case in a population where all individuals are susceptible to infection. It is mathematically expressed as $R_0 = \beta/\gamma$. The values for R_0 vary from disease to disease, but values above 20 are unrealistic (for human diseases at least). Since we know that γ cannot be larger than 1, this places an upper bound on β of $R_0^{max}\gamma^{max} = 20$. As the disease is known to propagate, so R_0 must be larger than one. This gives a lower bound of $R_0^{min}\gamma^{min} = 0.01$. Therefore, this implies the prior $\beta \sim \text{uniform}(0.01,20)$.
- How about ϕ? We perhaps know that it must, in some sense, be "small", as most infections are expected to occur between individuals rather than from this external source of infections. Suppose our period of study is T=100 days. Realistically ϕ must be below $1/T$, otherwise the vast majority of individuals in the system would become externally infected during this period. This places an upper bound on the prior, hence the selection $\phi \sim \text{uniform}(0,0.01)$ would be sensible.

It's important to note that our final prior specification ($\beta \sim \text{uniform}(0.01,20)$, $\gamma \sim \text{uniform}(0.01,1)$, $\phi \sim \text{uniform}(0,0.01)$) is still extremely broad. We have made no real assumptions, apart for arguing what is realistic. But the crucial point is that it is more that 5 million times smaller than our initial guess above! This can really help MCMC to be initialised with something sensible from which posterior predictions can be made.

4.2.4 Requirements on choosing priors

Distributions appear in both priors and also the system dynamics (*e.g.* determining transition times). The parameters for these distributions (means, standard deviations etc…) must be constrained to appropriate bounds for reasons of numerical accuracy, or such that they do not become ill-defined (*e.g.* a negative standard deviation is invalid).

BICI requires the following constraints (otherwise an error message is generated):

[21] A "rate" here is a probability of the event per unit time.

- **inverse(min,max)** – 'min' and 'max' must both be positive and 'min' must be smaller than 'max'.
- **uniform(min,max)** – 'min' must be smaller than 'max'.
- **power(min,max)** – 'min' and 'max' must both be positive and 'min' must be smaller than 'max'.
- **exp(mean)** – 'mean' must be larger than 10^{-9}.
- **exp(rate)** – 'rate' must be larger than 10^{-9}.
- **normal(mean,sd)** – 'mean' must be in the range $-10^{9} – 10^{9}$, 'sd' must be in the range $10^{-9} – 10^{9}$.
- **gamma(mean,cv)** – 'mean' must be in the range $10^{-14} – 10^{9}$, 'cv' must be in the range 0.01 – 10.
- **log-normal(mean,cv)** – 'mean' must be in the range $10^{-14} – 10^{9}$, 'cv' must be in the range 0.01 – 10.
- **beta(α,β)** – 'α' and 'β' must be in the range 0.01 – 100.
- **bernoulli(mean)** – 'mean' must be in the range 0 – 1 (inclusive).
- **weibull(scale,shape)** – 'scale' must be in the range $10^{-9} – 10^{9}$, 'shape' must be in the range 0.1 – 100.
- **Poisson(mean)** – 'mean' must be in the range $0 – 10^{9}$.
- **dirichlet(α)** – 'α' must be in the range 0.01 – 100.
- **period(time)** – 'time' must be greater than 10^{-9}.
- **mdir(σ)** – 'σ' must be in the positive.
- **mvn-jeffreys(min,max)"** – 'min' and 'max' must both be positive and 'min' must be smaller than 'max'. Convergence issues can become a problem if min is chosen too small (see Appendix F for details).
- **mvn-uniform(min,max)"** – 'min' and 'max' must both be positive and 'min' must be smaller than 'max'.

Note, while most of these ranges are very broad, some are restricted for reasons of numerical accuracy (in particular coefficients of variation and shape parameters). For these, it makes sense to use uniform distributions with bounds chosen to be within those permitted above.

For example, if the model contains transitions which are taken from the gamma(mean,cv) distribution, the coefficient of variation parameter 'cv' could have priors "uniform(0.01,2)" or "uniform(0.1,5)", but "uniform(0,10)" would likely cause an error as it would allow for 'cv' to stray outside the valid region.

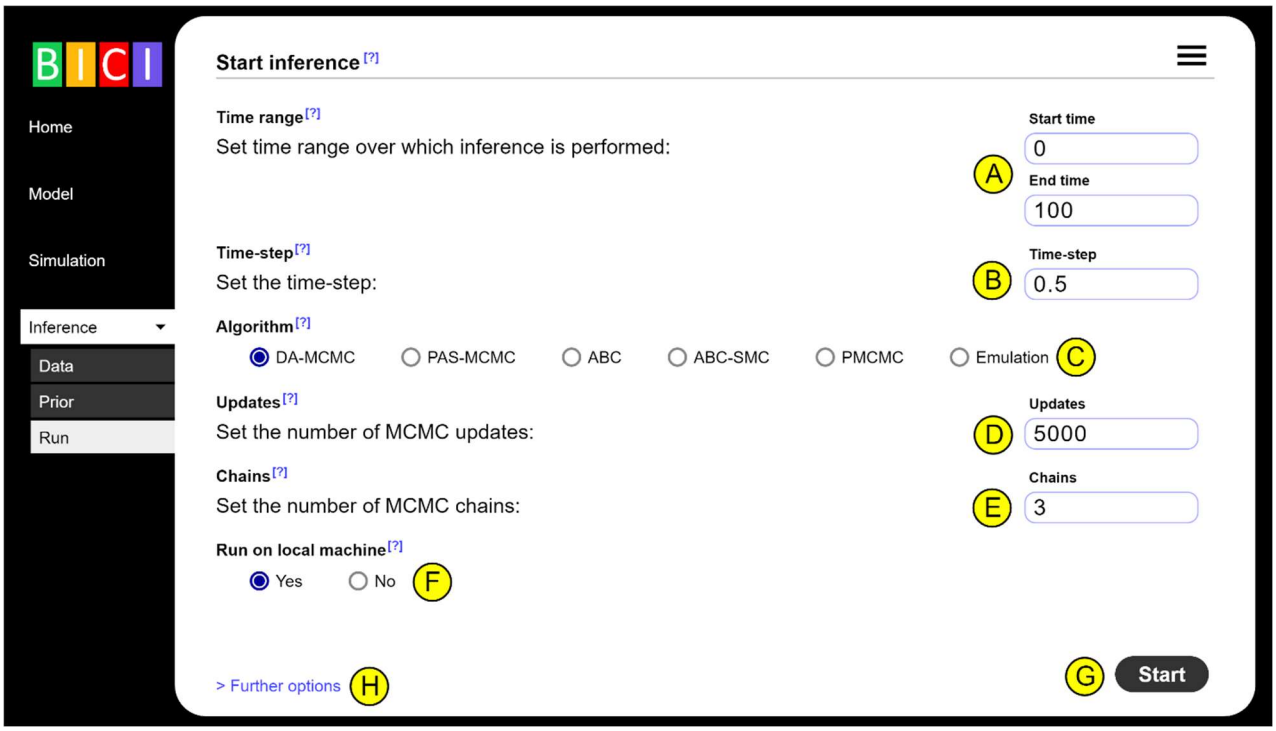


Figure 13 – *Running inference*. This page is used to set up inference. A: Time range, B: Time-step, C: Algorithm, D: Number of MCMC "updates", E: Number of MCMC chains, F: Where to run, G: Click to start, H: Further options available.

4.3 Run

The "Inference→Run" page allows the user to set various options before starting inference:

- **Time range** (Fig. 13A) – Sets the period over which inference is performed.
- **Time-step** (Fig. 13B) – For approximate methods it is often necessary to specify a time-step. A discussion on how this time-step should be chosen is given in §3.3.1.
- **Algorithm** (Fig. 13C) – Different algorithms can be used to perform inference (see next section).
- **Run on local machine** (Fig. 13F) – BICI can either be run on the local machine (for small jobs) or on a Linux cluster (for big jobs, see §6.4).

Inference is run by clicking on the "Start" button at Fig.13G. Once complete, the user is directed to the "Results" page (see §4.4).

4.3.1 Inference algorithms

Many different inference algorithms exist within the literature, of which a few have been implemented into BICI. Some are exact, some are approximate, and some require communications between interacting processes (which requires them to be run on a Linux cluster).

4.3.1.1 DA-MCMC

Data augmentation MCMC is the standard approach to Bayesian inference (although it can be slow for many individuals). This option requires further specifications:

- **Updates** (Fig. 13D) – An "update" constitutes a whole series of "proposals" that make changes to different aspects of the model (*i.e.* different parameters and/or parts of the state) in a range of ways. The number of updates informs how long MCMC takes to run. Running with more updates may produce a better approximation to the posterior, but it can also become computationally wasteful. *A priori* it is difficult to know what this number should be, but the default value of 5000 is usually a good first guess. MCMC diagnostics can be used to determine if inference needs to be rerun using more updates (see §4.4.7 for details).
- **Chains** (Fig. 13E) – MCMC "chains" are run in parallel, so helping to speed up computation. Each chain runs on a separate CPU core. Since modern computers typically have 4-8 cores, so the default number of chains is set to 3 (to avoid the computer slowing down too much). An additional advantage of running parallel chains is that it tells us something about the reliability of the results through the so-called "Gelman-Rubin" statistic (see §4.4.7).

4.3.1.2 PAS-MCMC

This generates a power posterior approximation that get successively closer to the true posterior over a series of generations. A "particle" denotes a combination of a parameter set and a system state. MCMC updates act on particles to change their state. In "particle annealed sampling" (PAS), particles initially map out the prior. Over successive generations (where particles are duplicated or culled) the algorithm passes from the prior to the posterior. Once this is complete, a sampling phase generates posterior samples in the same way as DA-MCMC. Requires:

- **Updates** – The same as DA-MCMC above.
- **Particles** – Sets the number of particles run in parallel. These help to stop the system getting stuck in metastable states in the burn-in phase. Particles become chains after burn-in.
- **Update per generation** – This sets how may MCMC updates are performed per generation during the burn-in phase. A large number implies slower annealing, but takes longer.

4.3.1.3 ABC

Approximate Bayesian Computation. Estimates the posterior by simulating from the model and comparing the output with the data (population-based data only). Only those samples that are sufficiently close (as measured by an error function) are used for the posterior approximation. For simplicity the error function is defined as minus the log of the observation probability. Requires:

- **Samples** – Sets the number of posterior samples to be generated.
- **Acceptance** – This gives the faction of simulated sample retained as posterior samples (must be between zero and one). Smaller values yield better results, but take computationally longer.

4.3.1.4 ABC-SMC

Approximate Bayesian Computation using Sequential Monte Carlo (ABC-SMC) is a version of ABC that improves posterior estimates over a series of generations (population-based data only). Requires:

- **Samples** – Sets the number of posterior samples to be generated.
- **Generations** – This gives the number of generations over which the posterior estimate is improved (typically between 5 and 10).

4.3.1.5 PMCMC

Particle MCMC targets the true posterior and makes use of multiple "particles" (stochastic simulations from the model) run in parallel.

4.3.1.6 Emulation

Here an emulator is used to approximate the posterior surface. This can be much faster than other approaches, provided the number of model parameters is not too large.

4.3.2 Further options

Additional features can be specified by clicking on "Further options" (Fig. 13H):

- **Output parameter samples** (DA-MCMC and PAS-MCMC) – Because MCMC samples are correlated, it makes sense to thin the results (to limit computational processing and memory requirements). BICI works by specifying a certain number of output parameter samples to be generated, which are drawn equally along the MCMC chain(s). Consequently, increasing the number of MCMC updates does not change the number of output parameter samples generated, but it does reduce any potential correlations between those samples. By default, 1000 parameter samples are generated. Selecting a higher value yields smoother posterior distribution plots, but requires more memory and CPU processing.
- **Output state samples** (DA-MCMC and PAS-MCMC) – As above, but for state samples. By default, 200 state samples are generated (note, this is significantly lower than the default number of parameter samples due to the larger memory requirement needed to represent each state). Selecting a higher value yields less noisy plots for populations, transitions and individuals, but requires more memory and CPU processing.
- **Burn-in** (DA-MCMC) – The burn-in phase in MCMC is the initial period that moves from a starting random guess for parameter values (typically sampled from the prior) to a sample representative of the posterior.
  The burn-in period is usually specified as a percentage of the overall number of MCMC updates (by default, 20% is used).
  Annealing - This an approach that can help to stop MCMC becoming stuck in metastable states during the burn-in phase. This introduces an "inverse temperature" parameter ϕ that determines to what extent the data is incorporated into the analysis. When ϕ =0, MCMC maps out the prior and when ϕ=1, the normal posterior is generated.
  Annealing works by gradually varying ϕ from 0 to 1 (at which point the posterior is then sampled). Precisely how this is done can be selected from the following options:
  - *none* – No annealing is performed and ϕ=1 during burn-in.
  - *scan* – ϕ is optimally scanned using a specified 'rate'. Note, here the burn-in period depends on this rate (with smaller values yielding more gradual annealing, but taking computationally longer), rather than a percentage.
  - *log-auto* – An automatic logarithm model is applied to ϕ.
  - *power-auto* – An automatic power-law model is applied to ϕ.
  - *power* – A power law model is applied to ϕ (here the 'power' property must be set).
- **Acceptance fraction** (ABC-SMC) – In ABC-SMC, first samples are generated and then a cut-off in the error function is automatically chosen such that a specified fraction of samples make up the next generation. Smaller values take computationally longer, but with fewer generations required.
- **Kernel size** (ABC-SMC) – Within ABC-SMC particles are selected from the previous generation with their parameter values perturbed by a kernel. Here the kernel is set to be a multivariate-normal approximation to the posterior, scaled by a selected size. A size of 0.5 works well in most cases.

- **Individual max** – When one or more species use an individual-based model, this sets the maximum number of individuals allowed (exceeding this limit results in an error message). Such a threshold is introduced to stop BICI running out of memory when the individual number diverges. By default, this is set to 20,000 individuals. The text box can be used to make this limit higher, if required.
- **Parameter output** – If the number of elements in a tensor exceeds this value, results are not output. This limit is introduced to stop very large output files when the tensor is not of direct interest (for example it might avoid the distance matrix being output every MCMC iteration). By default, this is set to 1000, but this can be changed in the text box.
- **Set random seed** – Simulation and inference rely on pseudorandom number generators. These start with an initial value called a "seed". If the seed option is turned off (as it is by default), this seed is randomly generated by the BICI interface (in which case BICI will produced different results each time it is run). If the seed is set, however, BICI will consistently give the same results, despite being stochastic. The seed can take any value between 0 and 10,000.
- **Proposal synchronisation** – Set if proposal synchronisation is used. This can take the values "off" or "on" (by default set to "on"). MCMC chains can either be run entirely separately, or information can be used across chains to improve the adaptation of proposals (in which case each chain uses the same synchronised set of proposals). Note, synchronisation will only be actually implemented if it is possible, *e.g.* it is not possible when BICI is running locally on more than one core (due to the fact that local threads can't communicate).

4.3.3 Running time

Every effort has been taken to optimise the speed of BICI (*e.g.* (a) carefully calculating the change in likelihood as a result of a given MCMC proposal[22], (b) automatic simplification of equations, (c) using sequential memory access for the most computationally intensive parts of the algorithm, (d) adapting MCMC proposals to help optimise them, and (e) pooling the common parts of equations to avoid calculating quantities multiple times). However, the time it takes for BICI to run can be fast or slow, and depends on numerous factors. Here we provide some general features which determine running speed:

- Population-based models are generally faster than individual-based model.
- The time-step plays a crucial role, with longer time steps running faster but potentially introducing discretisation errors (see §3.3.1).
- When transition rates depend linearly on populations, run time can be substantially faster (this is because BICI makes use of this linearity to speed up many types of MCMC proposals). Fortunately, most spatial/demographic epidemiological models fall into this category.
- The speed of BICI has a complicated relationship with the amount of data. Having more data increases the computation per update, but also the size of the posterior in parameter/event space is shrunk, so leading to short mixing times.
- Partial confounding of model parameters can lead to much longer mixing times.
- Multimodality in the posterior. If many different parameter combinations yield dynamics with good agreement to the data, MCMC can struggle (*e.g.* becoming stuck in locally optimal solutions). Approaches like annealing and PAS-MCMC are aimed at mitigating against this, but some problems are still very challenging.

[22] Such that the entire likelihood isn't calculate under each proposal, which would be computationally very slow.

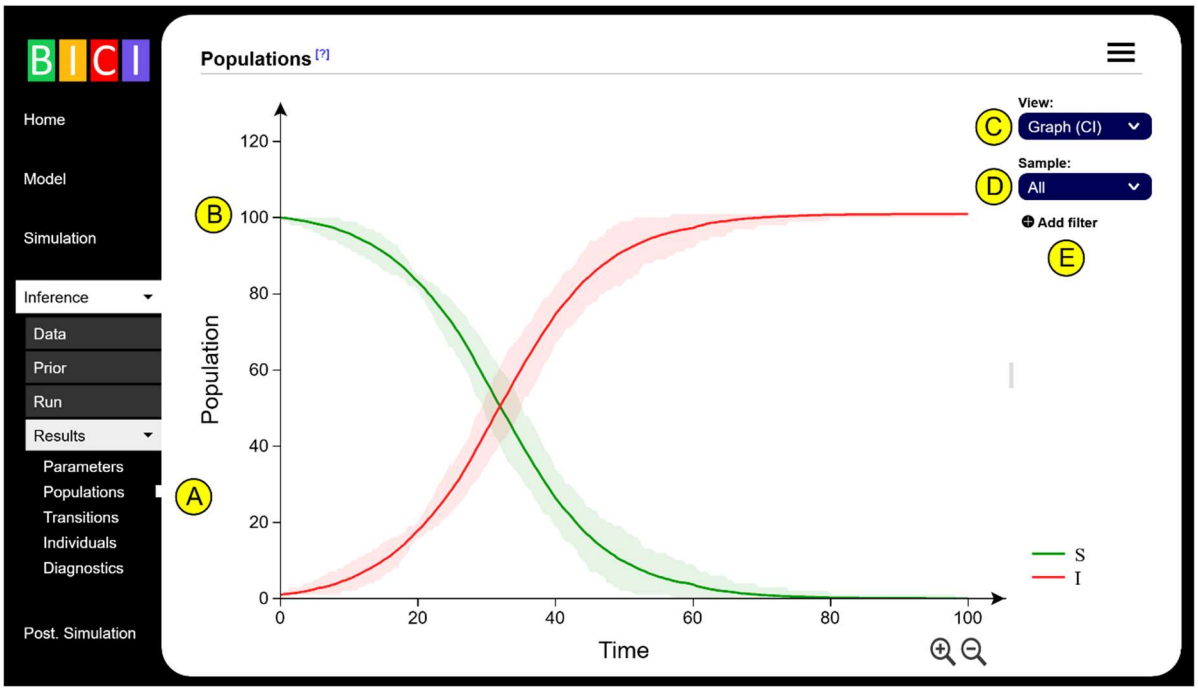


Figure 14 – *Results from inference*. A: Different types of output can be selected on the main menu, B: This shows an example of a posterior population distribution, C: Different ways to view the posterior can be selected, D: Individual samples can be selected, E: Filters can be added.

4.4 Results

Based on the data entered in §4.1 and prior knowledge from §4.2, it is generally not possible to identify model parameters with perfect precision (or transition events for that matter, unless they are specified by the data). Rather, there exists a distribution in these quantities known as the "posterior", which expresses both a best guess for parameters (*i.e.* posterior means) and range in values consistent with the data (*i.e.* credible intervals).

A description of various potential warning messages provided after code execution is given in Appendix G.

The "Inference→Results" page (Fig. 14) shows outputs to visualise this posterior (see Appendices H-K for examples). The menu bar (Fig. 14A) lists different possible features of the posterior to investigate, and these are discussed in the following sections:

4.4.1 Parameters

These plots provide visualisations and statistics for the parameter samples. Various options can be selected on the right-hand menu (depending on the situation):

- **View** – This list is divided into a number of different possibilities:
    - *Univariate* – These show results for all univariate parameters in the model.
    - *Multivariate* – Any parameters represented by vectors, matrices or tensors are given a separate section that can be viewed.
    - *Scatter* – Shows scatter plots of one parameter against another.
    - *Correlation* – Shows a matrix of correlation coefficients between different model parameters.

- *Likelihoods* – Shows different likelihoods and the prior (all log transformed) which add up to give the posterior probability.

- **Graph** – Provides alternative ways to view the selected parameter:
  - *Trace* – Shows consecutive parameter samples drawn from an MCMC chain.
  - *Sample* – Provides raw samples for ABC approaches.
  - *Distribution* – Shows an estimated posterior distribution for a given parameter. This can either be viewed through binning or using kernel density estimation (KDE) [2] (selected using the settings button in the bottom left-hand corner). Under inference the Bayes factor can be calculated by clicking on the "BF" button (see §4.4.6) [3].
  - *Statistics* – For each parameter provides the posterior mean, 95% credible interval as well as the effective sample size (ESS) and the Gelman-Rubin statistics (GR) (see §4.4.7 for details).

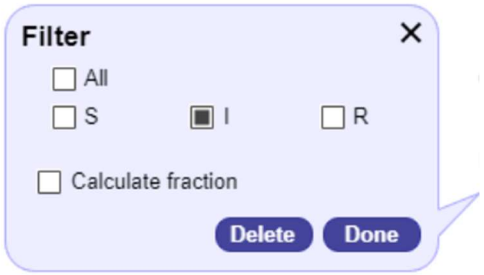
4.4.2 Populations

An example is shown in Fig. 14B, which shows the posterior distribution for the populations in different compartments for a selected classification as a function of time (means are shown by the solid lines and the shading represents 95% credible intervals).

Various options can be selected on the right-hand menu (depending on the situation):

- **View** (Fig. 13C) – Determines how the results are plotted:
  - *Graph (CI)* – Gives the population means, with shaded regions providing 95% credible intervals.
  - *Graph (line)* – Plots lines for each sample.
  - *Slice* – Provides posterior distributions at a specified time.
  - *Compartment* – Plots the compartmental model, with colours representing the populations in different compartments (*e.g.* useful for looking at the spatial spread of disease).
  - *Density* – Like 'Compartment' but expands the shading using Voronoi tessellation (*e.g.* good when plotting geographical points).
  - *Data* – Allows for inferred populations to be compared to measured population data.
- **Sample** (Fig. 13D) – Allows for individual MCMC samples to be viewed.
- **Time** – Specifies the time used with the 'Slice' option.
- **Classification** – Select the classification being viewed.
- **Chain** – Select to filter results for a specific MCMC chain.

Population filters – The "⊕ Add filter" option (Fig. 14E) allows for filters to be added (*e.g.* to focus on individuals in a given disease state or at a specific location). Once a classification has been chosen, the checkboxes allow the user to specify only certain compartments (*e.g.* in the example on the right, only the infected 'I' individuals are selected).



Clicking on the "Calculate fraction" checkbox means results are shown as a fraction given by the ratio of the population in the select compartment(s) compared to the population in all of them (this is particularly useful for plotting quantities such are the disease prevalence).

4.4.3 Transitions

These plots show how the rates in different transitions vary as a function of time.

Various options can be selected on the right-hand menu (depending on the situation):

- **View** – Determines how the results are plotted:
  - *Graph (CI)* – Gives the rate means, with shaded regions providing 95% credible intervals.
  - *Graph (line)* – Plots lines for each sample.
  - *Data* – Allows for the inferred transition rate to be compared to measured population-level transition data.
- **Sample** – Allows for individual samples to be viewed.
- **Classification** – Select the classification being viewed.
- **Time-step** – Selects the time-step over which rates are estimated (larger values generate smoother graphs, but sacrifice temporal accuracy).
- **Chain** – Select to filter results for a specific MCMC chain.

The "⊕ Add filter" option allows for population filters to be added (see previous section).

4.4.4 Individuals

Various options can be selected on the right-hand menu (depending on the situation):

- **View** – Determines how the results are plotted:
  - *Timeline* – Gives timelines for each individual in the system. These plots show how the compartmental status of different individuals in the system change as a function of time. The colours on the timelines correspond to those used for the compartments in the model. Individuals can filtered using the "Add filter" button or by focusing on a specific set of individuals using the "Ind. Group" dropdown menu.
  - *Ind. Eff.* – In cases in which the model uses individual effects, several different types of plots can made (see Graph below).
  - *Table* – Provides information about each individual (such as individual-level data, covariates for fixed effect or posterior estimates for individual effects).
  - *Statistics* – When there are individual effects in the system, this provides information about the means of those effects (and also the mean of the log of the individual effects).

- *Phylo. Tree* – Shows a phylogenetic tree (available only when genetic data is added to the analysis).
- *Trans. Tree* – Shows a transmission tree (available only when transmission tree is turned on within the species). See §2.1.1 for details.
- *First Inf*. *Time* – Shows the posterior probability distribution for the first infection time (when transmission tree).
- *First Inf*. *Ind.* – Shows the distribution for the first infected individual (when transmission tree).
- *First Inf*. *Comp.* – Shows the distribution in compartment for the first infection (when transmission tree).

- **Graph** – This shows possible different visualisations for individual effects:
  - *Pred. Acc* – In cases in which known individual effects have been loaded up, this shows how well the inferred individual effects agree with the true values. This is summarised with a prediction accuracy, which is a Pearson correlation between the logs of the individual effects and their true values.
  - *Dist*. *(log)* – This provides the distribution in the mean of the log of individual effects.
  - *Dist*. *(norm)* – This provides the distribution in the mean of individual effects.
  - *Scatter* – In cases in which there are more than one individual effect, this can be used to see how different effects are related.
- **Classification** – Select the classification being viewed.
- **Sample** – Allows for individual samples to be viewed.
- **Chain** – Select to filter results for a specific MCMC chain.
- **Ind**. **Group** – In cases in which specific groups have been identified (see §4.1.4.2) this allows the user to select which group to view.
- **Sort** – In the case of individual effects, table entries can sorted either from low-high or high-low. For example, this can be used to reveal which individual are most susceptible or least infectious.
- **Colour** – In the case of scatter plots for individual effects, different colouring on points can be usefully used to identify potential issues with the model (*e.g.* if different groups cluster instead of being random draws from the overall distribution).

Individual posterior plots – Under the 'timeline' view, clicking on one of the ID names on the left-hand side allows for plots looking at that specified individual's posterior. These show time variation in the posterior probability of being in different compartments. Various options can be selected on the right-hand menu (depending on the situation):

- **View** – This is divided into a number of different possibilities:
  - *Graph* – Gives the mean posterior probability, with shaded regions providing 95% credible intervals.
  - *Compartment* – Plots the mean posterior probability using the compartmental model (here colour is used to represent compartments with high probability).

- *Density* – Like 'Compartment' but expands the shading using Voronoi tessellation.

- **Sample** – Allows for individual samples to be viewed.
- **Classification** – Select the classification being viewed.
- **Chain** – Select to filter results for a specific MCMC chain.

4.4.5 Diagnostics

This provides a variety of visualisations to determines how well the model is fitting the data.

Various options can be selected on the right-hand menu (depending on the situation):

- **View** – Determines what is plotted:
  - *Trans. (exp.)* – Shows how transition rates in the posterior compare to rates estimated from the model. Systematic differences can help to pinpoint misspecification in the model.
  - *Trans. (dist.)* – Shows the cumulative probability distribution (see below) for a given transition. If the model is entirely correct, this distribution should be uniform between zero and one. Deviation from uniform indicates misspecification in the model. If the distribution is skewed to the right-hand-side it indicates that transitions have a faster rate (or shorter duration) compared to what would be expected from the model, and *vice-versa*.
  - *Trans. (bias)* – Shows the degree of skewness in the cumulative probability distribution, as measured by the first moment. A positive value implies that the transition rate in the posterior is in excess of the rate predicted by the model, and *vice-versa*.
  - *Trans. (p-val)* – Deviations from uniform for the cumulative probability distribution can indicate model misspecification. Such deviations do not necessarily affect the skewness (*e.g.* fitting an exponential distribution which in reality is gamma distributed) and so an alternative measure is provided. Here the distribution is split into 10 equal divisions, and a p-value is calculated using Pearson's chi-squared test. This plot shows the posterior distribution in -log(p-value), where large values indicate significant deviation from the model (the dashed line represents a notional p-value of 0.05).
  - *Proposals* – Provided information about MCMC proposals and other diagnostic information.
- **Sample** – Allows for individual samples to be viewed.
- **Time-step** – Selects the time-step over which rates are estimated (larger values generate smoother graphs, but sacrifice temporal accuracy).
- **Classification** – Select the classification being viewed.
- **Chain** – Select to filter results for a specific MCMC chain.

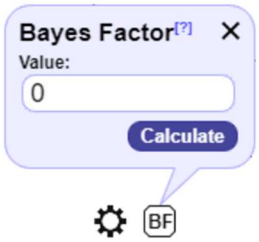
The "⊕ Add filter" option allows for population filters to be added (see §4.4.2).

Cumulative probability — The calculation of this quantity depends on the type of model:

- **Individual-based** – Here each individual transition is sampled from a distribution. The cumulative probability represents the probability of sampling a time later than the realised transition time.

- **Population-based** – The realised number of transitions within a timestep is sampled from a Poisson distribution with a given mean. In this case, the cumulative probability is the probability of obtaining this realised number or fewer.

4.4.6 Estimating the Bayes factor



A Bayes factor (BF) is the ratio of the likelihood for one particular hypothesis to the likelihood for another [3]. The BF comparing the full model to one in which a particular parameter is fixed (usually to zero, such that the parameter is effectively removed from the model) can be calculated using the "BF" button under a posterior parameter distribution (see §4.4.1). This is one way to determine which parameters in the model are redundant, so allowing for the model to be simplified in a step-wise fashion. A BF between 3 and 10 represents moderate evidence for one model over another, and exceeding 10 is considered strong [4].

4.4.7 MCMC diagnostics

Diagnostics provide an objective way of determining if a sufficient number of MCMC updates have been iterated to reliably trust the results (if this is not the case, the number of updates in Fig. 13D must be increased and the analysis re-run). Two different measures are shown on the parameter statistics page (see §4.4.1):

- **Effective sample size** (ESS) – This takes into account the fact that successive MCMC samples are correlated, and provided an estimate for the effective number of independent draws from the posterior. An ESS exceeding 200 is considered indicative of good mixing. Note, the ESS is not guaranteed to monotonically increase as the number of MCMC updates goes up (in fact if it is less than 100 it often fluctuates wildly).
- **The Gelman-Rubin statistic** $\hat{R}$ – On top of poor mixing, another difficulty faced by MCMC is multimodality. This characterises a situation in which one MCMC chain ends up in one posterior mode (and perhaps mixes well), but another goes to a completely different mode. One way to check for this problem is to calculate the Gelman–Rubin convergence diagnostic $\hat{R}$ (otherwise known as the "potential scale reduction factor") [5]. This is defined as the ratio of the overall pooled variance (*i.e.* calculated using samples taken from all chains combined) to the mean variance within each chain. Since MCMC is initialised randomly in parameters space, $\hat{R}$ starts large and is expected to approach one after many MCMC updates. Values less than 1.05 are considered to be indicative of convergence. If, however, $\hat{R}$ is larger than this threshold (even after a very large number of updates) it points to the existence of multimodality. Under these circumstances the results from BICI cannot be trusted. Note, $\hat{R}$ relies on comparing independent MCMC chains, and so cannot be calculated for a single chain.

For easy identification, statistics which are not converged are shown in orange (close to convergence) and red (poor convergence).

4.4.8 Extending inference

Often when performing inference using MCMC it is found that diagnostics (from previous section) tell us it has not converged. How frustrating! However, all is not lost (at least for the DA-MCMC and PAS-MCMC algorithms). Rather than starting the inference from the scratch, the user can extend the inference to give more MCMC updates until (hopefully) convergence is achieved. This can either be expressed as a new (higher) number of updates, or as a percentage, *e.g.* 200% means that twice as many MCMC updates are

generated (*i.e.* the existing ones plus the same number again). Corresponding to the additional updates, the number of parameter and state samples taken from the posterior are also increased.

Having the opportunity to extend MCMC makes it tempting to start with a very small number of updates and incrementally extend until convergence is achieved. Such an approach, however, is not advised. This is because BICI uses the initial burn-in period to optimise MCMC proposals. If this period is too short, the proposals may be inefficiently optimised (resulting in a much larger number of updates overall). Visual inspection of the trace plots can be a way to assess if burn-in has been successful.

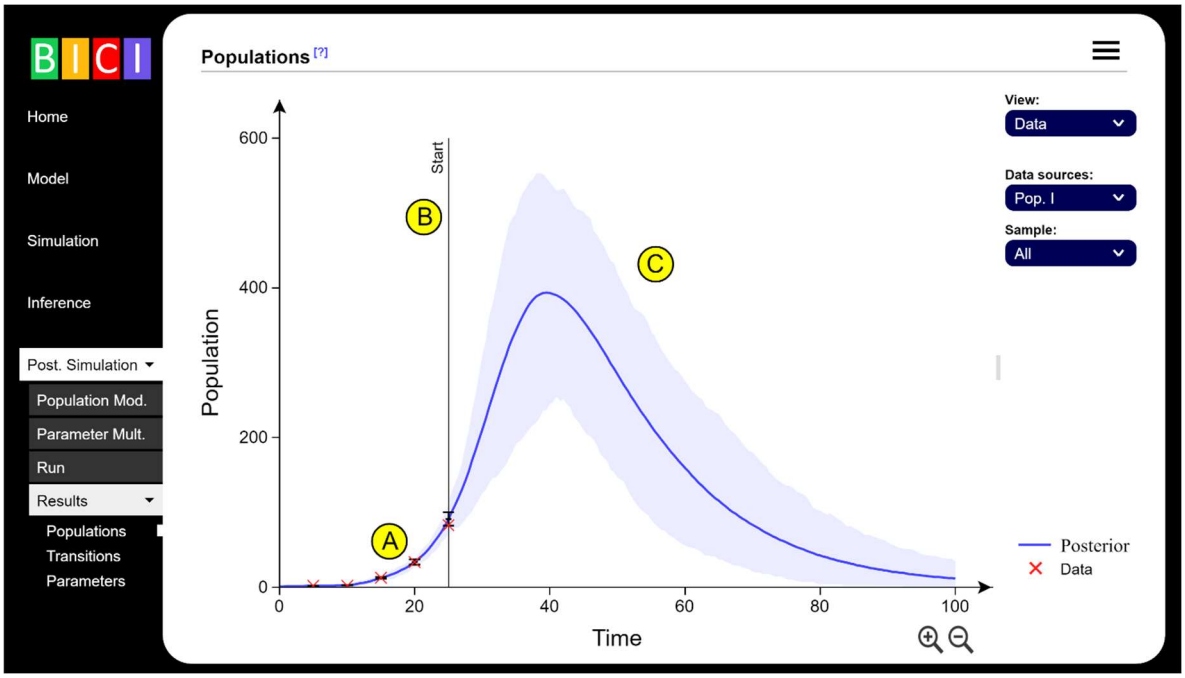


Figure 15 – *Posterior simulation*. This example shows future predictions. A: The red crosses represent hypothetical data from mid-way through an epidemic (this data gives the estimated number of infected individuals in the population), B: This vertical line represent the present time, C: This represents the predicted future dynamics of the epidemic (solid line represents mean and the shading represents a 95% probability region).

5) Posterior simulation

So-called "posterior simulations", simulate from the model whilst making use of the parameter and state samples generated under inference. They can be used in four different contexts:

- **Posterior predictive check** – This is used to check if the model is replicating the dynamics of the real system. It's not a formal model fit diagnostic, but rather more of a sanity check to make sure the model is reasonable. Here the model is simulated using the posterior parameter samples and the output is compared against the data used for inference. If this data lies within the 95% probability region, it is indicative the model is a good one. If not, an alternative model may be sought.
- **Future prediction** – As above, but here the end time is extended such that future behaviour is estimated based on current data. An example of this is given in Fig. 15.
- **Scenario analysis** – As above, but with future parameters modified to see what effect this has on system dynamics.

- **Counterfactual analysis** – This takes all the data (*e.g.* after an epidemic) and asks the question, how would things have turned out differently if certain interventions had been made?

5.1 Population modification

The "Post. Simulation→Population Mod." page allows for addition, removal and enforced movement of individuals on top of that used for the inference model (see §3.1 for details).

5.2 Parameter multipliers

The "Post. Simulation→Parameter Mult." page allows for parameter "multipliers" to be added. A multiplier is a (potentially) time-varying factor that multiplies a specified parameter within the model. They are defined in the same way as splines (see §1.4.5 for an example).

5.3 Run

The "Post. Simulation→Run" page allows the user to set various options before starting a posterior simulation:

- **Time range** – Sets the period over which the posterior simulation is performed. How this is chosen is dependent on the analysis being performed:
    - *Posterior predictive check* – This time range would be the same as for inference (this is the default setting).
    - *Future prediction and scenario analysis* – The start time would be set to the end time of inference and the end time would be extended to a time point in the future.
    - *Counterfactual analysis* – The start time would be set to the time of a proposed intervention and the end time would be the same as for inference.
- **Simulation number** – This sets how many posterior simulations are performed. Because simulations are stochastic, so each will provide a different set of results (by default, 200 are generated).

5.4 Results

The "Post. Simulation→Results" page shows results from posterior simulation. The outputs are essentially the same as for inference, so please refer to §4.4 for details (and see Appendices H-K for examples).

A description of various potential warning messages provided after code execution is given in Appendix G.

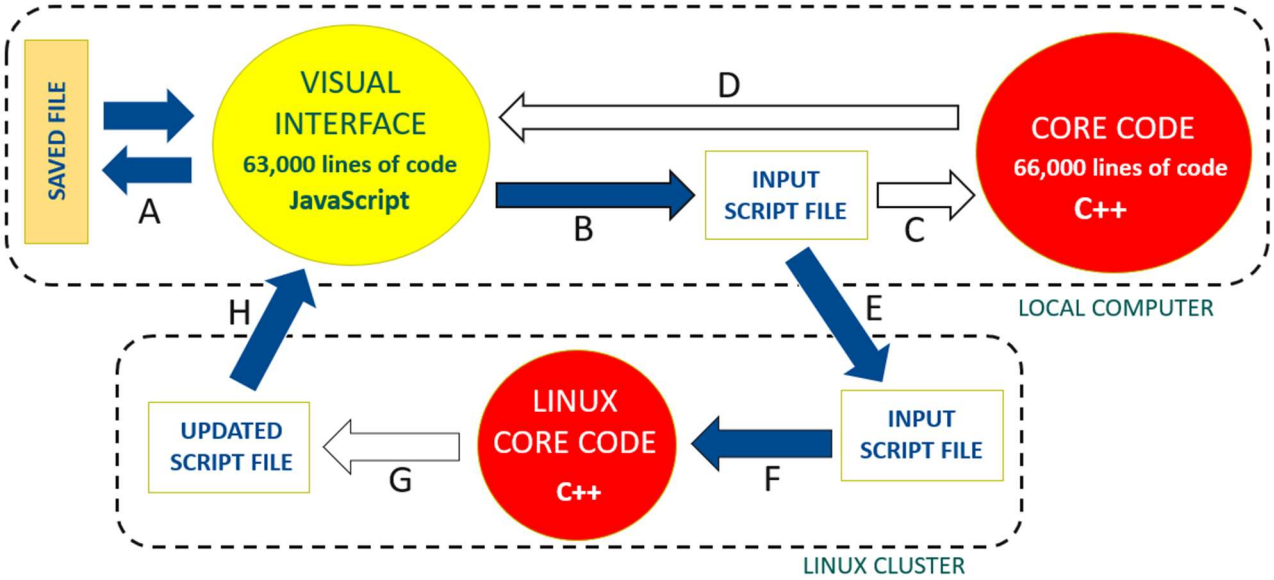


Figure 16 – *BICI architecture*. This diagram represents how BICI works. Arrows show interactions between different elements (solid/empty correspond to user initiated or not). A: Files can be loaded and saved from the interface in '.bici' format, B: When the "Start" button is pressed the interface creates an input BICI-script file, C: When running on the local machine, this file gets loaded by the core code, which performs the analysis, D: The core code seamlessly passes the results back to the interface for visualisation, E: If the code is to be run on a Linux cluster, the script file is copied to the cluster, F: Execution is initiated using a terminal command (see §6.4), G: The core code is run and the results are embedded into the script file, H: The script file is copied back to the local machine and loaded into the interface for visualisation.

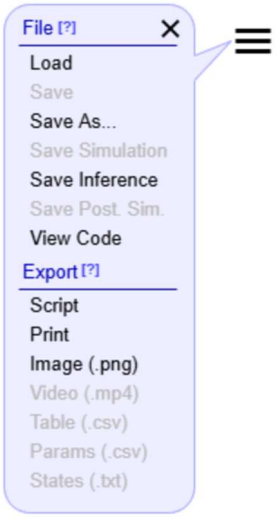
6) Inputs and outputs

6.1 BICI architecture

The diagram in Fig. 16 shows how BICI works. The code is actually split in two:

- **Visual interface** – This is written in JavaScript and runs on the desktop by means of NW.js.
- **The core code** – Performs analysis when BICI is executed (*i.e.*, by clicking on the "Start" button). This is written in highly efficient C++ code.

The various arrows in the diagram show routes by which the different elements interact.



6.2 Loading and saving BICI files

Loading and saving is done from the drop-down menu in the top right-hand corner of the page. The following file options are available:

- **Load** – This can be used to load a model/analysis in '.bici' format.
- **Save** – Saves the model and data in '.bici' format using the current path (note, this doesn't save any results, so the file size is relatively small).
- **Save As...** – Saves the model and data in '.bici' format using a specified path.

- **Save Simulation** – Saves simulation results (*i.e.* any parameter and state samples), along with the model/data used to generate them.
- **Save Inference** – Saves inference results, along with the model/data used to generate them (note, these files can be large because they store many samples of the model state).
- **Save Post. Sim.** – Saves posterior simulation results, along with the model/data used to generate them and inference samples.
- **View Code** – Used to view the code for the currently loaded model.

6.3 Exporting

Exporting from BICI is done from the drop-down menu in the top right-hand corner.

The following export options are available:

- **Script** – This is like "Save", but in addition to a '.bici' script file, a data folder is created that contains all the data (this makes the BICI-script more readable).
- **Print** – Prints the graph/plot/table currently being viewed.
- **Image** – Creates an image file for the graph/plot/table currently being viewed (in '.png' format).
- **Video** – Creates a video file for the animation currently being viewed (in '.mp4' format).
- **Table** – Creates a file for a table currently being viewed (in '.csv' format).
- **Params** – Exports posterior parameter samples.
- **States** – Exports posterior state samples.

6.4 Running on a Linux cluster

On the "Simulation→Run", "Inference→Run" and "Post. Simulation→Run" pages there is the option to "Run on local machine". If this is selected as "No" it is assumed BICI is to be run on a Linux cluster.

To run BICI in this way, the following steps must be followed:

- **Create BICI file** – Click on the "Start" button (Fig. 13G) to create and save the BICI-script file. Copy this to the cluster where BICI is going to be run.
- **Executable** – The executable "bici-para" must also be copied from the BICI main folder to the cluster (*e.g.* this could be in the same directory as the BICI file).
- **Run** – BICI is run using:

  mpirun -n [core] ./bici-para [file.bici] [run]

  where:

  - [core] is replaced with the number of CPU cores (this is typically the number of MCMC chains or PAS particles, but may be different if, e.g., more than one chain is run per core).
  - [file.bici] is replaced by the name of the BICI file that was created.
  - [run] is replaced with "sim", "inf" or "post-sim", depending on what is being run.

  Note, if the error "mpirun: command not found..." is encountered, MPI can be loaded with the terminal command "*module load mpi/openmpi-x86_64".*

If the time bars do not update as BICI is executed then running with “mpirun --output :raw” may solve the problem.

- **Visualise** – Once BICI has run, it embeds the results into [file.bici]. Copy this back to your local computer and load using the BICI interface.

```
####################### DATA DIRECTORY ######################

data-dir folder="EX_M1-1-data-files"

####################### DESCRIPTION #########################

description text="description.txt"

######################### DETAILS ###########################

simulation start=0 end=100 timestep=0.5

inference start=0 end=100 timestep=0.5 nchain=3

post-sim start=0 end=100

######### DEFINE MODEL AND DATA FOR SPECIES PEOPLE #########

species name="People" type="population"

# SPECIES MODEL

class name="DS" index="a"

comp name="S" color="#009900" x=-10 y=0
comp name="I" color="#ff2222" x=10 y=0

trans name="S->I" value="exp(rate:β×{I})"

# SIMULATION INITIAL CONDITIONS

init-pop-sim type="fixed" focal="DS" file="init-pop-sim-People.csv"

####################### PARAMETERS ######################

param name="β" value="0.001" prior="uniform(0,0.01)"
```

Figure 17 – *Example of BICI-script*. This example is taken from M1.1 in §8. See §7.1 for a description of the various commands (note, details on all BICI commands are given in Appendices A, B and C).

7) BICI-script

BICI-script is the means by which models and analyses are stored and processed. A typical script contains commands which set up the compartmental model structure, add model parameters and priors and

incorporate any available data. When run using the core BICI code, the results are added into the script itself (overwriting any previous results) such that they can be read back into the visual interface.

Appendix A provides a comprehensive list of BICI commands, ordered by section. Appendix B gives an alphabetically ordered index of commands for easy reference. Appendix C provides some illustrative examples.

Inspecting the BICI-script files in the 'Examples' directory can be a good way to help understand setting up different types of model (see §8).

7.1 A simple example

Figure 17 shows a simple BICI-script taken from M1.1 in §8. Different commands inform different parts of the model. This sets up a population-based species 'People' and within this creates a classification 'DS' (short for 'disease status'). It adds compartments S and I into 'DS', and a transition going between the two with rate '$\beta \times \{I\}$'.

A brief description of the various commands in Fig. 17 is as follows:

data-dir – Informs the location of the data directory, which stores any data files (note, the path for this directory can either be specified relative to the BICI-script or using a full path name[23]). The 'data-dir' command can be omitted when all data is embedded within the script (see §7.2).

description – Provides a text description of the analysis (optional). Here this is provided within the file 'description.txt':

```
# Model 1.1: Simple SI model

## Objective
- Introduce the simplest possible population-based epidemiological model.

## Model
- A single population-based species 'People' is created.
- This contains a classification 'DS' which stands for 'disease status'.
- DS contains two compartments: susceptible $S$ and infected $I$. Together they are known as the
“SI model”.
- A transmission rate $β$ determines the rate at which individuals become infected.

## Population
- This consists of 100 initially susceptible individuals and one infected.
```

A markdown format is used when displayed in the visual interface (this allows for the following concise formatting: "# Title", "## Subtitle", "- Bullet point", "*italic text*" and "**bold text**"). Additionally, parameters can be added by enclosing within dollar symbols, *e.g.* "a" or "b^super_sub".

simulation – This informs the time range and time-step used under simulation. Note, to actually perform simulation it is necessary to use "sim" when executing the BICI core code (see §7.3).

inference – This informs the time range and time-step used under inference. To perform this, it is necessary to use "inf" when executing the BICI core code.

[23] In this example the data folder is called "EX_1-1-data-files", which is stored in the same "Examples" directory as the BICI-script file.

post-sim – This informs the time range used under posterior simulation (here the time-step is automatically set to that under inference). To perform this, it is necessary to use "post-sim" when executing the BICI core code.

species – Determines the name and type (population or individual-based) for a new species (see §2.1.1).

class – Determines the name and index[24] for a new classification (see §2.1.2).

comp – Determines the name of a new compartment within the classification (see §2.1.3). The colour and position information used by the visual interface are optional[25].

trans – Determines a new transition between two compartments (see §2.1.4).

init-pop-sim – Contains information about the initial population under simulation, as stored in the file 'init-pop-sim-People.csv' (see §3.1.1):

```
"Compartment","Population"
"S",100
"I",1
```

This states that the S and I compartments start with 100 and 1 individuals, respectively.

param – Provides information about a model parameter. Here 'value' would be used under simulation and 'prior' under inference (note, posterior simulation uses parameter samples generated under inference, hence values don't need to be specified).

Note, many of the commands above reply on default settings that can be changed. For example, the default setting under inference uses the DA-MCMC algorithm, but other methods are available. See Appendix C for a collection of BICI-script examples that help to illustrate many of the different possibilities.

For the most part, BICI commands can be declared in any order. A notable exception is when the command relates to a specific species or classification. For example, "comp" and "trans" add compartments and transitions to a specific classification, and so rely on that classification being declared earlier in the BICI-script[26].

7.2 Formatting

In terms of formatting, BICI-script can be written in Unicode or plain ASCII (which allows for simplicity of typing and editing if working directly with the script). For the latter, Greek characters can be added by using Latex notation, *e.g.* '\beta' becomes 'β' when loaded into BICI. Furthermore, the '*' character can be used instead of '×' and left '⟨' and right '⟩' triangular brackets (used for fixed effects) can be substituted by the greater and less than symbols '<' and '>'. See §2.1.12 for naming restrictions on species, compartments *etc*...

In the example in Fig. 17, a data directory is specified in which data files are separated from the BICI-script itself. This is usually the best way to directly work with BICI-script when using the command line instead of the visual interface (because the script is relatively concise). It's worth mentioning, however, that loading

[24] A single letter used in mathematical expressions to represent compartments within the classification.
[25] Note, position information is required if the distance matrix is used in the model.
[26] Note, the "set" command can be used to shift focus to a particular species or classification at any point in the script.

and saving BICI files from the interface is done using just a single '.bici' file. This is achieved by embedding any additional data files within the script itself (by surrounded the file text using "[[…]]" quotes). Selecting to export 'Script' from the main menu in the interface (instead of 'Save As') generates BICI-script with a corresponding data directory, which can subsequently be worked on from the terminal.

7.3 Directly running BICI core code

Whilst running BICI is very easy using the interface, there may be instances when the user might want to run BICI-script directly using the core code, i.*e.* bypassing the interface entirely. For example, BICI could be used as part of a pipeline in which the script is generated, run using the BICI core code, and the outputs used for further processing.

The core code can be run using the following command:

./bici-core.exe [file.bici] [run]

where:

- [file.bici] is the name of the BICI file.
- [run] is replaced with "sim", "inf" or "post-sim", depending on whether simulation, inference or posterior simulation are being run. The "ext" option can be used to extend inference for more updates (see below).

Once run, the outputs get incorporated into the '.bici file'. For example, if the script in Fig. 17 is run using:

./bici-core.exe example.bici sim

the following text will be added to the end of the file 'example.bici':

```
# OUTPUT

sim-param file="Sample/param_0.csv"

sim-state file="Sample/state_0.csv"
```

The command "param-sim" and "state-sim" store the parameters and states generated by the simulation (see §7.4.1 and §7.4.3 for formatting information). Because the 'data-dir' command is defined in Fig.17, so BICI creates a directory "Sample" into which the parameter and state samples are saved. Otherwise, they are embedded directly into the BIC-script (see §7.2).

Note, running a new simulation deletes any existing "param-sim" and "state-sim" commands (and similarly for inference and posterior simulation).

In the case of inference, a further command "param-stats-inf" stores posterior summary statistics for the parameters (see §7.4.2).

	A	B	C	D	E	F	G	H	I	J	K
1	State	β	L^markov	L^non-markov	L^ie	L^dist	L^obs	L^genetic-proc	L^genetic-obs	L^init	Prior
2	0	0.00622167	-47.5336	0	0	0	-7583.64	0	0	0	4.60517
3	16	0.000963481	-168.288	0	0	0	-14.2494	0	0	0	4.60517
4	31	0.000670784	-164.802	0	0	0	-13.6196	0	0	0	4.60517
5	46	0.00102211	-162.565	0	0	0	-14.8831	0	0	0	4.60517
6	61	0.000986288	-166.84	0	0	0	-13.4952	0	0	0	4.60517
7	76	0.000842911	-156.489	0	0	0	-12.1035	0	0	0	4.60517
8	91	0.000911668	-155.954	0	0	0	-12.7896	0	0	0	4.60517
9	106	0.000994754	-148.155	0	0	0	-11.767	0	0	0	4.60517
10	121	0.000951873	-149.254	0	0	0	-12.0442	0	0	0	4.60517

Figure 18 – *Parameter output file*. This is an extract from of a parameter output file generated using example B1 (.csv file viewed in Excel). The column "State" gives the MCMC update number from which the sample is taken (note, because of thinning this number goes up in steps of 5). Different rows give different posterior samples, for example parameter β has different sampled values going down the second column. Other columns related to log-likelihoods and the log of the prior probability (see §7.4.1 for details)

	A	B	C	D	E	F	G	H	I	J	K	L
1	name	mean	sd	CI min	CI max	ESS	GR					
2	\beta_M	0.62292	0.0157	0.59358	0.65519	394	1.00834					
3	\beta_F	0.39761	0.01041	0.37696	0.41777	515	1.00735					
4	\gamma_M	0.09855	0.00289	0.0929	0.10421	530	1.00125					
5	\gamma_F	0.20277	0.01006	0.18547	0.22357	400	1.00379					

Figure 19 – *Parameter statistics file*. This shows the parameter posterior statistics file, as obtained by performing inference on example B3 (.csv file viewed in Excel).

7.3.1 Extending inference

See §4.4.8 for a description of how extending inference can be used to ensure MCMC convergence. This is implemented by running BICI-script using the 'ext' option (or 'extend').

For example, the following command:

./bici-core.exe example.bici inf

could be used to perform inference (which generates 5000 updates) and then:

./bici-core.exe example.bici ext 10000

would extend that inference up to 10000 updates (*i.e.* double).

Alternatively, the extension could be expressed as a percentage. Here the equivalent command would be:

./bici-core.exe example.bici ext 200%

Once run, the number of updates in the 'inference' command gets automatically changed within the BICI-script (so the initial value 'update=5000' becomes 'update=10000'). Similarly, 'param-output' and 'state-output' are increased to double their previous value.

7.4 BICI-script outputs

7.4.1 Parameter samples

These are stored in the BICI-script using the commands "param-sim", "param-inf" or "param-post-sim" (depending on whether simulation, inference or posterior simulation have been performed).

Figure 18 shows an example taken from inference using B1 in §8. Here the first few posterior parameter samples generated are displayed (typically, the data table will contain 1000 rows in all).

7.4.2 Parameter statistics

These are generated under inference and stored in the BICI-script using the commands "param-stats-inf". Figure 19 shows an example taken from inference using example B3 in §8.

The columns give the parameter name (with any Greek letters replaced with Latex equivalents), posterior mean, standard deviation and 95% credible interval. The effective sample size (ESS) and Gelman-Rubin statistic (GR) are diagnostics used to determine if the posterior has been estimated accurately or not (see §4.4.7).

```
{
  timepoint 0:0.5:100
}

<<STATE 1000>>

<PARAMETERS>
"β",0.00104521
L^markov,-144.692
L^non-markov,0
L^ie,0
L^dist,0
L^obs,-31.4348
L^init,0
Prior,2.30259

<SPECIES "People">

<INITIAL POPULATION>
compartment,population
S,100
I,1

<TRANSITIONS>
S->I:Z10,1,Z18,1,Z9,1,Z8,1,Z14,1,Z3,1,Z3,1,Z1,1,Z5,1,1,Z1,2,Z3,1,1,Z1,1,3,1,2,Z1,2,1,1,1,Z2,1,1,2...
```

Figure 20 – *State output file (population-based)*. This is an extract from of a state output file. The curly brackets at the top denote header information (applicable to all samples) and the various tags specify different aspects of the state sample (see §7.4.3 for details).

```
{
  # Time divisions
  timepoint 0:0.5:100
  # Individual index
  0:Sim-People-Ind-1
  1:Sim-People-Ind-2
  2:Sim-People-Ind-3
    ⋮     ⋮     ⋮
}

<<STATE 1>>

<PARAMETERS>
"β",0.001
L^markov,-419.935
L^non-markov,0
L^ie,0
L^dist,0
L^obs,0
L^init,0
Prior,0


<SPECIES "People">

<INDIVIDUALS>
index,source,events
0,no,S:0 S->I:59.4155
1,no,S:0 S->I:66.0036
2,no,S:0 S->I:48.9163
   ⋮     ⋮     ⋮
```

Figure 21 – *State output file (individual-based)*. This is an extract from of a state output file. The curly brackets at the top denote header information (applicable to all samples) and the various tags specify different aspects of the state sample (see §7.4.3 for details).

7.4.3 State samples

These are stored in the BICI-script using the commands “state-sim”, “state-inf” or “state-post-sim” (depending on whether simulation, inference or posterior simulation have been performed).

The formatting for the state sample depends on whether it has been derived from a population or individual-based species:

Population-based sample – Figure 20 shows an example taken from inference using M1.1 in §8. The curly brackets at the top denote header information (which applies to all state samples). “timepoint” here indicates that the inference time goes from 0 to 100 in steps of 0.5.

“<<STATE 1000>>” is used to denote the fact the sample was taken after 1000 MCMC updates.

The various tags in Fig. 20 provides information about the sample:

- <PARAMETERS> – This stores the parameter’s value.
- <SPECIES> – Indicates which species is being referred to.
- <INITIAL POPULATION> – The initial population in different model compartments.
- <TRANSITIONS> – For each transition, this stores the number of individuals passing down the transition for each time-step. Note, this uses concise formatting, where the “Z” character followed by a number indicates the number zeros represented (such an approach drastically reduces file size when transitions are sparse).

Individual-based sample – Figure 21 shows an example taken from inference using M1.2 in §8. The header provides a “dictionary” that relates individual names to index numbers used in the state samples (this is done to reduce file size). The example shows that individual ‘Sim-People-Ind-1’ has an index 0.

Many of the tags are the same as above. However, rather than transition information being stored, here the dynamics of individuals are represented:

- <INDIVIDUALS> – This table provides information about the individual state. The first column “index” refers to the individual index, as referenced in the header information. The second column “source” determines if the individual originates from a source transition. The final column “events” gives the individual’s initial compartment and any subsequent events they undergo.

As can be seen, storing the system state typically requires a much larger files size than storing the model parameters. For this reason, state sample are usually thinned more than parameter samples.

In other circumstances, further tags can provide additional information:

- <POPULATION CHANGE> – Stores population changes from adding or removing populations
- <TRANSDISTPROB> – Stores information about cumulative probability distributions (used when displaying diagnostics).
- <DERIVED> – Information about time-varying derived variables.
- <TRANSTREE> – Stores the transmission tree (if pathogen genetic data has been added to the analysis this also captures the phylogenetic tree).

8) Examples

Example applications can be selected on the main page (see Fig. 1D). These demonstrate the versatility of BICI applied to a wide variety of different models and data scenarios. They can be altered or experimented with in any way, and new users are encouraged to try different possibilities as a way to familiarise themselves with the BICI interface (note, reloading examples from the main page returns them to their default settings). The source BICI-scripts for these examples can be found in the "Examples" directory.

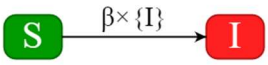
Sections M1- M6 go through different types of model. These examples can be simulated to see their dynamic behaviour (using different visualisations). Simulated data can be generated on the "Simulation→Generate Data" page (see §3.5), from which inference can then be performed.

M1 Simple epidemiological models

This section gives some examples of simple epidemiological models, illustrating both individual and population-based approaches.

M1.1: SI population-based model (PBM)

Purpose:
- Introduce the simplest possible population-based epidemiological model.
- This type of model keeps track of how the populations in the different compartments change over time.

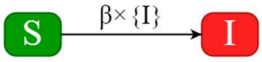


Model:
- A single population-based species "People" is created.
- This contains a classification "DS", which stands for "disease status".
- DS contains two compartments: susceptible S and infected I. Together they are known as the "SI model".
- A transmission rate β determines the rate at which individuals become infected.

System:
- 100 initially susceptible individuals and one infected.

M1.2: SI individual-based model (IBM)

Purpose:
- Introduce the simplest possible individual-based epidemiological model.
- Under simulation or inference, we see timelines for each of the individuals in the system.



Model:
- A single individual-based species "People" is created.
- This contains a classification "DS", which stands for 'disease status'.
- DS contains two compartments: susceptible S and infected I.
- A transmission rate β determines the rate at which individuals become infected.

System:
- 100 initially susceptible individuals and one infected.

M1.3: SIR model (IBM)

Purpose:
- Introduce the SIR epidemiological model.

Model:
- A single population-based species "People" is created.
- This contains a classification "DS", which stands for 'disease status'.
- DS contains three compartments: susceptible S, infected I and recovered R. Together they are known as the "SIR model".
- A transmission rate β determines the rate at which individuals become infected.

• A recovery rate γ determines the rate at which individuals recover.

System: • 100 initially susceptible individuals and one infected

M1.4: SIR model with Erlang distribution (PBM)

Purpose: • Introduce the Erlang distribution.

• In population-based models, transitions are normally required to be exponentially distributed.

• The Erlang distribution is a gamma distribution with a specified integer shape parameter.

• This can be used to more realistically model disease progression within individuals.

• Note, Erlang transitions must be unbranching.

Model: • A single population-based species "People" is created.

• This contains a classification "DS", which stands for "disease status".

• DS contains three compartments: susceptible S, infected I and recovered R.

• A transmission rate β determines the rate at which individuals become infected.

• A recovery duration m determines how long it takes for individuals to recover on average.

• The shape parameter is fixed to $k=3$.

System: • 100 initially susceptible individuals and one infected.

M1.5: SIR model with gamma distribution (IBM)

Purpose: • Introduce non-Markovian (NM) transitions.

• In individual-based models there is greater flexibility to specify the probability distribution over which transitions occur.

• This example uses a gamma distributed infectious period with a mean and coefficient of variation (which is a dimensionless number that specifies the standard deviation divided by the mean).

• Other possibilities include: Erlang, log-normal, Weibull, and fixed period.

• NM transitions can be used to more realistically model disease progression within individuals.

Model: • A single population-based species "People" is created.

• This contains a classification "DS", which stands for "disease status".

• DS contains three compartments: susceptible S, infected I and recovered R.

• A transmission rate β determines the rate at which individuals become infected.

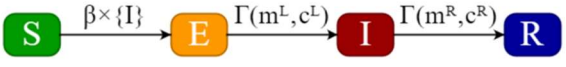
• A recovery duration m determines how long it takes for individuals to recover on average.

• A coefficient of variation parameter c determines the variation about the mean m.

System: • 100 initially susceptible individuals and one infected.

M1.6: SEIR model with exposed period (IBM)

Purpose: • Introduce a model with a so-called "exposed" compartment.



• This new compartment allows for a period of time after an individual becomes infected but before they are infectious.

Model: • A single population-based species "People" is created.

• This contains a classification "DS", which stands for "disease status".

• DS contains four compartments: susceptible S, infected I, exposed E and recovered R. Together they are known as the "SEIR model".

• A transmission rate β determines the rate at which individuals become infected.

• A latent duration m^L (with coefficient of variation c^L) determines how long, on average, individuals remain in the exposed state.
• An infectious duration m^I (with coefficient of variation c^I) determines how long, on average, individuals can pass on their infection to others.

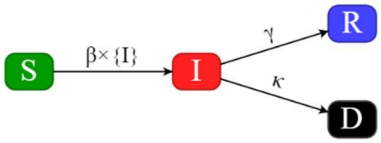
System: • 100 initially susceptible individuals and two infected.

M2 Additional epidemiological models

These examples look at some additional features, such as branching, reverse transitions and stratification.

M2.1: SIRD model with branching using transition rates (PBM)

Purpose: • Introduce branching in a population-based model.
• Here infected individuals can either recover or die.

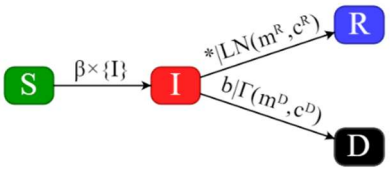


Model: • A single population-based species "People" is created.
• This contains a classification "DS", which stands for "disease status".
• DS contains four compartments: susceptible S, infected I, recovered R and dead D. Together they are known as the "SIRD model".
• A transmission rate β determines the rate at which individuals become infected.
• A recovery rate γ determines the probability per unit time an infected individual recovers.
• A mortality rate κ determines the probability per unit time an infected individual dies

System: • 100 initially susceptible individuals and two infected.

M2.2: SIRD model with branching probability (IBM)

Purpose: • Introduce branching in an individual-based model.
• Here infected individuals can either recover or die.
• A branching probability is used to determine if an individual dies.
• Note, one of the branches had probability "*" because it is derived from others (such that the total adds to one).



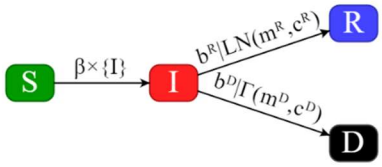
• In this example the transition distributions down the branches are different (one being log-normal and the other gamma distributed).

Model: • A single individual-based species "People" is created.
• This contains a classification "DS", which stands for "disease status".
• DS contains four compartments: susceptible S, infected I, recovered R and dead D.
• A transmission rate β determines the rate at which individuals become infected.
• Branching probability b determines the probability of dying.
• Mean m^R and coefficient of variation c^R specify the log-normally distributed period to recovery.
• Mean m^D and coefficient of variation c^D specify the gamma distributed period to death.

System: • 100 initially susceptible individuals and two infected.

M2.3: SIRD model with branching factors (IBM)

Purpose:
- Introduce branching factors in an individual-based model.
- Here infected individuals can either recover or die.
- Unlike branching probabilities, branching factors specify the *relative* probability of going down a branch (note, their sum does not need to add to one).
- They are implemented by clicking on the I compartment and selecting 'Use branching factors'.



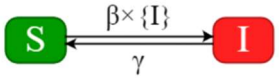
- Branching factors are useful, *e.g.*, for models in which time-varying covariates affect branching probabilities.
- The transition distributions down the branches are different (one being log-normal and the other gamma distributed).

Model:
- A single individual-based species "People" is created.
- This contains a classification "DS", which stands for "disease status".
- DS contains four compartments: susceptible S, infected I, recovered R and dead D.
- A transmission rate β determines the rate at which individuals become infected.
- Branching factors b^D and b^R determine the *relative* probability of dying or recovering.
- The probability of dying is $b^D/(b^D + b^R)$.
- Mean m^R and coefficient of variation c^R specify the log-normally distributed period to recovery.
- Mean m^D and coefficient of variation c^D specify the gamma distributed period to death.

System:
- 100 initially susceptible individuals and two infected.

M2.4: SIS model (PBM)

Purpose:
- Introduce a model that allows individuals to be infected multiple times.
- This can lead to endemic diseases.

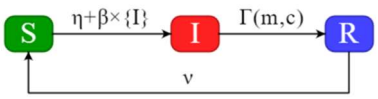


Model:
- A single population-based species "People" is created.
- This contains a classification "DS", which stands for "disease status".
- DS contains two compartments: susceptible S and infected I.
- The backward transition to the I state means that this model is known as the "SIS model".
- A transmission rate β determines the rate at which individuals become infected.
- A recovery rate γ determines the rate at which individual return to being susceptible.

System:
- 100 initially susceptible individuals and one infected.

M2.5: SIRS model with waning immunity (IBM)

Purpose:
- Introduce the concept of waning immunity.
- For many diseases recovery in not permanent.
- Due to mutations in the pathogen, an individual can become reinfected at a later date.



- This waning immunity is incorporated into the standard SIR model by a backward transition from R to S.
- Such a model can generate multiple epidemics.

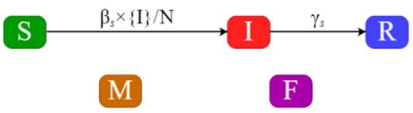
Model:
- A single population-based species "People" is created.
- This contains a classification "DS", which stands for "disease status".
- DS contains three compartments: susceptible S, infected I and recovered R.

• An external force of infection parameter η provides a small proportion of infections from outside the system.
• A transmission rate β determines the rate at which individuals become infected.
• A recovery duration m (and coefficient of variation c) determines how long, on average, it takes for individuals to recover.
• A waning immunity coefficient v determines the rate at which individuals become susceptible again.

System: • 100 initially susceptible individuals and two infected.

M2.6: SIR model with demographic stratification (PBM)

Purpose: • Introduce an example of stratification into the SIR epidemiological model.
• Here the population is divided into males and females.

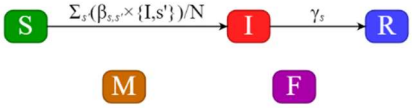


Model: • A single population-based species "People" is created.
• This contains classifications "DS", which stands for "disease status", and "Sex".
• "DS" contains three compartments: susceptible S, infected I and recovered R.
• "Sex" contains the compartments M and F for male and female.
• A transmission rate β_s determines the rate at which individuals with sex s become infected.
• A recovery rate γ_s determines the rate at which individuals recover with sex s.
• N is the total number of individuals (this model assumes frequency dependent transmission).

System: • 10,000 initially susceptible individuals and 2 infected.
• A 50/50 split between males and females.

M2.7: SIR model with differential infectivity and demographic stratification (PBM)

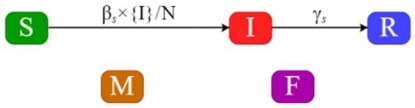
Purpose: • Introduce differences in infectivity in a stratified SIR epidemiological model.
• Here the population is divided into males and females.
• A 2×2 transmission matrix is used to describe transmission between these two demographic groups.



Model: • A single population-based species "People" is created.
• This contains classifications "DS", which stands for "disease status", and "Sex".
• "DS" contains three compartments: susceptible S, infected I and recovered R.
• "Sex" contains the compartments M and F for male and female.
• A transmission rate $\beta_{s,s'}$ determines the rate at which individuals with sex s become infected from individuals with sex s'.
• A recovery rate γ_s determines the rate at which individuals recover with sex s.
• N is the total number of individuals (this model assumes frequency dependent transmission).

System: • 10,000 initially susceptible individuals and 2 infected.
• A 50/50 split between males and females.

M2.8: SIR model with demographic stratification (IBM)



Purpose:
- The same as M2.6, but this uses an individual-based model.
- Introduce an example of stratification into the SIR epidemiological model.
- Here the population is divided into males and females.

Model:
- A single population-based species "People" is created.
- This contains classifications "DS", which stands for "disease status", and "Sex".
- "DS" contains three compartments: susceptible S, infected I and recovered R.
- "Sex" contains the compartments M and F for male and female.
- A transmission rate β_s determines the rate at which individuals with sex s become infected.
- A recovery rate γ_s determines the rate at which individuals recover with sex s.
- N is the total number of individuals (this model assumes frequency dependent transmission).

System:
- 10,000 initially susceptible individuals and 2 infected.
- A 50/50 split between males and females.

M3 Spatial epidemiological models

These models incorporate spatial stratification in different ways.

M3.1: Metapopulation model using geographical regions (PBM)

Purpose:
- Introduce a spatial epidemiological model that uses regions (these are bounded areas on a map).
- These boundaries are loaded using GeoJSON format (a commonly used format, often available to download for geographical, political and administrative boundaries).
- Here Scotland is divided into 32 local authorities.
- A constant matrix of interactions is derived from census commuting data.

Model:
- A single population-based species "People" is created.
- This contains classifications "DS", which stands for "disease status", and "Location".
- "DS" contains three compartments: susceptible S, infected I and recovered R.
- "Location" contains 32 regions.
- A transmission rate β determines the rate at which individuals transmit disease.
- A constant matrix of interactions $M_{l,l}$, is proportional to the contact rate between individuals in the same and different regions (this can be estimated from census commuting data).
- A recovery rate γ determines the rate at which individuals recover.
- An external force of infection η is used to initiate an epidemic.

System:
- Initially susceptible individuals, with regional populations taken from census data.

M3.2: Metapopulation model using geographical points (PBM)

Purpose:
- Introduce a spatial epidemiological model that uses geographical points (these have a defined longitude and latitude).
- The 100 largest cities in the world are incorporated into the model.

Model:
- A single population-based species "People" is created.
- This contains classifications "DS", which stands for "disease status", and "Location".
- "DS" contains three compartments: susceptible S, infected I and recovered R.
- "Location" contains compartments for 100 world cities.
- A transmission rate β determines the rate at which individuals transmit disease within a city.
- Parameter λ controls the spread of disease between cities.
- A recovery rate γ determines the rate at which individuals recover.
- Constant matrix $M_{l,l'}$ is proportional to the interactions between cities (*e.g.*, this can be obtained from movement data).
- N_l is the population of city l.
- P is the global population.

System:
- Initially susceptible individuals, with city populations taken from census data.
- Two infected individuals start in Tokyo.

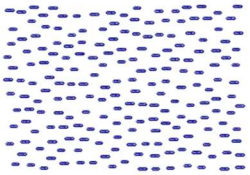
M3.3: Metapopulation model using a distance kernel (PBM)

Purpose:
- Introduce spatial kernels.
- A kernel is a specified function that gives the probability the disease is transmitted a certain distance.
- Typically, they have a high value at close distance, but decay away to zero in the large distance limit.
- This example uses a so-called "power-law" spatial kernel.
- To increase computational speed, the kernel is truncated to a maximum distance of 200km.
- The 100 largest cities in the UK are incorporated into the model.

Model:
- A single population-based species "People" is created.
- This contains classifications "DS", which stands for "disease status", and "Location".
- "DS" contains three compartments: susceptible S, infected I and recovered R.
- "Location" contains 100 geographical points.
- A transmission rate β determines the rate at which individuals transmit disease within a city.
- The parameter λ determines the rate at which individuals transmit disease between cities.
- η is an external force of infection.
- A recovery rate γ determines how fast individuals recover.
- A matrix $M_{l,l'}$ gives the transmission between cities. This is reparameterised as a power-law distribution, where Δ gives the range over which transmission occurs and α is the exponent (larger values imply a quicker decay with distance).
- N_l is the population in city l.
- P is the UK population.

System:
- Initially susceptible individuals, with city populations taken from census data.

M3.4: Farm-based model using a distance kernel (IBM)

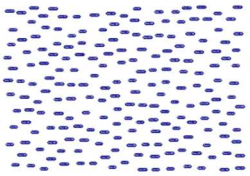


Purpose:
- Introduce farm-based models.
- In these models farms are represented as geographical points (either as compartments with x,y positions, or using longitudes and latitudes).
- Each farm contains a single "individual" that is used to denote the disease status of that farm.
- A spatial kernel allows for spread of disease between farms.

Model:
- A single individual-based species "Farm" is created.
- This contains classifications "DS", which stands for "disease status", and "Location".
- "DS" contains three compartments: susceptible farm S, infected farm I and recovered farm R.
- "Location" contains 200 farm positions.
- A transmission rate β determines the rate at which susceptible farms become infected due to surrounding infected farms.
- A recovery rate γ determines how quickly farms recover.

System:
- A single individual is added to each farm location. All farms are initially susceptible except for one infected.

M3.5: Farm-based model with density dependency (IBM)



Purpose:
- Introduce density dependency in farm-based models.
- In these models farms are represented as geographical points (either as compartments with x,y positions, or using longitudes and latitudes).
- Each farm contains a single "individual" that is used to denote the disease status of that farm.
- A spatial kernel allows for spread of disease between farms.
- A density-dependent susceptibility is employed. This is implemented using the 'RDEN' vector, which given the relative density of farms (this has an average of one and is calculated using KDE with a radius of 30km).

Model:
- A single individual-based species "Farm" is created.
- This contains classifications "DS", which stands for "disease status", and "Location".
- "DS" contains three compartments: susceptible farm S, infected farm I and recovered farm R.
- "Location" contains 800 farm positions (a sub-sample of poultry farms across the UK).
- A transmission rate β determines the rate at which susceptible farms become infected due to surrounding infected farms.
- A rate η determines infections from outside the system.
- A recovery rate γ determines how quickly farms recover.

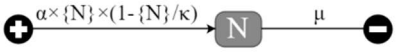
System:
- A single individual is added to each farm location. All farms are initially susceptible.

M4 Ecological models

Compartmental models are often applied to epidemiological applications, but they can equally be used in other settings. Here we look at some examples from ecology.

M4.1: Logistic growth population model (IBM)

Purpose:
- Introduce a simple model to represent the dynamics of population in a wildlife setting.
- This uses a logistic growth model.
- Initially the rate at which individuals are born is proportional to the number of individuals currently present.
- Finite resources limit the size of the population through a carrying capacity κ.

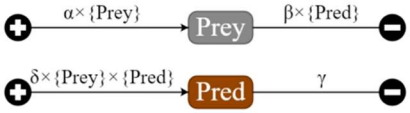


Model:
- A single population-based species "Animal" is created.
- This contains a classification "Population".
- Population contains a compartment N, which represents the number of individuals.
- A birth rate α determines how fast individuals are born.
- A mortality rate μ determines how fast individuals die.

System:
- The initial state starts with just a single individual.

M4.2: A predator-prey model (PBM)

Purpose:
- Introduce a simple model to demonstrate predator-prey dynamics.
- This uses the well-known Lotka-Volterra equations.
- The birth rate of prey is proportional to their population.
- Prey die at a rate proportional to the number of predators.
- The birth rate of predators is proportional to their population as well as the population of prey (because predators rely on prey for food).
- Predators die at a constant rate.

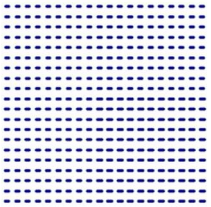


Model:
- A single population-based species "Animal" is created.
- This contains a classification "Population".
- Population contains two compartments: "Prey" and "Pred".
- A birth rate α determines how fast prey are born.
- A mortality rate β determines how fast prey die.
- A birth rate φ determines how fast predators are born.
- A mortality rate γ determines how fast predators die.

System:
- This initial state starts with 100 predators and 100 prey.

M4.3 A spatial diffusion model (IBM)

Purpose:
- Introduce a model in which a landscape is divided into a grid of cells.
- Individuals move across this landscape by randomly hopping from one grid cell to a neighbouring grid cell.



Model:
- A single individual-based species 'Animal' is created.
- This contains a classification "Location".
- "Location" contains a 20x20 grid of cells represented by compartments.
- A hopping rate ω determines the rate at which individuals move between cells.

System:
- 100 individuals initially in cell G10-10.

M4.4: A species presence/absence distribution model (IBM)

Purpose:
- Introduce a simple presence/absence model for an invasive plant species.
- A map of the UK is split into a grid of 25km squares.
- A covariate is used that affects the probability the species will invade a given square (*e.g.* it could represent some environmental variable such as altitude or mean temperature).

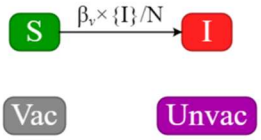
Model:
- A single individual-based species "Plant" is created.
- This contains classifications "Presence" and "Location".
- "Presence" contains two compartments: "No" and "Yes".
- "Location" contains compartments representing the grid cells (559 in total).
- Each grid cell contains a single individual that represents the cell's status.
- A transmission rate β determines the rate at which cells start growing the plant.
- A fixed effect $\langle g \rangle$ is used to add a covariate affecting plant spread.
- A spatial power-law transmission kernel is used to model the geographical spread of the plant.

System:
- Initially, a single grid cell is set to contain the invasive plant species.

M5 Disease transmission experiments

Disease transmission experiments can be used to identify factors affecting the passing of infection from one individual to another. These can either be fixed categorical effects, such as vaccination status, covariates, such as weight, or quantitative genetic traits.

M5.1: Single contact group, investigating susceptibility (IBM)



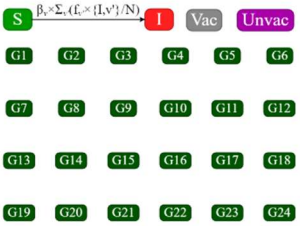
Purpose:
- Introduce the concept of a disease transmission experiment.
- These are used to identify factors which affects how readily disease transmits between individuals.
- This example considers the effect of vaccination on susceptibility to disease.

Model:
- A single individual-based species "Animal" is created.
- This contains classifications "DS", which stands for "disease status", and "Vaccination".
- "DS" contains two compartments: susceptible S and infected I.
- "Vaccination" contains two compartments: "Vac" and "Unvac", which represent vaccinated and unvaccinated individuals.
- A transmission rate β_v determines the rate at which individuals with vaccination status v become infected.

System:
- 100 initially susceptible individuals and one infected.

M5.2: Multiple contact groups, investigating infectivity (IBM)

Purpose:
- Introduce disease transmission experiment for quantifying differences in infectivity.
- For this it is necessary to combine information from multiple contact groups.
- In this example we consider the effect of vaccination on susceptibility and infectivity.

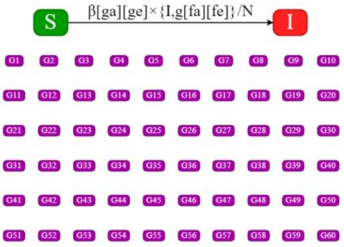


Model:
- A single individual-based species 'Animal' is created.
- This contains classifications “DS”, which stands for “disease status”, “Vaccination” and “Group”.
- “DS” contains two compartments: susceptible S and infected I.
- “Vaccination” contains two compartments: “Vac” and “Unvac”, which represent vaccinated and unvaccinated individuals.
- “Group” contains 24 contact groups.
- A transmission rate β_v determines the susceptibility of individuals with vaccination status v.
- A factor f_v determines the relative infectiousness of individuals with vaccination status v.

System:
- Each group consists of 10 individuals, half of which are initially infected.
- The vaccination status of the infected and susceptibility individuals is permuted to create blocks of 4 that are replicated 6 times (to make 24 groups in total).

M5.3: Quantitative genetics model for susceptibility/infectivity (IBM)

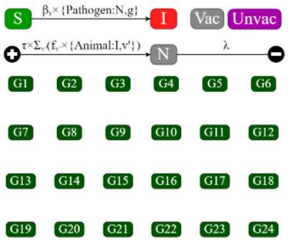
Purpose:
- Introduce disease transmission experiment for estimating quantitative genetic contributions to susceptibility and infectivity.
- A population of full-sib families is randomly allocated into contact groups.



Model:
- A single individual-based species “Animal” is created.
- This contains classifications “DS” and “Group”.
- “DS” contains two compartments: susceptible S and infected I.
- “Group” contains 60 contact groups.
- A transmission rate β determines how fast individuals become infected.
- Individual effects [ga] (for genetic contribution) and [ge] (for environmental contribution) characterise individual variation in susceptibility.
- Individual effects [fa] (for genetic contribution) and [fe] (for environmental contribution) characterise individual variation in infectivity.
- $\mathbf{\Omega}^{gen}$ gives the genetic variances and correlations.
- $\mathbf{\Omega}^{env}$ gives the environmental variances and correlations.

System:
- Each contact group consists of 10 individuals, two of which have been randomly infected.
- The population is made up of 60 sires who each have 10 progeny. The 600 progeny overall are randomly allocated across the contact groups.

M5.4: Environmental pathogen accumulation model (IBM/PBM)



Purpose:
- Rather than modelling direct transmission of disease between individuals, here we assume that pathogen accumulates in the environment.
- A disease transmission experiment with multiple contact groups is assumed.

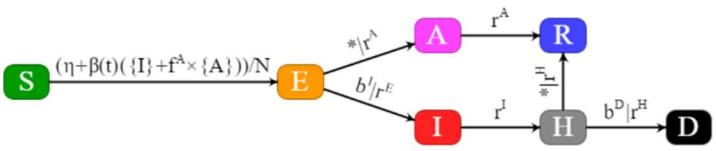
Model:
- An individual-based species "Animal" is created.
- This contains classifications "DS", which stands for "disease status", "Vaccination" and "Group".
- "DS" contains two compartments: susceptible S and infected I.
- "Vaccination" contains two compartments: "Vac" and "Unvac", which represent vaccinated and unvaccinated individuals.
- "Group" contains 24 contact groups.
- A population-based species "Pathogen" models the environmental accumulation.
- This contains classifications "Population", which stores the amount of pathogen, and "Group", which is cloned from 'Animal'.
- A transmission rate β_v determines the susceptibility of individuals with vaccination status v.
- A factor f_v determines the relative infectiousness of individuals with vaccination status v.
- τ determines the pathogen shedding rate from infected individuals.
- λ determines the decay of the pathogen in the environment.

System:
- Each group consists of 10 individuals, half of which are initially infected.
- The vaccination status of the infected and susceptibility individuals is permuted to create blocks of 4 that are replicated 6 times (to make 24 groups in total).

M6 Covid-19 models

These examples look at a variety of different models aimed at understanding real-world epidemic outbreaks.

M6.1: Simple (PBM)



Purpose:
- Show a possible model of Covid-19.
- Here additional compartments are used to not only more accurately model the disease, but such that the model aligns with publicly available data.

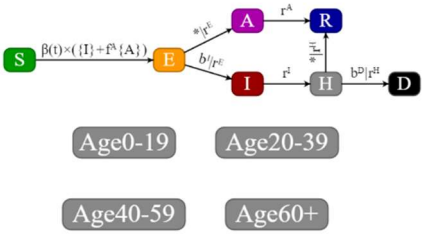
Model:
- A single population-based species "People" is created.
- This contains a classification "DS", which stands for "disease status".
- DS contains seven compartments: susceptible S, exposed E, infected I, recovered R, asymptomatic A, hospitalised H and dead D.
- An external force of infection is controlled by parameter η (in reality this could be a spline estimated from raw case data and flight information).
- The transmission rate β(t) is time varying to capture the influence of lockdown measures and vaccination.
- Parameter f^A determines the relative infectiousness of asymptomatic A individuals (comparted with infected I individuals).

System:
- The UK population, which is initially susceptible.

M6.2: Age-structured model (PBM)

Purpose:
- This adds age stratification into model 6.1.

Model:
- A single population-based species "People" is created.
- This contains classifications "DS", which stands for "disease status", and "Age".
- "DS" contains seven compartments: susceptible S, exposed E, infected I, recovered R, asymptomatic A, hospitalised H and dead D.



- "Age" contains 4 age classifications: 0-19, 20-39, 40-59 and 60+.
- An external force of infection is controlled by parameter η (in reality this could be a spline estimated from raw case data and flight information).
- The transmission rate β(t) is time varying to capture the influence of lockdown measures and vaccination.
- An age mixing $M_{a,a'}$ captures the relative contact rate between different age groups.

System:
- The UK population, which is initially susceptible.

The following illustrate feature of simulation, inference or posterior simulation:

A Simulation features and initial conditions

These examples look at various simulation features and ways for specifying the initial conditions.

A1: Multiple simulations (PBM, M1.1)

Purpose:
- On the "Simulation→Run" page the number of simulations is set to 100 (instead of the usual 1).
- When run, the simulated outputs exhibit a range of system dynamics reflecting inherent stochasticity within the system.
- Shaded regions in the population plots show the envelope containing 95% of simulations.

Model:
- A single population-based species "People" is created.
- This contains a classification "DS", which stands for "disease status".
- DS contains two compartments: susceptible S and infected I.
- A transmission rate β determines the rate at which individuals become infected.

System:
- 100 initially susceptible individuals and one infected.

.

A2: Uncertain initial conditions for PBM — single classification (PBM, M1.1)

Purpose:
- Introduce uncertainty in the initial population for a population-based model.
- This considers the case of a model with just a single classification.
- On the 'Simulation→Initial Conditions' page the initial population is set as a distribution (rather than fixed).
- 100 simulations are performed to see stochastic variation.

Model:
- A single population-based species "People" is created.
- This contains a classification "DS", which stands for "disease status".
- DS contains two compartments: susceptible S and infected I.
- A transmission rate β determines the rate at which individuals become infected.

System:
• The initial population in the S compartment is sampled uniformly in the range between 900 and 1100.
• The initial population in the I compartment is sampled uniformly in the range between 0 and 50.

A3: Uncertain initial conditions for PBM — multiple classifications — focal selected (PBM, M2.6)

Purpose:
• Introduce specification of uncertainty in the initial population population-based model when there is more than one classification.
• The initial population is given on the "Simulation→Initial Conditions" page.
• Here the disease status is a specified "focal" classification.
• Distributions for the initial populations in focal compartments are set (in this case chosen to be uniform distributions).
• For the other classification ("Sex") a Dirichlet distribution is used to sample the *fraction* of individuals in each compartment.
• 100 simulations are performed to see stochastic variation.

Model:
• A single population-based species "People" is created.
• This contains a classification "DS", which stands for "disease status", and a classification "Sex".
• "DS" contains three compartments: susceptible S, infected I and recovered R.
• "Sex" contains the compartments M and F for male and female.
• A transmission rate β_s determines the rate at which individuals with sex s become infected.
• A recovery rate γ_s determines the rate at which individuals recover with sex s.
• N is the total number of individuals (this model assumes frequency dependent transmission).

System:
• This consists of initially S uniformly sampled between 9000 and 11000, I uniformly sampled between 0 and 100 and R uniformly sampled between 0 and 100.
• There is a 30/70 split between males and females, as determined from the Dirichlet distribution.

A4: Uncertain initial conditions for PBM — multiple classifications — total population selected (PBM, M2.6)

Purpose:
• Introduce specification of uncertainty in the initial population population-based model when there is more than one classification.
• The initial population is given on the 'Simulation→Initial Conditions' page.
• In this case a prior is placed on the total population.
• A Dirichlet distribution is placed on all compartmental combinations.
• 100 simulations are performed to see stochastic variation.

Model:
• A single population-based species "People" is created.
• This contains classifications "DS", which stands for "disease status", and "Sex".
• "DS" contains three compartments: susceptible S, infected I and recovered R.
• "Sex" contains the compartments M and F for male and female.
• A transmission rate β_s determines the rate at which individuals with sex s become infected.
• A recovery rate γ_s determines the rate at which individuals recover with sex s.
• N is the total number of individuals (this model assumes frequency dependent transmission).

System:
• The initial population is sampled from a uniform distribution between 9000 and 11000.
• A Dirichlet distribution is used to sample the compartmental composition of the initial state (with most individuals in the S compartment).

A5: Uncertain initial conditions for IBM using individual state (IBM, M1.2)

Purpose:
- Introduce uncertainty in the initial population for an individual-based model.
- On the 'Simulation→Initial Conditions' page the individuals added to system have uncertainty regarding their initial state.
- 100 simulations are performed to see stochastic variation.

Model:
- A single population-based species "People" is created.
- This contains a classification "DS", which stands for "disease status".
- DS contains two compartments: susceptible S and infected I.
- A transmission rate β determines the rate at which individuals become infected.

System:
- The initial population consists of 1000 individuals added at the start time $t=0$.
- The initial compartmental specification for these individuals is given by "S:0.98|I:0.02".
- This means that there is a 98% probability of being in the S compartment and a 2% probability of being in the I compartment.

A6: Uncertain initial conditions for IBM using population distribution (IBM, M1.2)

Purpose:
- Introduce uncertainty in the initial population for an individual-based model.
- On the 'Simulation→Initial Conditions' page a population distribution is specified as an initial condition.
- Compared to example A5, this allows for greater flexibility.
- 100 simulations are performed to see stochastic variation.

Model:
- A single population-based species "People" is created.
- This contains a classification "DS", which stands for "disease status".
- DS contains two compartments: susceptible S and infected I.
- A transmission rate β determines the rate at which individuals become infected.

System:
- A population distribution fixes a total population of 1000 individuals.
- A Dirichlet distribution sets the susceptible and infected populations.
- $\alpha^S=10$ and $\alpha^I=0.2$ implies that there is a 10/(10+0.2)=98% mean probability of being in the S compartment and a 2% probability of being in the I compartment initially.
- Compared to example A5, the distribution for the number of infected individuals is much broader.
- Increasing α^S and α^I by a factor keeps the percentages the same but reduce the uncertainty in the distributions.
- The initial population consists of 1000 individuals added at the start time $t=0$.
- They have an unspecified initial state (denoted by '.', which is equivalent to an equal probability of being in either state 'S|I').
- Note, if a population distribution is specified, individual added at the start time must have an unspecified initial state.

A7: Add / remove individuals (PBM, M2.6)

Purpose:
- Introduce adding and/or removing populations after the initial conditions are set.
- Relevant for a population-based model.
- The 'Simulation→Initial Conditions' page show how the additions and removals are defined.

Model:
- A single population-based species "People" is created.
- This contains classifications "DS", which stands for "disease status", and "Sex".
- "DS" contains three compartments: susceptible S, infected I and recovered R.
- "Sex" contains the compartments M and F for male and female.
- A transmission rate β_s determines the rate at which individuals with sex s become infected.
- A recovery rate γ_s determines the rate at which individuals recover with sex s.

• N is the total number of individuals (this model assumes frequency dependent transmission).

System:
• 10,000 initially susceptible individuals and 2 infected.
• A 50/50 split between males and females.

A8: Add / move / remove individuals (IBM)

Purpose:
• To show how individuals can be added, moved and removed from the system.
• Relevant for an individual-based model.
• 100 susceptible individuals are added at time t=0.
• Here we have an SIR model within two fish tanks: T1 and T2.
• 10 infected individuals are added to tank T1.
• At time t=100, 10 specified individuals are moved from to tank T2. This then initiates an epidemic in tank T2.

Model:
• A single population-based species "People" is created.
• This contains classifications "DS", which stands for "disease status", and "Tank".
• "DS" contains two compartments: susceptible S and infected I.
• "Tank" contains two fish tank compartments: T1 and T2.
• A transmission rate β determines the rate at which individuals become infected.

System:
• 100 initially susceptible individuals and one infected.

B) Population-level data types

These examples look at different types of population-level data.

B1: Time series population observations (PBM, M1.1)

Purpose:
• Inference is done using time-series population data.

Data:
• On the 'Inference→Data' page, data is added that gives a time-series estimate for the number of individuals in the I compartment.

Model:
• A single population-based species "People" is created.
• This contains a classification "DS", which stands for "disease status".
• DS contains two compartments: susceptible S and infected I.
• A transmission rate β determines the rate at which individuals become infected.

System:
• 100 initially susceptible individuals and one infected.

B2: Time series population-level transition observations (PBM, M1.1)

Purpose:
• Inference is done using time-series population-level transition data.

Data:
• On the 'Inference→Data' page, population-level transition data is added.
• This gives time-series estimates for the number infection transitions at periodic intervals with timestep Δt=7.

Model:
• A single population-based species "People" is created.
• This contains a classification "DS", which stands for "disease status".
• DS contains two compartments: susceptible S and infected I.
• A transmission rate β determines the rate at which individuals become infected.

System:
• 100 initially susceptible individuals and one infected.

B3: Stratified time series population observations (PBM, M2.6)

Purpose:
• Introduce inference on stratified population data.

Data:
• On the "Inference→Data" page, data is added which gives time series estimates for the populations in each the compartmental combinations.

Model:
- A single population-based species “People” is created.
- This contains classifications “DS”, which stands for “disease status”, and “Sex”.
- “DS” contains three compartments: susceptible S, infected I and recovered R.
- “Sex” contains the compartments M and F for males and females.
- A transmission rate β_s determines the rate at which individuals with sex s become infected.
- A recovery rate γ_s determines the rate at which individuals recover with sex s.
- N is the total number of individuals (this model assumes frequency dependent transmission).

System:
- 10,000 initially susceptible individuals and 2 infected.
- A 50/50 split between males and females.

B4: Stratified population observations from multiple compartments (PBM, M2.6)

Purpose:
- Introduce inference on multiple population time series that target different combinations of compartments.

Data:
- Stratified population data is added on the 'Inference→Data' page.
- One time series provides stratified observations on the infected population.
- A second time series provides overall observations for individuals in *either* compartments I *or* R. For example, this could represent a serological test.

Model:
- A single population-based species “People” is created.
- This contains classifications “DS”, which stands for “disease status”, and “Sex”.
- “DS” contains three compartments: susceptible S, infected I and recovered R.
- “Sex” contains the compartments M and F for males and females.
- A transmission rate β_s determines the rate at which individuals with sex s become infected.
- A recovery rate γ_s determines the rate at which individuals recover with sex s.
- N is the total number of individuals (this model assumes frequency dependent transmission).

System:
- 10,000 initially susceptible individuals and 2 infected.
- A 50/50 split between males and females.

B5: Combined population-based data sources in a Covid-19 model (PBM, M6.1)

Purpose:
- Show how multiple population-based data can be used to inform a Covid-19 model.
- Using this data, it is possible to estimate branching probabilities and the time variation in the transmission rate $\beta(t)$.
- However, this sort of data will not estimate transition rates once infected, so informative priors have been used for these.

Data:
- Population-level transition data is available for E→I (cases), I→H (hospitalisations) and H→D (hospital deaths).
- Population data for the combined population in E, I and A (survey PCR data).
- Population data for the combined population in E, I, A and R (seroprevalence data).

Model:
- A single population-based species “People” is created.
- This contains a classification “DS”, which stands for “disease status”.
- DS contains seven compartments: susceptible S, exposed E, infected I, recovered R, asymptomatic A, hospitalised H and dead D.
- An external force of infection is controlled by parameter η (in reality this could be time-varying and estimated from raw case data and flight information).
- The time-varying spline $\beta(t)$ captures transmission between individuals.

System:
- The UK population, which is initially susceptible.

B6: Time series population observations (IBM, M1.2)

Purpose: • Introduce population-based data incorporated into an IBMs.
Data: • On the 'Inference→Data' page population data is added.
• This gives time-series estimates for the number of individuals in the infected I compartment.
Model: • A single population-based species "People" is created.
• This contains a classification "DS", which stands for "disease status".
• DS contains two compartments: susceptible S and infected I.
• A transmission rate β determines the rate at which individuals become infected.
System: • 100 initially susceptible individuals and one infected.

C) Individual-level data types

These examples look at different types of individual-level data.

C1: Known transition events — infection and recovery (IBM, M1.3)

Purpose: • Shows an inference example where all transition events are known.
Data: • Known infection and recovery times for all individuals in an SIR model.
Model: • A single population-based species "People" is created.
• This contains a classification "DS", which stands for "disease status".
• DS contains three compartments: susceptible S, infected I and recovered R.
• A transmission rate β determines the rate at which individuals become infected.
• A recovery rate γ determines the rate at which individuals recover.
System: • 100 initially susceptible individuals and one infected.

C2: Incomplete transition events - recovery only (IBM, M1.3)

Purpose: • Show an inference example where only recovery transition events are known.
• We see that it is still possible to infer β and γ, albeit with more uncertainty.
Data: • Recovery times for individuals in an SIR model.
Model: • A single population-based species "People" is created.
• This contains a classification "DS", which stands for "disease status".
• DS contains three compartments: susceptible S, infected I and recovered R.
• A transmission rate β determines the rate at which individuals become infected.
• A recovery rate γ determines the rate at which individuals recover.
System: • 100 initially susceptible individuals and one infected.

C3: Compartmental observations (IBM, M1.3)

Purpose: • Illustrate compartmental observations.
Data: • The disease status of every individual is measured with a 20 time unit interval.
Model: • A single population-based species "People" is created.
• This contains a classification "DS", which stands for "disease status".
• DS contains three compartments: susceptible S, infected I and recovered R.
• A transmission rate β determines the rate at which individuals become infected.
• A recovery rate γ determines the rate at which individuals recover.
System: • 100 initially susceptible individuals and one infected.

C4: Disease diagnostic test results (IBM, M1.3)

Purpose: • Show an inference example where disease diagnostic tests are used.

Data:
- Tests are made on individuals every 10 time units.
- The test has a sensitivity of 0.6 and a specificity of 0.99.

Model:
- A single population-based species "People" is created.
- This contains a classification "DS", which stands for "disease status".
- DS contains three compartments: susceptible S, infected I and recovered R.
- A transmission rate β determines the rate at which individuals become infected.
- A recovery rate γ determines the rate at which individuals recover.

System:
- 100 initially susceptible individuals and one infected.

C5: A partially observed transition (IBM, M1.3)

Purpose:
- Show an inference example where some transitions are partially observed.

Data:
- Here the data collected assumes that all recovery events are observed, but infections are only observed with a f=50% probability.

Model:
- A single population-based species "People" is created.
- This contains a classification "DS", which stands for "disease status".
- DS contains three compartments: susceptible S, infected I and recovered R.
- A transmission rate β determines the rate at which individuals become infected.
- A recovery rate γ determines the rate at which individuals recover.

System:
- 100 initially susceptible individuals and one infected.

C6: A transition observed over a time window (IBM, M1.3)

Purpose:
- Shows an inference example where a transition is only observed over a specified time window.

Data:
- All recovery events are observed.
- Infection events are only observed between times t=5 and t=15.

Model:
- A single population-based species "People" is created.
- This contains a classification "DS", which stands for "disease status".
- DS contains three compartments: susceptible S, infected I and recovered R.
- A transmission rate β determines the rate at which individuals become infected.
- A recovery rate γ determines the rate at which individuals recover.

System:
- 100 initially susceptible individuals and one infected.

C7: A transition observed in a demographic category (IBM, M2.8)

Purpose:
- This example looks at when some transitions are only observed within a subpopulation.

Data:
- All recovery transitions are observed.
- Infections are only observed in males.

Model:
- A single population-based species "People" is created.
- This contains classifications "DS", which stands for "disease status", and "Sex".
- "DS" contains three compartments: susceptible S, infected I and recovered R.
- "Sex" contains the compartments M and F for male and female.
- A transmission rate β_s determines the rate at which individuals with sex s become infected.
- A recovery rate γ_s determines the rate at which individuals recover with sex s.
- N is the total number of individuals (this model assumes frequency dependent transmission).

System:
- 100 initially susceptible individuals and 2 infected.
- A 50/50 split between males and females.

C8: Uncertain compartmental observations (IBM, M1.3)

Purpose:
- This example allows for uncertainty in compartmental observation.
- Here fixed probabilities are used, but the observation model can also include parameters that can be estimated during inference.

Data:
- Periodic uncertain compartmental observations are made every 20 time units.

Model:
- A single population-based species "People" is created.
- This contains a classification "DS", which stands for "disease status".
- DS contains three compartments: susceptible S, infected I and recovered R.
- A transmission rate β determines the rate at which individuals become infected.
- A recovery rate γ determines the rate at which individuals recover.

System:
- 100 initially susceptible individuals and one infected.

D) Time variation

These examples look at different ways in which time variation can be added into the model.

D1: Time variation in transmission rate (PBM, M1.1)

Purpose:
- Introduce a model with time variation in disease transmission.
- The transmission rate $\beta(t)$ is taken to be a piecewise linear spline with knots taken at 6 time points.

Data:
- Time series estimates of the infected population.

Model:
- A single population-based species "People" is created.
- This contains a classification "DS", which stands for "disease status".
- DS contains two compartments: susceptible S and infected I.
- A transmission rate $\beta(t)$ determines the rate at which individuals become infected.

System:
- 100 initially susceptible individuals and one infected.

D2: Time variation in transmission rate using a trigonometric function (PBM, M1.1)

Purpose:
- In this example the transmission rate is taken to have a time-varying profile using a cosine curve.

Data:
- Time series population estimates.

Model:
- A single population-based species "People" is created.
- This contains a classification "DS", which stands for "disease status".
- DS contains two compartments: susceptible S and infected I.
- The transmission rate is taken to be $\beta(1+\Delta\times\cos(\alpha\times t))$.

System:
- 1,000 initially susceptible individuals and one infected.

D3: Time variation in transmission rate through a covariate (PBM, M1.1)

Purpose:
- Introduce time-varying covariate into transmission rate.
- The covariate is exponentiated (to ensure it is positive) and modulates the transmission rate through a strength parameter Δ.

Data:
- Time series population data is used to inform inference.

Model:
- A single population-based species "People" is created.
- This contains a classification "DS", which stands for "disease status".
- DS contains two compartments: susceptible S and infected I.
- The transmission rate is given by $\beta\times\exp(\Delta\times f(t))$, where $f(t)$ is a time-varying covariate.

System:
- 1,000 initially susceptible individuals and one infected.

D4: Time variation in population-level transition observation probability (PBM, M1.1)

Purpose:
- Incorporate time variation in observing population transitions.
- The probability is taken to have the functional form $a+b\times\cos(0.06\times t)$.
- To simulate the data $a=0.6$ and $b=0.3$ were used. These values can be inferred from the data.

Data:
- Partially observed population-level transition data.
- Time series population data.

Model:
- A single population-based species "People" is created.
- This contains a classification "DS", which stands for "disease status".
- DS contains two compartments: susceptible S and infected I.
- A transmission rate β determines the rate at which individuals become infected.

System:
- 1,000 initially susceptible individuals and one infected.

D5: Time variation in individual transition observation probability (PBM, M1.2)

Purpose:
- Incorporate time variation in the probability of observing individual transitions.
- Like D4 but with an IBM and individual-level data.
- The probability of observing transitions is given the functional form $a+b\times\cos(0.03\times t)$.
- The values $a=0.5$ and $b=0.3$ were used to simulate the data.
- Parameters a and b can be inferred from inference.

Data:
- Partially observed transitions.
- Periodic compartmental observations.

Model:
- A single population-based species "People" is created.
- This contains a classification "DS", which stands for "disease status".
- DS contains two compartments: susceptible S and infected I.
- A transmission rate β determines the rate at which individuals become infected.

System:
- 100 initially susceptible individuals and one infected.

D6: Time variation in population observation probability (PBM)

Purpose:
- Incorporate time variation in observing the number of individuals in a population.
- Here population observations only detect a fraction f of the true population.
- This, for example, could represent the number of individuals captured during a series of capture campaigns.
- Because capturing individuals is often seasonally dependent, we apply a trigonometric function $f=a\times\exp(b\times\cos(0.06\times t))$.

Data:
- Time series population estimates.
- Simulated data used $a=0.1$ and $b=0.3$.

Model:
- A single population-based species "Animal" is created.
- This contains a classification "Area" with a single compartment "P".
- "P" represents the animal population.

System:
- 1000 individuals.

D7: Time variation in branching probability (PBM, M2.1)

Purpose:
- Illustrate time variation in branching probability for a population-based model.

Data:
- Time series population measurements.

Model:
- A single population-based species "People" is created.
- This contains a classification "DS", which stands for "disease status".
- DS contains four compartments: susceptible S, infected I, recovered R and dead D.
- A transmission rate β determines the rate at which individuals become infected.

- A spline $b(t)$ is used to represent the branching probability of dying.
- Parameter γ specifies the recovery/death rate.

System:
- 1000 initially susceptible individuals and two infected.

D8: Time-varying covariate affecting branching probability (IBM, M2.2)

Purpose:
- Introduce time variation in a covariate that affects branching probability.
- This is implemented using 'branching factors'.

Data:
- Known event times are used to inform inference.

Model:
- A single individual-based species "People" is created.
- This contains a classification "DS", which stands for "disease status".
- DS contains four compartments: susceptible S, infected I, recovered R and dead D.
- A transmission rate β determines the rate at which individuals become infected.
- The *relative* probability of dying is given by branching factor $a \times \exp(b \times \cos(0.06 \times t))$.
- The *relative* probability of recovery is given by branching factor 1.
- Mean m^R and coefficient of variation c^R specify the log-normally distributed period to recovery.
- Mean m^D and coefficient of variation c^D specify the gamma distributed period to death.

System:
- 100 initially susceptible individuals and two infected.

E) Individual-based variation

These examples look at how individual-based variation can be incorporated into the model.

E1: Individual fixed effect applied to a transition (IBM, M1.2)

Purpose:
- Introduce individual fixed effect applied to the infection transition.

Data:
- Infection event times.

Model:
- A single individual-based species "People" is created.
- This contains a classification "DS", which stands for "disease status".
- DS contains two compartments: susceptible S and infected I.
- A transmission rate β determines the rate at which individuals become infected.
- A fixed effect $\langle g \rangle$ varies the susceptibilities of individuals based on a covariate.
- A parameter μ^g governs the strength of the fixed effect.

System:
- 100 initially susceptible individuals and one infected.

E2: Individual fixed effect applied to a population (IBM)

Purpose:
- Introduce adding a fixed effect into a population.
- In the SI model the infected population $\{I\}$ determines how quickly other individual become infected.
- Individual variation in infectivity can incorporated through an individual fixed effect.
- The term $\{I\langle f \rangle\}$ sums over infected individuals, but rather than each individual having a unit contribution it is modified by a fixed effect factor $\langle f \rangle$.

Data:
- Infection times.

Model:
- A single individual-based species "Animal" is created.
- This contains classifications "DS", which stands for "disease status", and "Group".
- "DS" contains two compartments: susceptible S and infected I.
- "Group" contains 24 contact groups.
- A transmission rate β determines the rate at which individuals become infected.
- Fixed effect parameter μ^f determines the strength of the fixed effect.

System:
- Each of the 24 groups consist of 10 individuals, two of which are infected.

E3: Individual effect applied to a transition (IBM, M1.2)

Purpose:
- Introduce a random individual effect affecting the susceptibility of individuals.

Data:
- Infection event times.

Model:
- A single individual-based species "People" is created.
- This contains a classification "DS", which stands for "disease status".
- DS contains two compartments: susceptible S and infected I.
- A transmission rate β determines the rate at which individuals become infected.
- An individual effect [g] allows for susceptibility variation across individuals.
- The strength of this variation is determined through variance Ω^{gg}.

System:
- 100 initially susceptible individuals and one infected.

E4: Correlated individual effect applied to a transition (IBM, M1.2)

Purpose:
- Introduce a random individual effect affecting the susceptibility of individuals.
- Unlike E3, the individual effect between individuals is correlated through an $\mathbf{A}$ relationship matrix.

Data:
- Infection event times.

Model:
- A single individual-based species "People" is created.
- This contains a classification "DS", which stands for "disease status".
- DS contains two compartments: susceptible S and infected I.
- A transmission rate β determines the rate at which individuals become infected.
- An individual effect [g] allows for susceptibility variation across individuals.
- The strength of this variation is determined through variance Ω^{gg}.
- An $\mathbf{A}$ relationship matrix determines correlations in individual effect.

System:
- This consists of 60 groups each with 10 half-sib individuals.
- 598 initially susceptible individuals and two infected.

E5: Correlated individual effect applied to a population (IBM, M5.3)

Purpose:
- Introduce an individual effect applied to vary the infectiousness of individuals.
- In the SI model the infected population $\{I\}$ determines how quickly other individual become infected.
- The term $\{I[f]\}$ sums over infected individuals, but rather than each individual having a unit contribution it is modified by an individual effect factor [f] (this factor has a population average of 1 but allows for individual variation).
- The individual effect is correlated through a relationship matrix.

Data:
- Known infection times.

Model:
- A single individual-based species "Animal" is created.
- This contains classifications "DS" and "Group".
- "DS" contains two compartments: susceptible S and infected I.
- "Group" contains 60 contact groups.
- A transmission rate β determines how fast individuals become infected.
- [f] give individual effects for infectivity.
- Ω^{ff} give the genetic variance in infectivity.

System:
- Each contact group consists of 10 individuals, two of which have been randomly infected.
- 60 sires who each have 10 progeny. These 600 progeny are randomly allocated across the contact groups.

E6: Fixed effect applied to a branching probability (IBM, M2.3)

Purpose:
- Introduce individual-variation in branching probability through a fixed effect.
- Branching factors are used.

Data:
- Periodic compartmental observations.

Model:
- A single individual-based species "People" is created.
- This contains a classification "DS", which stands for "disease status".
- DS contains four compartments: susceptible S, infected I, recovered R and dead D.
- A transmission rate β determines the rate at which individuals become infected.
- The *relative* probability of dying is given by b⟨g⟩.
- The *relative* probability of recovering is given by 1.
- Mean m^R and coefficient of variation c^R specify the log-normally distributed period to recovery.
- Mean m^D and coefficient of variation c^D specify the gamma distributed period to death.
- The strength of the fixed effect is determined through parameter μ^g.

System:
- 100 initially susceptible individuals of which two are infected.

E7: Individual effect applied to a branching probability (IBM, M2.3)

Purpose:
- Introduce individual-variation in branching probability through an individual effect.
- Branching factors are used.

Data:
- Known event times.

Model:
- A single individual-based species "People" is created.
- This contains a classification "DS", which stands for "disease status".
- DS contains four compartments: susceptible S, infected I, recovered R and dead D.
- A transmission rate β determines the rate at which individuals become infected.
- The *relative* probability of dying is given by b[g].
- The *relative* probability of recovering is given by 1.
- Mean m^R and coefficient of variation c^R specify the log-normally distributed period to recovery.
- Mean m^D and coefficient of variation c^D specify the gamma distributed period to death.
- The variance of the individual effect is determined by Ω^{gg}.
- Individual effects are correlated through a relationship matrix $\mathbf{A}$.

System:
- 60 sires each have 10 progeny. These 600 progeny are used in the system (of which 2 are initial infected).

E8: Correlated individual effect applied to a transition with pedigree (IBM, M1.2)

Purpose:
- Introduce a random individual effect affecting the susceptibility of individuals.
- The individual effects between individuals are correlated through an $\mathbf{A}$ relationship matrix. Unlike E4, this relationship is defined through a pedigree.

Data:
- Infection event times.
- True values for individual effect [g] obtained from simulation (this allows for the prediction accuracies to be estimated).
- Three groups are defined: 'sire', 'dam' and 'prod'. These denote the sires and dams in the base population and the progeny.

Model:
- A single individual-based species "People" is created.
- This contains a classification "DS", which stands for "disease status".
- DS contains two compartments: susceptible S and infected I.
- A transmission rate β determines the rate at which individuals become infected.
- An individual effect [g] allows for susceptibility variation across individuals.
- The strength of this variation is determined through variance Ω^{gg}.
- An $\mathbf{A}$ relationship matrix determines correlations in individual effect.

System:
- 60 sires and dams in the base population, with each couple having 10 progeny.

• Only the 600 progeny are incorporated into the compartmental model, with 598 initially susceptible individuals and two infected (note, individual effects for sires and dam not in the system are also estimated).

F) Parameter definitions

These examples look at how parameters can be reparameterised, set to distributions or derived from other quantities.

F1: Reparameterisation (IBM, M3.4)

Purpose:
• Introduce a farm-based model with reparameterisation.
• A matrix of interactions $M_{l,l}$, represents transmission of infection between farms.
• It is reparameterised as a power-law distribution kernel $1/\left(1+\left(\frac{D_{l,l'}}{\Delta}\right)^{\alpha}\right)$.
• $D_{l,l}$, is a reserved parameter in BICI which gives the distance between locations.
• A maximum interaction distance is set, which is used to significantly speed up the algorithm (because distant interactions of negligible effect are removed).

Data:
• Event times known.

Model:
• A single population-based species "Farm" is created.
• This contains classifications "DS", which stands for "disease status", and "Location".
• "DS" contains three compartments: susceptible farm S, infected farm I and recovered farm R.
• "Location" contains farm positions.
• A transmission rate β determines the rate at which infected farms transmit disease to surrounding farms.
• A recovery rate γ determines how quickly farms recover.

System:
• A single individual is added to each farm location. All farms are initially susceptible except for one infected.

F2: Parameter distribution (IBM, M5.2)

Purpose:
• In some instances, it is useful for a parameter to be sampled from a distribution.
• An example is a random effect in a mixed model.
• This is illustrated here with a disease transmission experiment.

Data:
• Known transition times.

Model:
• A single individual-based species "Animal" is created.
• This contains classifications "DS", which stands for "disease status", "Vaccination" and "Group".
• "DS" contains two compartments: susceptible S and infected I.
• "Vaccination" contains two compartments: "Vac" and "Unvac", which represent vaccinated and unvaccinated individuals.
• "Group" contains 24 contact groups.
• A transmission rate β_v determines the susceptibility of individuals with vaccination status v become infected.
• A factor f_v determines the relative infectiousness of individuals with vaccination status v
• The rate of transmission in different groups is modulated by a group-specific factor G_g.
• The elements in parameter vector G_g are assumed to be drawn from a normal distribution with mean zero and standard deviation σ^G.

System:
• Each group consists of 10 individuals, half of which are infected.
• The vaccination status of the infected and susceptibility individuals is permuted to create blocks of 4 which are replicated 6 times (to make 24 groups in total).

F3: Derived quantities (IBM, M1.3)

Purpose:
- Derived quantities are functionally dependent on model parameters or populations, but are not themselves within the model.
- In this example the basic reproduction number R_0 is calculated.

Data:
- Known event times.

Model:
- A single population-based species "People" is created.
- This contains a classification "DS", which stands for "disease status".
- DS contains three compartments: susceptible S, infected I and recovered R.
- A transmission rate β determines the rate at which individuals become infected.
- A recovery rate γ determines the rate at which individuals recover.
- This example shows different derived expressions:
  1. The basic reproduction number R_0 is derived by user-defined equation $N\beta/\gamma$, where N is the number of individuals.
  2. The time-varying reproduction number is calculated using the function RN(...), where the infected states are placed within the brackets (see Appendix D for details).
  3. The effective reproduction number is calculated using the function RNE(...).
  4. The generation time is calculated using the function GT(...).
  5. The estimated number of infections NI is found by integrating over the force of infection multiplied by the number of susceptible.
  6. The estimated number of recoveries NR in the first 20 time units is derived.

System:
- 100 initially susceptible individuals and one infected.

F4: Factor (PBM, M2.6)

Purpose:
- This is the same model as M2.6, but here the difference is disease transmission between sexes is implemented as a factor.

Data:
- Initial population in each compartment.
- Time-series measurements on infected population, stratified by sex.

Model:
- A single population-based species "People" is created.
- This contains classifications "DS", which stands for "disease status", and "Sex".
- "DS" contains three compartments: susceptible S, infected I and recovered R.
- "Sex" contains the compartments M and F for male and female.
- A transmission rate β determines the rate at which individuals become infected.
- A factor f_s allows for differences in susceptibility between the sexes.
- A recovery rate γ_s determines the rate at which individuals recover with sex s.
- N is the total number of individuals (this model assumes frequency dependent transmission).

System:
- 10,000 initially susceptible individuals and 2 infected.
- A 50/50 split between males and females.

F5: Spline reparameterisation (IBM, M3.4)

Purpose:
- Provide an example of when splines are reparameterised.
- In the special case in which the spline is square, values can depend on populations at the knot time points.
- This is used here to incorporate the effect of disease transmission through marketplaces in a farm-based model.
- Farms are represented as geographical points (either as compartments with *x* / *y* positions, or using longitudes and latitudes).

	• Each farm contains a single "individual" that is used to denote the disease status of that farm.
Data:	• Event times known.
Model:	• A single population-based species “Farm” is created. • This contains classifications “DS”, which stands for “disease status”, and “Location”. • “DS” contains three compartments: susceptible farm S, infected farm I and recovered farm R. • “Location” contains farm positions. • A transmission rate β determines the rate at which susceptible farms become infected due to infection coming from markets. • Square spline $f_l(t)$ represent the disease risk of acquiring infection from one of the other farms which have been at market. It has knot times at daily intervals. • A recovery rate γ determines how quickly farms recover.
System:	• A single individual is added to each farm location. All farms are initially susceptible except for one infected.

G) Incorporating pathogen genetics

G1: Matrix of genetic differences (IBM, M1.2)

Purpose:	• Genetic observations on the pathogen allow for the transmission tree to be estimated.
Data:	• Known event times. • Genetic observations are taken every 20 time units on all individuals in the system. • A matrix of genetic differences between the observations is used to inform inference.
Model:	• A single individual-based species “People” is created. • This contains a classification “DS”, which stands for “disease status”. • DS contains two compartments: susceptible S and infected I. • A transmission rate β determines the rate at which individuals become infected.
System:	• 100 initially susceptible individuals and one infected.

9) License and warranty

BICI is free software under the terms of the GNU General Public License version 3 www.gnu.org/licenses/gpl-3.0.en.html. This allows users to redistribute and/or modify BICI. The program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY.

10) Citing BICI

We kindly request that those who do use BICI analysis in their publications cite this tool:

Pooley CM, Doeschl-Wilson AB, Marion G., *BICI – A flexible tool for simulation, inference and posterior simulation from compartmental models*. To be submitted (2020)

Acknowledgments

BICI makes use of the software NW.js (from the website nwjs.io/) to create the interface. This software is excellent and highly recommended.

References

[1] C. M. Pooley, S. C. Bishop, A. B. Doeschl-Wilson, and G. Marion, "Estimating genetic and non-genetic effects for host susceptibility, infectivity and recoverability using temporal epidemic data," *bioRxiv,* p. 618363, 2019.

[2] E. Parzen, "On estimation of a probability density function and mode," *The annals of mathematical statistics,* vol. 33, no. 3, pp. 1065-1076, 1962.

[3] R. E. Kass and A. E. Raftery, "Bayes factors," *Journal of the American Statistical Association,* vol. 90, no. 430, pp. 773-795, 1995.

[4] H. Jeffreys, *The theory of probability*. OUP Oxford, 1998.

[5] A. Gelman and D. B. Rubin, "Inference from iterative simulation using multiple sequences," *Statistical science,* vol. 7, no. 4, pp. 457-472, 1992.

[6] B. Carpenter *et al.*, "Stan: A probabilistic programming language," *Journal of statistical software,* vol. 76, no. 1, 2017.

Appendix A: BICI-script commands ordered by section

BICI-script contains 57 possible commands that can define models, load data, specify analyses, and store outputs. This appendix orders these commands following the menu tree structure. A brief description is given, along with a reference to the section in the manual where more details can be found. Appendix B orders commands alphabetically, and provides further implementation details. Appendix C provides illustrative examples of BIC commands.

Model	Command	Description
→ *Compartments*	**species** (see §2.1.1)	Adds a species to the model.
	classification / class (see §2.1.2)	Adds a classification to a species.
	compartment / comp (see §2.1.3)	Adds a compartment to a classification.
	compartment-all / comp-all	Adds a list of compartments to a classification, as specified in a data file.
	transition / trans (see §2.1.4)	Adds a transition to a classification.
	transition-all / trans-all	Adds a list of transitions to a classification, as specified in a data file.
	set (see §7.1)	In the BICI-script this sets the current species and/or classification being focused on.
→ Parameters	**parameter / param** (see §2.2)	Sets details of a parameter.
	fixed-effect (see §2.2.3)	Provides information about the covariate vector $\mathbf{X}$ used for a fixed effect.
	ind-effect (see §2.2.2)	Provides information about a single (or multiple if correlated) individual effect.
	derived / der (see §2.2.8)	Sets a derived quantity in the model that is functionally related to other model parameters.
→ *Annotations*	**box** (see §2.1.8)	Adds a box around a specified set of compartments in the workspace.
	label (see §2.1.8)	Adds a label to the model.
	map (see §2.1.8)	Adds a map to the workspace (only in 'latlng' coordinates).
→ *Miscellaneous*	**description /desc** (see §2.3)	Adds a description about the model.
	view	Sets how the classification is viewed in the BICI interface.
	data-dir (see §7.1)	Sets the directory where data is stored. This is either relative to the .bici file or using a path.

Simulation	Command	Description
→ *Initial Conditions*	**init-pop-sim** (see §3.1.1)	Sets the initial population.
	add-ind-sim (see §3.1.4)	Adds individuals to the system at specified times and into specified compartments.
	remove-ind-sim (see §3.1.5)	Removes individuals from the system at specified times.
	move-ind-sim (see §3.1.6)	Moves individuals to specified compartments at specified times.
	add-pop-sim (see §3.1.2)	Adds populations of individuals into specified compartments at specified times.
	remove-pop-sim (see §3.1.3)	Removes populations of individuals at specified times from specified compartments
→ *Parameters*	**parameter / param** (see §3.2)	Sets parameter values.
→ *Run*	**simulation / sim** (see §3.3)	Specifies simulation details.
→ *Output*	**param-sim** (see §7.4.1)	Command created by the BICI core code to store parameter samples generated by simulation.
	state-sim (see §7.4.3)	Command created by the BICI core code to store state samples generated by simulation.
	warning-sim (see Appendix G)	Stores a run-time simulation warning.

Inference	Command	Description
→ *Data Init. Conds.*	**init-pop-inf** (see §3.1.1)	Sets the initial population.
	add-ind-inf (see §3.1.4)	Adds individuals to the system at specified times and into specified compartments.
	remove-ind-inf (see §3.1.5)	Removes individuals from the system at specified times.
	move-ind-inf (see §3.1.6)	Moves individuals to specified compartments at specified times.
	add-pop-inf (see §3.1.2)	Adds populations of individuals into specified compartments at specified times.
	remove-pop-inf (see §3.1.3)	Removes populations of individuals at specified times from specified compartments.
→ *Data Individual*	**comp-data** (see §4.1.2.1)	Provides information about the compartment individuals are in at specified time points.
	trans-data (see §4.1.2.2)	Provides information about the timings of individual transitions.
	test-data (see §4.1.2.3)	Adds disease diagnostic test data.
	genetic-data	Incorporates genetic sequence data.

	(see §4.1.2.4)	
→ *Data Population*	**pop-data** (see §4.1.3.1)	Adds population data.
	pop-trans-data (see §4.1.3.2)	Adds aggregated population-level transition data (*e.g.* number of cases per week).
→ *Data Additional*	**ind-effect-data** (see §4.1.4.1)	Adds individual effect data.
	ind-group-data (see §4.1.4.2)	Defines grouping of individuals.
→ *Prior*	**parameter / param** (see §4.2)	Sets parameter prior.
→ *Run*	**inference / inf** (see §4.3)	Specifies inference details.
→ *Output*	**param-inf** (see §7.4.1)	Command created by the BICI core code to store parameter samples generated by inference.
	param-stats-inf (see §7.4.2)	Command created by the BICI core code to provide parameter statistics from inference
	state-inf (see §7.4.3)	Command created by the BICI core code to store state samples generated by inference.
	diagnostics-inf	Outputs diagnostic information about MCMC samplers used in DA-MCMC or PAS-MCMC.
	generation-inf	Stores information about different generations under PAS-MCMC and ABC-SMC algorithms.
	trans-diag-inf	Provides diagnostic information about transitions.
	proposal-inf	Stores information about optimised proposals (used if 'ext' extends inference updates).
	warning-inf (see Appendix G)	Stores a run-time inference warning.

Posterior Simulation	Command	Description
→ *Pop. Modification*	**add-ind-post-sim** (see §3.1.4)	Adds additional individuals (on top of those already defined by the posterior) to the system at specified times and into specified compartments.
	remove-ind-post-sim (see §3.1.5)	Removes additional individuals from the system at specified times.
	move-ind-post-sim (see §3.1.6)	Moves additional individuals to specified compartments at specified times.
	add-pop-post-sim (see §3.1.2)	Adds additional populations of individuals into specified compartments at specified times
	remove-pop-post-sim (see §3.1.3)	Removes additional populations of individuals at specified times from specified compartments.
→ *Parameter Modification*	**param-mult** (see §5.2)	Sets up time varying factor that multiplies a model parameter.

→ *Run*	**posterior-simulation / post-sim** (see §5.3)	Specifies posterior simulation details.
→ *Output*	**param-post-sim** (see §7.4.1)	Command created by the BICI core code to store parameter samples generated by posterior sim.
	state-post-sim (see §7.4.3)	Command created by the BICI core code to store state samples generated by posterior sim.
	warning-post-sim (see Appendix G)	Stores a run-time posterior simulation warning.

Appendix B: BICI-script commands alphabetically ordered

This appendix provides an alphabetical list of commands used in BICI-script.

Note, all commands that contain 'file' also allow for the optional '*cols*' specification – This can be used to set alternative column headings, *e.g.* if the headings "ID" and "t" are expected then cols="Name,Time" would use "Name" in place of "ID" and "Time" in place of "t". This would allow, for example, for one data table to contain all the information about a set of individuals, and for different commands to references different columns in that table.

Command	Description
add-ind-sim / add-ind-inf / add-ind-post-sim (see §3.1.4)	Adds individuals to the system at specified times and into specified compartments ('add-ind-sim' is for simulation and 'add-ind-inf' is for inference). In the case of posterior simulation, 'add-ind-post-sim' adds individuals on top of those already defined in the posterior (and so for a posterior-predictive checks, this command is not required). Note, these commands are used for individual-based models only. *E.g.* add-ind-sim name="Individuals" file="add-ind-data.csv" *name* (optional) – The name of the data source, as used when generating errors or warnings (by default set to "Added individuals"). *file* – The name of the file that contains the data table. This table must contain at least the following columns: "ID" a unique identifier for individuals, "t" the times at which the individuals are added, and columns for each classification to denote into which state the individuals enter. See §3.1.4 for a description of how uncertainty in compartmental state is implemented.
add-pop-sim / add-pop-inf / add-pop-post-sim (see §3.1.2)	Adds populations of individuals into specified compartments at specified times ('add-pop-sim' is for simulation and 'add-pop-inf' is for inference). In the case of posterior simulation, 'add-pop-post-sim' adds populations on top of those already defined in the posterior. *E.g.* add-pop-sim name="Added pop." file="add-pop-data.csv" *name* (optional) – The name of the data source, as used when generating errors or warnings (by default set to "Added populations"). *file* – The name of the file that contains the data table. This table must contain at least the following columns: "t" the times at which the populations are added,

	columns for each of the classifications to denote into which compartment the individuals enter the system, and “Population”, which gives the number of individuals entering.
box (see §2.1.8)	In the workspace this adds a labelled box around a specified set of compartments. *E.g.* box text="This is a bounding box" comps="S,I" *text* – Provides the text at the top of the bounding box. *comps* – Lists the compartments (comma separated). *text-size* (optional) – Positive integer that sets the size of the text (10 by default). *color* (optional) – Sets the colour of the box (in hexadecimal or RGB format).
classification / class (see §2.1.2)	Adds a classification to the species. *E.g.* classification name="DS" index="d" *name* – The name of the classification (note, this must be unique and not the same as any species or compartment). *index* – This is a single lower-case letter used as a mathematical index that represents compartments in the classification. Each classification must have a unique index (note, ‘t’ cannot be used as that is reserved to represent time or ‘z’, which is reserved for indices on covariance matrices). *coord* (optional) – Sets the coordinate system (set to "cartesian" by default). • "cartesian" – Positions of compartments made on an x,y grid. • "latlng" – Positions of compartments made using latitude and longitude. *map* (optional) – Loads a default world map (only if using "lnglat" coordinates) • "load" – Load the map. **Cloning** – A classification can be cloned from another species. In this case it adopts exactly the same set of compartments (although its transitions may be defined differently). For cloning the syntax is given by: *E.g.* classification name="DS" clone="People” *name* – The name of the classification. *clone* – The name of the species from which the classification is cloned.
compartment / comp (see §2.1.3)	This adds a compartment to a classification. *E.g.* compartment name="S" x=0 y=1 color="#ff0000" *name* – The name of the compartment. *color* – The colour, as specified in hexadecimal or RGB format. *x,y* (optional) – The position in the model (in "cartesian" coordinates). If these values are not set, BICI will automatically generate the position (this, however, can sometimes lead to confusing compartmental diagrams). *lat,lng* (optional) – Latitude and longitude (in "latlng" coordinates). Must be set if the longitude and latitude are being used for the classification.

	boundary (optional) – Instead of a location, this sets a file with boundary data in GeoJSON format (in 'latlng' coordinates). *fix* (optional) **–** Fixes the position of the compartment (*i.e.,* so it cannot be dragged by the mouse in the workspace). This can take the values "true" or "false" (by default set to "false"). *branch-prob* (optional) – If all transitions leaving a compartment are exponentially distributed, setting this option to "true" ensures that branching probabilities are used. This can take the values "true" or "false" (by default set to "false"). *infected* (optional) – When 'trans-tree' is set for the species, this option specifies that the compartment is associated with the "infected" state. This can take the values "true" or "false" (by default set to "false").
compartment-all / comp-all	Adds a list of compartments to a classification, as specified in a data file (if there are numerous compartments this is more concise than using many 'compartment' commands). Note, this option cannot be used with boundary data. *E.g.* comp-all color="#00ff00" file="comp-all-data.csv" *file* – The name of the file that contains the data table. This table must contain at least the following columns: "name", which provides the name of the compartment, and "x" and "y" (in "cartesian" coordinates) or "lat" and "lng" (in "latlng" coordinates). Other columns in this table can have headings given by any of the properties below (such that they can be individually specified for each compartment). *color* (optional) – The colour as specified in hexadecimal or RGB format. *fix* (optional) **–** Fixes the position of the compartment (*i.e.* so it cannot be dragged by the mouse in the workspace). This can take the values "true" or "false" (by default set to "false"). *branch-prob* (optional) – If all transitions leaving a compartment are exponentially distributed, setting this option to "true" ensures that branching probabilities are used. This can take the values "true" or "false" (by default set to "false"). *infected* (optional) – When 'trans-tree' is set for the species, this option specifies that the compartment is associated with the "infected" state. This can take the values "true" or "false" (by default set to "false").
comp-data (see §4.1.2.1)	Provides information about the compartment individuals are in at specified time points. *E.g.* comp-data name="DS observations" class="DS" file="comp-data.csv" *name* – The name of the data source, as used when generating errors or warnings. *class* – The name of the classification on which compartmental information is taken. *file* – The name of the file that contains the data table. This table must contain at least the following columns: "ID" a unique identifier for individuals, "t" the

	measurement times, and the name of the classification on which the state measurement is made. The last column contains entries such as 'S' to denote the individual is susceptible at a given time point. See §4.1.2.1 for details on how uncertainty is implemented.
data-dir (see §7.1)	Sets the directory where data is stored. This is either relative to the .bici file or using a path. *E.g.* data-dir folder="data-files" *folder* – The directory path.
derived / der (see §2.2.8)	Sets a derived quantity in the model that is functionally related to other model parameters through an equation. Such a quantity usually has some physical meaning (*e.g.* the reproduction number R_0 in an epidemic model, which is related to the transmission and recovery rates). *E.g.* derived name="R0" eqn="β/γ" *name* – The name of the new parameter. *eqn* – The equation that relates it to other parameters in the model.
description / desc (see §2.3)	This adds a description about the model. *E.g.* description text="file.txt " *text* – A text file provides a description of the model and analysis. A markdown format is used, *i.e.* this allows for the following concise formatting: "# Title", "## Subtitle", "- Bullet point", "*italic text*" and "**bold text**". Parameters can be added by enclosing in dollar symbols, *e.g.* "a" or "b^super_sub".
diagnostics-inf	Outputs MCMC diagnostic information for the DA-MCMC and PAS-MCMC algorithms. *E.g.* diagnostics-inf chain=1 file="diagnostics0.txt" *chain* (optional) – The MCMC chain number. *file* – The name of the file providing diagnostic information.
fixed-effect (see §2.2.3)	Provides information about a covariate vector $\mathbf{X}$ used for a fixed effect. *E.g.* fixed-effect name="m" X="X-data.csv" *name* – The name of the fixed effect. *X* – The name of the file that contains the data table giving information on the covariate vector. This table must contain at least the following columns: "ID" a unique name for each individual and "Value" giving the value of the covariate.
generation-inf	This command is created by the BICI core code and stores information about different generations when using the PAS-MCMC or ABC-SMC algorithms. *E.g.* generation-inf file="generation-data.txt" *file* – Store information.

genetic-data (see §4.1.2.4)	Incorporates genetic sequence data into the analysis. *E.g.* genetic-data name="Genetic data" type="snp" mut-rate="m" seq-var="v" root="SNP" file="genetic-data.csv" *name* – The name of the data source, as used when generating errors or warnings. *type* – Provides the type of genetic data. There are two possibilities: • "matrix" – This makes use of a matrix of base-pair differences for a set of pathogen genetic observations. • "snp" – This makes use of raw SNP data for a set of pathogen genetic observations. *root* (optional) – Gives the root name for the SNP in the file. Must be specified if 'type' is set to "snp". *file* – The name of the file that contains the data table. This table must contain at least the following columns: "ID" a unique name for each individual, "t" giving the time of the observation. The following columns depend on the type of observation: • "matrix" – A column "Obs" provides a unique name for each genetic observation. Elements down this column must also be columns in the table that represent a square matrix giving base-pair differences between the observations. • "snp" – The data table has a series of columns giving sequence data which have 'root' at the beginning of their column name. Elements of these columns are expected to contain two characters denoting the version of the SNP, *e.g.* 'AA', 'AB', 'BA', or 'BB'.
ind-effect (see §2.2.2)	Provides information about a single or multiple (if correlated) individual effects. *E.g.* ind-effect name="gen" ie="f,g" A="A-matrix-data.csv" *name* – A short descriptive name (e.g. "gen" for genetic effect or "env for environmental effect"). *ie* – The individual effects are listed (comma separated). *A* (optional) – When individual effects are correlated across individuals (*e.g.* genetically) it is necessary to set up an $\mathbf{A}$ relationship matrix (that could be pedigree-based or represent a genomic relationship matrix). This is loaded by means of a file that has the individuals in the population in the column headers and a square matrix represented in the table. $\mathbf{A}$ can also include IDs of individuals who do not appear in the compartmental model itself, and individual effect estimates can be made for these (this also applies to other methods for loading $\mathbf{A}$ below. *pedigree* (optional) – An alternative to 'A' in which the parents of each individual are specified. This requires a data table containing at least the following columns: "ID" a unique identifier for individuals, and "sire" and "dam" which give the names of the male and female parents (".” is used if a parent is not in the data-set).

	Ainv (optional) – This loads up the inverse of the relationship matrix by means of a file that has the individuals in the population in the column headers and a square matrix represented in the table. *A-sparse* (optional) – An alternative to 'A' and used for a sparse representation of the matrix. This table must have three columns: "x" and "y" giving the x and y coordinates in the matrix and then "Value". The matrix is assumed to be symmetric, so only values in the upper triangular part of the matrix need to be specified. The order of individuals is provided by 'ind-list' (note, zero indexing is used, such that a value of zero in columns "x" or "y" corresponds to the first individual on the list). Any unspecified elements are assumed to be zero. *ind-list* (optional) – Loads a table with heading "Individual", which gives the order of individuals used for 'A-sparse'.
ind-effect-data (see §4.1.4.1)	Adds individual effect data (note, such information is usually unknown and typically only available when the data is generated through simulation). *E.g.* ind-effect-data name="True ind. eff." ie="g" file="ind-effect-data.csv" *name* – The name of the data source, used for when generating errors or warnings. *ie* – The name of the individual effect *file* – The name of the file that contains the data table. This table must contain at least the following columns: "ID" a unique identified for individuals and "Value", which gives values for the individual effect.
ind-group-data (see §4.1.4.2)	Defines a grouping of individuals. Note, this is not used in simulation or inference, but rather allows for visualisation of specified groupings of individuals (*e.g.* sires, progeny) in the interface, as well as estimates of prediction accuracy for individual effects. *E.g.* ind-group-data name="Progeny" file="ind-group-data.csv" *name* – The name of the data source, as used when generating errors or warnings. *file* – The name of the file that contains the data table. This table must contain an "ID" column, a unique identifier for individuals.
inference / inf (see §4.3)	Specifies inference details. *E.g.* inference start=0 end=100 timestep=0.1 update=1000 algorithm="DA-MCMC" *start* – The start time for inference. *end* – The end time for inference. *timestep* – Sets a finite time-step used to approximate the governing equations. A smaller value provides more accurate results but takes computationally longer. Typically setting this to around 20% of the shortest expected transition period gives fast, accurate results. *algorithm* (optional) – The algorithm used to perform inference (by default set to "DA-MCMC"). It can take the following possibilities:

- "DA-MCMC" – Data augmentation MCMC provides the most accurate inference (although it can be slow for many individuals).
- "PAS-MCMC" – Particle annealed sampling (PAS) generates a power posterior approximation that get successively closer to the true posterior over a series of generations. After annealing, PAS-MCMC gathers parameter and state samples in the same way as DA-MCMC. PAS-MCMC cannot be run on the local machine.
- "ABC" – Approximate Bayesian Computation. Estimates the posterior by simulating from the model and comparing the output with the data (population-based data only). Only those samples which are sufficiently close (as measured by an error function) are used for the posterior approximation.
- "ABC-SMC" – Approximate Bayesian Computation using Sequential Monte Carlo. A version of ABC that improves posterior estimates over a series of generations (population-based data only).
- "PMCMC" – Particle MCMC targets the true posterior and makes use of multiple "particles" (stochastic simulations from the model) run in parallel.
- "MFA" – Stands for mean field approximation. Here an emulator is used to approximate the posterior surface. This can be much faster than other approaches, provided the number of model parameters is not too large.

update (optional, for DA-MCMC or PAS-MCMC) – Sets the number of updates MCMC is iterated for (by default set to 5000). A higher value takes longer to run but may improve the results (because output samples become less correlated).

nchain (optional, for DA-MCMC) – The number of MCMC chains to run (by default set to 3).

chain-per-core (optional, for DA-MCMC) – The number of MCMC chains run per core (by default set to 1). The number of chains must be a multiple of this value.

npart (optional, for PAS-MCMC) – Sets the number of particles used (by default set to 3).
part-per-core (optional, for PAS-MCMC) – The number of particles run per core (by default set to 1). The number of particles must be a multiple of this value.

gen-percent (optional, for PAS-MCMC) – Sets the number of MCMC updates used per generation in the annealing phase as a percentage of the number using in sampling phase, as defined by 'update' (by default set to 1, hence 100 generation would yield approximately equal CPU time spent on annealing and sampling phases).

param-output (optional, for DA-MCMC or PAS-MCMC) – Sets the number of parameter samples generated in the output (by default set to 1000). Note, this is distributed across all chains or particles[27] an includes the burn-in period[28]. A larger value yields smoother parameter distribution plots, but consumes more memory and CPU processing when visualised.

state-output (optional, for DA-MCMC or PAS-MCMC) – Sets the number of state samples generated in the output (by default set to 200). A larger value yields

[27] This is done as equally as possible, *e.g.* if there are 3 chains then 333 parameter samples will be taken from each.
[28] Note, this does not include the annealing phase under PAS-MCMC. Information about annealing comes from the 'generation-inf' command.

smoother population, transition and individual plots, but consumes more memory and CPU processing when visualised.

sample (optional, for ABC) – *Sets* the number of samples generated (by default set to 1000).

acc-frac (optional, for ABC) – S*ets* the fraction of simulated samples accepted to be posterior samples (by default set to 0.1). Making smaller improves the posterior estimate but takes longer to run.

sample (optional, for ABC-SMC) – S*ets* the number of samples generated (by default set to 1000).

acc-frac (optional, for ABC-SMC) – S*ets* the fraction of simulated samples accepted to be posterior samples (by default set to 0.5).

gen (optional, for ABC-SMC) – S*ets* the number of generations (by default set to 5).

kernel-size (optional, for ABC-SMC) – S*ets* the size of the proposal kernel (by default set to 0.5). This size scales the current MVN approximation to the posterior.

burnin-percent (optional, for DA-MCMC or PAS-MCMC) – The percentage of the chain in which burn-in is performed (by default set to 20). Must be in the range 1-90.

anneal (optional, for DA-MCMC) – Type of annealing performed during the burn-in phase (by default "none"). It can take the following possibilities:

- "none" – The inverse temperature is fixed to one during burn-in.
- "scan" – The inverse temperature is scanned from zero to one during burn-in. The ‘rate’ property must be set.
- "log-auto" – An automatic logarithm model is applied to the inverse temperature.
- "power-auto" – An automatic power-law model is applied to the inverse temperature.
- "power" – A power law model is applied. The ‘power’ property must be set.

rate (optional, for DA-MCMC) – When ‘anneal’ is set to "*scan*", this sets how fast burn-in takes place (under this option the overall time for burn-in is *a priori* unknown). By default, set to "0.01".

power (optional, for DA-MCMC) – When ‘anneal’ is set to "power", this sets the power which controls how the inverse temperature is changed. By default, set to "4".

ind-max (optional) – Sets the maximum number of individuals allowed (by default set to 20000). If exceeded, an error message is generated.

param-output-max (optional) – Sets the maximum size of tensors which can be output (by default set to 1000). This avoids very large output files.

seed (optional) – An integer between 0 and 10,000 that sets the computational seed for the pseudorandom number generator (by default set to a fixed value). Changing this value alters the stochastic realisation of the output.

diagnostics (optional) – Determines if MCMC diagnostics are output (these are used in DA-MCMC and PAS-MCMC to determines how well proposals are

	working). Can take the values "off" or "on" (by default set to "off"). If "on" the 'diagnostics-inf' command is generate when inference is run. *sync* (optional) – Set if proposal synchronisation is used. Can take the values "off" or "on" (by default set to "on"). MCMC chains can either be run entirely separately, or information can be used across chains to improve the adaptation of proposals (in which case each chain uses the same synchronised set of proposals). Note, synchronisation will only be actually implemented if it is possible, *e.g.* it is not possible when BICI is running locally on more than one core (due to the fact that local threads can't communicate).
init-pop-sim / init-pop-inf (see §3.1.1)	Sets the initial population (init-pop-sim is for simulation and init-pop-inf is for inference). *E.g.* init-pop-sim type="fixed" focal="DS" file="init-pop-data.csv" *name* (optional) – The name of the data source, as used when generating errors or warnings (by default set to "Initial population"). *file* – The file used to load the data table (see below for details on required columns in this table). *focal* (optional) – Set to a classification name. If this is set, it means that a focal classification is used to specify the population/distribution (see below). *type* (optional) – Determines how the initial population is defined (by default set to "fixed"): • "fixed" – The initial population is exactly defined. **If 'focal' is set** – The file must contain two columns with titles "Compartment" and "Population". The "Compartment" column must go through all compartments in the focal classification and all but one for all other classifications (the remaining is automatically calculated such that percentages/fractions add up to 100%). In the case of the focal classification, the "Population" column gives the number of individuals in each compartment. Other classifications are represented by percentages. **If 'focal' not set** – The file must contain columns for each classification and a final column "Population". This last column gives the number of initial individuals. • "dist" – The initial population is sampled from a distribution. **If 'focal' is set** – The file must contain two columns with titles "Compartment" and "Distribution". The "Compartment" column must go through all compartments in the model. In the case of the focal classification, the "Distribution" column gives the distribution in population number sampled from, *e.g.* "uniform(10,20)". Other classifications are represented by α values used to defined Dirichlet priors. **If 'focal' not set** – The file must contain columns for each classification and a final column "Alpha". This column defines a Dirichlet prior across all possible compartment combinations. For this option a prior for the overall population number must be set using the 'prior' property. *prior* (optional) – When 'type' is set to "dist" and 'focal' us unset, this defines the prior for the overall population number, *e.g.* "gamma(1000,0.5)" (see §4.2.1 for prior options).

label (see §2.1.8)	Adds a label to the model workspace. *E.g.* label text="This is some text" x=13 y=16 color="#ff2222" text-size=20 *text* – The text to be displayed. *x*,*y* (optional) – The position in the workspace (in "cartesian" coordinates). lat,lng (optional) – Latitude and longitude (in 'latlng' coordinates). *color* (optional) – The colour of the text (black by default). *text-size* (optional) – The size of the text (set to 10 by default).
map (see §2.1.8)	Adds a map to the workspace (only in 'latlng' coordinates). *E.g.* map file="UK-boundary.geojson" *file* – The name of the map file (currently only GeoJSON format using latitude and longitude is accepted).
move-ind-sim / move-ind-inf / move-ind-post-sim (see §3.1.6)	Moves individuals to specified compartments at specified times ('move-ind-sim' is for simulation and 'move-ind-inf' is for inference). In the case of posterior simulation, 'move-ind-post-sim' moves individuals on top of those already defined in the posterior (and so for a posterior-predictive checks, this command is not required). Note, these commands are used for individual-based models only. *E.g.* move-ind-sim name="Movements" class="DS" file="move-ind-data.csv" *name* – The name of the data source, as used when generating errors or warnings. *class* – The name of the classification in which the compartmental movement is made. *file* – The name of the file that contains the data table. This table must contain at least the following columns: "ID" a unique identifier for individuals, "t" the times at which the individuals are moved, and "To", which gives the compartment to which the individuals are moved.
parameter / param (see §2.2 / §3.2 / §4.2)	Sets details of a parameter. *E.g.* param name="β_l(t)" value="value-data.csv" knot-times="start,10,end" smooth="normal(0.2)" *name* – The name of the parameter. If it depends on one or more classification, its name is followed by an underscore '_' followed by a comma separated list of any classification indices on which the parameter depends. If the parameter has a time dependency it ends with '(t)'. *value* (optional) – Sets the value for the parameter under simulation. If the parameter is multivariate (*i.e.* it has a classification or time dependency), the name of a specification file is used. This file has columns with headings given by indices for each of the dependences (with "t" being using in the case of a time dependency) followed by a column with the heading "Value". Any elements not specified in the rows of the table are set to zero by default.

const (optional) – Like ‘value’, but sets the parameter to a constant (*i.e.* its takes the same fixed value for simulation, inference or posterior inference).

reparam (optional) – Sets the reparameterisation for the parameter. Here there are two possibilities.

- Equation – The parameter is reparameterised is terms of an equation.
- Split – Here ‘reparam’ is set to a file. This file has columns with headings given by indices for each of parameter dependences (with “t” being using in the case of a time dependency) followed by a column with the heading “Value”. Any elements not specified in the rows of the table are set to zero by default.

knot-times (optional) – When the parameter is represented by a spline (*i.e.* it has a ‘(t)’) this sets the times of the spline knots, comma separated. The key words ‘start’ and ‘end’ are used to denote the start and end times of the simulation/inference.

smooth (optional) – When the parameter is represented by a spline (*i.e.* it has a ‘(t)’) this sets a smoothing condition on the spline. It can take the following values (where the standard deviation SD determines the strength of the smoothing):

- "normal(SD)" – Used for parameters that can become negative or positive.
- "log-normal(SD) " – Used for parameters that are strictly positive.

spline-type (optional) – When the parameter is represented by a spline (*i.e.* it has a ‘(t)’) this sets the type of spline (linear by default). Possibilities are:

- "linear" – Piecewise linear. Here the values of the spline are specified at the knot times. All intermediate points are linearly extrapolated.
- "square" – These splines are flat between the knot times. The value is specified using the knot time at the start of each interval.
- "cubic +ve" – This fits a cubic spline to the log of the parameter value (hence this spline is guaranteed to be positive).
- "cubic" – This fits a cubic spline to the parameter values at the knot times (this spline is not guaranteed to be positive).

prior (optional) – Sets the prior for the parameter. The prior for most variables can take any of the following options:

- "inverse(min,max)" – An inverse distribution truncated to lie between a specified minimum and maximum, *e.g.* "inverse(0.01,1)". Note, this only applies to positive quantities. It is a commonly used distribution because it represents the Jeffreys prior for means and rates.
- "uniform(min,max)" – A uniform distribution with specified minimum and maximum, *e.g.* "uniform(4,5)".
- "exp(mean) " – An exponential distribution with specified mean (only appropriate if the parameter is strictly positive).
- "normal(mean,sd)" – A normal distribution with a specified mean and standard deviation.
- "gamma(mean,cv)" – A gamma distribution with a specified mean and coefficient of variation (for positive parameters only).
- "log-normal(mean,cv)" – A log-normal distribution with a specified mean and coefficient of variation (for positive parameters only).

- "beta(α,β)" – A beta distribution with specified alpha and beta values (for parameters that go between 0 and 1 only). Note, the distribution mean is given by α/(α+β).
- "bernoulli(mean)" – A Bernoulli distribution with a specified mean (for binary parameters that can take the values 0 or 1). The mean must be between 0 and 1.
- "fix(value)" – Fixes the parameter to a particular value.

The prior when 'factor' is set to "true" can only take one option:

- "mdir(sigma)" – The modified Dirichlet distribution (see Appendix F and §2.2.9)

The prior when the parameter represents a covariance matrix must be:

- "mvn-jeffreys(min,max)" – The uninformative used for multivariate normal covariance matrices. Here the minimum specifies a limit on the determinant of the covariance matrix and the maximum sets a hard limit on the variance (see Appendix F for details).
- "mvn-uniform(min,max)" – The minimum and maximum limits determine the range in the diagonal elements of the covariance matrix (see Appendix F).

prior-split (optional) – In cases in which the parameter has dependencies, it may be useful to set the prior differently for each element. This option is set to a file name giving a table of prior definitions. That file has columns for each of the dependences follow by a column with the heading "Prior". The last column uses the same formatting as for 'prior' above.

dist (optional) – Sets a distribution for the parameter. This is like 'prior' (and has the same set of options), but here the distribution contains new parameters, *e.g.* "normal(m,s)", where 'm' and 's' would go on to have their own priors. So instead of becoming part of the prior the distribution forms part of a so-called 'hierarchical' model. For example, random effects have an assumed normal distribution with a standard deviation that is estimated.

dist-split (optional) – In cases in which the parameter has dependencies, it may be useful to set the distribution differently for each element. This option is set to a file that contains a table of distribution definitions. That file has columns for each of the dependences follow by a column with the heading "Prior". The last column uses the same formatting as for 'dist' above.

sim-sample (optional) – In the case in which a distribution is set (via the "dist" option), setting 'sim-sample' to "true" ensures what when the model is simulated, the parameter is sampled from the specified distribution (rather than specified directly by the user). It can take the values "true" or "false" (the value "true" is used by default).

factor (optional) – Set to "true" if the parameter is set to be a factor (see §2.2.9).

factor-weight (optional) – Sets the weight associated with different factor elements (see §2.2.9).

param-inf

(see §7.4.1)

This command is created by the BICI core code and stores parameter samples generated by inference.

E.g. param-inf chain=1 file="parameter-sample-data.csv"

chain (optional) – Denotes the MCMC chain from which the samples were taken.

	file – The data file which contains parameter samples generated from inference. The first column “State” stores the MCMC iteration on which the sample was taken. Other columns contain parameters as well as log-likelihoods and the prior.
param-mult (see §5.2)	Sets up a spline factor that multiplies a parameter when posterior simulation is performed. param name="β" constant="factor-data.csv" knot-times="start,10,end" *name* – The name of a parameter within the model. *constant* – Sets the constant values for the parameter factor along the spline. This file has columns with headings given by indices for each of the dependences in the parameter, “t” which gives the spline knot times, followed by a column with the heading “Value”. Any elements not specified in the rows of the table are set to zero by default. *knot-times* – This sets the times of the spline knots, comma separated. The key words ‘start’ and ‘end’ are used to denote the start and end times of the inference. *spline-type* (optional) – This sets the sets the type of spline (linear by default). Possibilities are: • "linear" – Piecewise linear. Here the values of the spline are specified at the knot times. All intermediate points are linearly extrapolated. • "square" – These splines are flat between the knot times. The value is specified using the knot time at the start of each interval. • "cubic +ve" – This fits a cubic spline to the log of the parameter value (hence this spline is guaranteed to be positive). • "cubic" – This fits a cubic spline to the parameter values at the knot times (this spline is not guaranteed to be positive).
param-sim / param-inf / param-post-sim (see §7.4.1)	These commands are created by the BICI core code and store parameter samples generated under simulation, inference or posterior simulation. *E.g.* param-inf file="parameter-sample-data.csv" *file* – The data file which contains parameter samples generated from inference. The first column “State” stores the posterior simulation number. Other columns contain parameter information as well as log-likelihoods and the prior.
param-stats-inf (see §7.4.2)	Outputs statistics for each parameter: mean, standard deviation (sd), 95% credible interval (CI min and max), effective sample size (ESS) and Gelman-Rubin statistics (GR) (see §4.4.7 for details). *E.g.* param-stats-inf file="param-stats-inf.csv" *file* – The name of file which contains the table of parameter statistics.
pop-data (see §4.1.3.1)	Adds population data to the analysis. *E.g.* pop-data name="Infected population" filter="DS=I" error="normal:10%" file="pop-data.csv"

	name – The name of the data source, as used when generating errors or warnings. *file* – The name of the file that contains the data table. This table must contain at least the following columns: “t” the times at which the population measurements are made, columns which specify any filters applied to specify the sub-population under observation (see ‘filter’ below), and “Population”, which gives the estimated population size. In the case in which ‘error’ is set to "normal:file", a final column “SD” is used to specify the standard deviation in the observation error. If ‘error’ is set to "neg-binomial:file", a final column “p” is used to specify the probability p in the negative binomial distribution. *filter* (optional) – Filters subpopulation. In the example above, the measured sub-population contains only individuals in the ‘I’ compartment in classification ‘DS’. Multiple compartments are separated using the ‘|’ character, *e.g.* "DS=E|I" would mean that the subpopulation includes those in either the E or I states. The fraction of individual observed can also be set, *e.g.* "DS=E:0.4|I:0.9" means that 40% of E individuals and 90% of I individuals are observed. Note, fractions can be larger than one because observations could represent something proportional to the number of infected individual (*e.g.* environmental pathogen concentration). Equations can be used to represent fractions, *e.g.* "DS=E:p|I:η(t)" where the parameter p and spline η(t) could be estimated during inference. Filtering can be done on more than one classification by comma separating, *e.g.* "DS=I, Sex=M" would mean that only male individual in the infected I state are observed. A filter can be placed into the data file using the ‘file’ keyword, *e.g.* "Location=file". In this case the file would contain a column with heading “Location” containing the filter. *error* (optional) – The observation error captures noise in the measurement process itself. This assumes that the observed value is drawn from a distribution with mean given by the true underlying value. There are three possibilities for this distribution: • **Poisson** – ‘error’ is set by "poisson". This has a variance equal to the mean. • **Normal** – This distribution can be chosen to have a variance less than the Poisson. It can be defined in one of three different ways: 1) As a percentage, *e.g.* "normal:10%" implies the standard deviation (SD) is 10% the mean, 2) As a SD, *e.g.* "normal:2.5" implies a fixed SD of 2.5, or 3) Defined by column “SD” in the file, *e.g.* "normal:file". • **Negative binomial** – This distribution has a variance greater than the Poisson. It can be defined in one of two different ways: 1) With a fixed p value, *e.g.* "neg-binomial:0.5" has p=0.5 or 2) Defined by a column “p” in the file, e.g. "neg-binomial:file".
pop-trans-data (see §4.1.3.2)	Adds aggregated population-level transition data to the analysis (*e.g.* the number of cases per week). *E.g.* pop-trans-data name="Male infections" trans="S->I" filter="M" file="pop-trans-data.csv" *name* – The name of the data source, as used when generating errors or warnings. *trans* – The name of the transition. Multiple transitions can be specified in the same classification, *e.g.* "E->A|E->I" would mean that both ‘E→A’ and ‘E→I’

	transition would both be counted in the observation. The fraction of observed transitions can also be applied, *e.g.* "S->I:0.5" would imply that only 50% of transitions are observed. Such fractions can be parameterised *e.g.* "S->I:p" or "S->I: η(t)". *file* – The name of the file that contains the data table. This table must contain at least the following columns: “Start” and “End” giving the time period over which transitions are counted, any filters applied to specify the sub-population under observation (see ‘filter’), and “Number” which gives the estimated number of transitions. In the case in which ‘error’ is set to "normal:file", a final column “SD” is used to specify the standard deviation in the observation model. If ‘error’ is set to "neg-binomial:file", a final column “p” is used to specify the probability p in the negative binomial distribution. *filter* (optional) – Filters subpopulation. This uses the same specification as ‘filter’ for ‘pop-data’. Note, filtering cannot be performed on the classification in which the transition occurs. *error* (optional) – The observation error captures noise in the measurement process itself. This assumes that the observed value is drawn from a distribution with mean given by the underlying value. There are three possibilities for this distribution: • **Poisson** – ‘error’ is set by "poisson". This has a variance equal to the mean. • **Normal** – This distribution can be chosen to have a variance less than the Poisson. It can be defined in one of three different ways: 1) As a percentage, *e.g.* "normal:10%" implies the standard deviation (SD) is 10% the mean, 2) As a SD, *e.g.* "normal:2.5" implies a fixed SD of 2.5, or 3) Defined by column “SD” in the file, *e.g.* "normal:file". • **Negative binomial** – This distribution has a variance greater than the Poisson. It can be defined in one of two different ways: 1) With a fixed p value, *e.g.* "neg-binomial:0.5" has $p=0.5$ or 2) Defined by a column “p” in the file, e.g. "neg-binomial:file".
posterior-simulation / post-sim (see §5.3)	Specifies posterior simulation details. Note posterior simulation requires that inference has first been performed. *E.g.* post-sim start=0 end=100 number=10 *start* (optional) – The start time for the posterior simulation (by default set to the inference start time). *end* (optional) – The end time for the posterior simulation (by default set to the inference end time). *number* (optional) – The number of posterior simulations to be run (by default set to 200). *ind-max* (optional) – Sets the maximum number of individuals allowed (by default set to 20000). If exceeded, an error message is generated. *param-output-max* (optional) – Sets the maximum size of tensors which can be output (by default set to 1000). This avoids very large output files. *resample* – Determines if distributions are resampled or taken directly from the posterior. Should be set to a comma separate list of parameter names that need resampling. Individual effects can also be resampled. If multiple individuals are

	grouped together, the names of the individual events should be separated by the '-' sign. For example, if the model contains individual effects [a] and [b], resample="a-b" would specify that these are resampled during posterior simulation. *seed* (optional) – An integer between 0 and 10,000 that sets the computational seed for the pseudorandom number generator (by default set to a fixed value). Changing this value alters the stochastic realisation of the output.
proposal-inf	Stores information about optimised proposals (used if 'ext' extends the number of inference updates). *E.g.* proposal-inf file="prop-info.txt" *file* – Store information.
remove-pop-sim / remove-pop-inf / remove-pop-post-sim (see §3.1.3)	Removes populations of individuals from specified compartments at specified times ('remove-pop-sim' is for simulation and 'remove-pop-inf' is for inference). In the case of posterior simulation, 'remove-pop-post-sim' removes populations on top of those already defined in the posterior. *E.g.* remove-pop-sim name="Removed" file="remove-pop-data.csv" *name* (optional) – The name of the data source, as used when generating errors or warnings (by default set to "Removed populations"). *file* – The name of the file that contains the data table. This table must contain at least the following columns: "t" the times at which populations are removed, columns for each of the classifications to denote from which compartment the individuals leave the system, and "Population", which gives the number of individuals leaving.
set (see §7.1)	Sets the current species and/or classification, such that all subsequent commands act on that species/classification. *E.g.* set species="People" *species* – The name of the species. *classification* – The name of the classification.
simulation / sim (see §3.3)	Specifies simulation details. *E.g.* simulation start=0 end=100 timestep=0.5 number=10 *start* – The start time for the simulation. *end* – The end time for the simulation. *timestep* – Sets a finite time-step used to approximate the governing equations. A smaller value provides more accurate results but takes computationally longer. Typically setting this to around 20% of the shortest expected transition period gives fast, accurate results. *number* (optional) – The number of simulations to run (by default set to 1). *ind-max* (optional) – Sets the maximum number of individuals allowed (by default set to 20000). If exceeded, an error message is generated.

	param-output-max (optional) – Sets the maximum size of tensors which can be output (by default set to 1000). This avoids very large output files. *seed* (optional) – An integer between 0 and 10,000 that sets the computational seed for the pseudorandom number generator (by default set to a fixed value). Changing this value alters the stochastic realisation of the output.
species (see §2.1.1)	Adds a species to the model. *E.g.* species name="People" type="individual" *name* – The name of the species. type – The type of model. This can take one of the following possibilities: • "individual" – An individual-based model (*i.e.* each individual in the system has a timeline). This allows for individual-based data as well as non-Markovian transitions. • "population" – A population-based model (here the populations in different compartments are tracked as a function of time). The transitions in the model must be Markovian and data must be population-based. *trans-tree* (optional) – Sets if the transmission tree is turned on (*i.e.*, information about who acquires infection from whom is stored). This can take the values "on" or "off" (set to "off" by default).
state-sim / state-inf / state-post-sim (see §7.4.3)	These commands are created by the BICI core code and store state samples generated by simulation, inference or posterior simulation. *E.g.* state-inf chain=1 file="state-sample-data.txt" *chain* (optional) – Denotes the MCMC chain from which the samples were generated. *file* – The data file which contains state samples generated from inference. See §7.4.3 for format details.
test-data (see §4.1.2.3)	Adds disease diagnostic test data. *E.g.* test-data name="Elisa" Se="1" Sp="1" pos="+" neg="-" comp="I" file="test-data.csv" *name* – The name of the data source, as used when generating errors or warnings. *Se* – The sensitivity of the test. This can either be a number between 0 and 1 or a parameter. *Sp* – The specificity of the test. This can either be a number between 0 and 1 or a parameter. *pos/neg* (optional) – Sets the text value in the data file used to represent a positive/negative test result (1/0 by default). *comp* – Sets the compartment(s), comma separated, that the test is sensitive to. *file* – The name of the file that contains the data table. This table must contain at least the following columns: “ID” a unique identifier for individuals, “t” the times at which the individuals are tested and “Result” which provides the test results.

trans-data (see §4.1.2.2)	Provides information about the timings of observed individual transitions. *E.g.* trans-data name="Male infections" trans="S->I" filter="Sex=M" obsrange="all" file="trans-data.csv" *name* – The name of the data source, as used when generating errors or warnings. *trans* – The name of the transition being observed, which is made up of the initial compartment name, followed by '->' and the final compartment name. Source transitions use the '+' character, *e.g.* "+->S". Sink transition use the '-' character, *e.g.* "R->-". Multiple transitions can be observed, *e.g.* "E->A|E->I" would imply that the observed transition would be one of these two possibilities. A probability can be assigned to observing the transitions, *e.g.* "S->I:0.3" would mean only a 30% probability of observing S→I transitions. This observation probability can be parameterised, *e.g.* "S->I:p" or "S->I:η(t)", where the parameters can be estimated during inference. *filter* (optional) – Filters the population being observed (this can apply to any classification apart from the one in which the transition occurs). In the example above, the measured sub-population contains only individuals in the 'M' compartment in classification 'Sex'. Multiple compartments are separated using the '|' character, *e.g.* "Location=A|C" would mean that the subpopulation includes those in either the A or C locations. The probability of observing the transition can also be given a factor, *e.g.* "Sex=M:0.4" means that the S→I transitions in M individuals are only observed with a 40% probability. Equations can be used to represent probabilities, *e.g.* "Sex=M:η(t)" where the observation probability spline η(t) could be estimated during inference. Filtering can be done on more than one classification by comma separating, *e.g.* "Location=A, Sex=M" would mean that only transitions on male individual at location A are observed. *obsrange* – Determines how the observation period for the transition is specified. It can take the following possibilities: • "all" – This means that the transition is observed over the entire inference period. • "specify" – The start and end of the observation period are specified (see below). *start/end* (optional) – Used to specify the observation period when 'obsrange' is set to "specify". *file* – The name of the file that contains the data table. This table must contain at least the following columns: "ID" a unique identifier for individuals and "t" the times at which individuals undergo transitions.
transition / trans (see §2.1.4)	This adds a transition to a classification. *E.g.* transition name="S->E" value="exp(rate:β*{I})" *name* – Identifies the transition, with the initial compartment name followed by "->" and the final compartment name. Source transitions use "+", *e.g.* "+->S". Sink transitions use "-", *e.g.*, "R->-". *value* – Sets the transition distribution. This can take one of the following possibilities:

	• "exp(rate:[eq])" – An exponential distribution, where "[eq]" is replaced with an equation that determines the rate at which transitions occurs, *e.g.* "exp(rate:r)" or "exp(rate:β×{I})". • "exp(mean:[eq])" – An exponential distribution with an equation that determines the mean. • "gamma(mean:[eq1],cv:[eq2])" – A gamma distribution with equations for mean and coefficient of variation. • "erlang(mean:[eq],shape:[integer])" – An Erlang distribution with equation for mean and fixed positive integer shape. • "log-normal(mean:[eq1],cv:[eq2])" – A log-normal distribution with equations for mean and coefficient of variation. • "weibull(scale:[eq1],shape:[eq2])" – A Weibull distribution with equations for scale and shape. • "period(time:[eq])" – For when the event happens after a certain prescribed period. Note, source transitions can only take the "*exp*" option. Population-based models are only valid for the "*exp*" *or* "*erlang*" options. *bp* (optional) – This sets the equation for the branching probability. This is only used if more than one transition leaves the initial compartment. *x, y* (optional) – In the case of a source or sink transitions, this positions the '+' or '-' symbol in the workspace (in "cartesian" coordinates). *lat, lng* (optional) – In the case of a source or sink transitions, this positions the '+' or '-' symbol in the workspace (in 'latlng' coordinates). *mid-x, mid-y* (optional) – Can be used to add extra points (comma separated) on the line from the initial to final compartments (in "cartesian" coordinates). *mid-lat, mid-lng* (optional) – Can be used to add extra points (comma separated) on the line from the initial to final compartments (in 'latlng' coordinates).
transition-all / trans-all	Adds a list of transitions to a classification, as specified in a data file (if there are numerous transitions, this is more concise than using many separate 'transition' commands). *E.g.* trans-all file="trans-all-data.csv" *file* – The name of the file that contains the transition data table. This table must contain at least the following columns: "name" and "value" (see 'transition' for formatting details). Optionally, additional columns can have headings given by other properties in 'transition', *i.e.* "bp", "x", "y", "lat", "lng", "mid-x", "mid-y", "mid-lat", "mid-lng".
trans-diag-inf	This command is created by the BICI core code and provides diagnostic information about different transitions in the model. *E.g.* trans-diag-inf chain=1 file="trans-diag-data.txt" *chain* (optional) – Denotes the MCMC chain from which the samples were generated. *file* – Store information.

view	This determines the position and scaling used when viewing the model in the BICI interface. This command ensures that when the model is saved and reloaded the workspace view is preserved. *E.g.* view x=4 y=0 scale=2 *x,y* (optional) – The x and y position at the centre of the workspace view (in "cartesian" coordinates). *lat,lng* (optional) – The latitude and longitude at the centre of the workspace view (in 'latlng' coordinates). *scale* – Determines how zoomed in the view is. *grid* (optional) – Plots the grid in the visual interface. Can take the values "off" and "on" (set to "off" by default). *comp-scale* (optional) – Scales the size of compartments as they appear on the screen. This can be used to tackle cases in which compartmental labels overlap each other (can take values between 0.1 and 10, by default set to one).
warning-sim / warning-inf / warning-post-sim / (see Appendix G)	Stores run-time warning messages for simulation, inference or posterior simulation. *E.g.* warning-inf text="This is a run-time warning message!" *text* – The warning message.

Appendix C: BICI-script command examples

To help understand the various possibilities, this appendix provides some illustrative examples for the most commonly used BICI-script commands (please refer to Appendix B for a comprehensive description):

'simulation' command

1) **simulation start=0 end=100 timestep=0.5**
   Simulates from time 0 to time 100 using a timestep of 0.5 (see §3.3.1 for a discussion on how to choose the time-step). Results from simulation are placed into newly created 'param-sim' and 'state-sim' commands, which store parameter values and the system state, respectively (see §7.4).

2) **simulation start=0 end=100 timestep=0.5 number=10**
   As (1), but 10 simulations are performed.

3) **simulation start=0 end=100 timestep=0.5 seed=5**
   As (1), but the random number generator starts with a different random "seed". This means that the stochastic outcome of the simulations will be different.

4) **simulation start=0 end=100 timestep=0.5 indmax=1000000**
   As (1), but here the maximum allowable number of individuals is increased from its default value of 2×10^4 to 10^6 (for individual-based models only).

'inference' command

1) **inference start=0 end=100 timestep=0.5 nchain=3**
   Performs inference on the system from time 0 to time 100 using a timestep of 0.5 (see §3.3.1 for a discussion on how to choose the time-step). This uses the default inference algorithm DA-MCMC (see §4.3.1.1) with three MCMC chains.

2) **inference start=0 end=100 timestep=0.5 nchain=3 update=100000**
   As (1), but perform 10^5 updates instead of the default 5,000 (increasing the number of updates may be necessary to obtain sufficiently representative samples from the posterior).

3) **inference start=0 end=100 timestep=0.5 nchain=3 param-output=100 state-output=50**
   As (1), but the number of parameter samples output is 100 (instead of the default 1000) and the number of state samples output is 50 (instead of the default 200). Reducing the number of output samples can be a way to reduce the size of the output files. On the other hand, increasing these number can make distribution plots smoother.

4) **inference start=0 end=100 timestep=0.5 algorithm="PAS-MCMC" npart=10**
   As (1), but uses an alternative inference algorithm particle anneal sampling MCMC (see §4.3.1.2) with 10 particles.

5) **inference start=0 end=100 timestep=0.5 algorithm="ABC-SMC" gen=5**
   As (1), but uses an alternative inference algorithm Approximate Bayesian Computation using Sequential Monte Carlo (see §4.3.1.4) with 5 generations.

‘post-sim’ command

1) **post-sim start=0 end=100**
   Performs a posterior simulation from time 0 to time 100 (typically coinciding with the time range used under inference). When run, this command randomly draws state samples from the defined ‘state-inf’ command (previously generated under inference), and uses the parameter values and initial conditions to simulate new states. These results are stored in the commands ‘param-post-sim’ and ‘state-post-sim’.

2) **post-sim start=0 end=200**
   As (1), but here the simulation time is extended into the future (note, any splines take a constant value taken from the last time point under inference).

3) **post-sim start=100 end=200**
   As (1), but here the simulation time goes from the end of the inference time to some point in the future. This is a way in which future prediction can be performed.

4) **post-sim start=0 end=100 number=1000**
   As (1), but generates 1000 samples instead of the default 200.

5) **post-sim start=0 end=100 resample="G_g"**
   When parameters are set as distributions (see ‘param’ command (11) below), the ‘resample’ option tells BICI to resample values from the model specified distribution (rather than take them from the posterior). Multiple resampled parameters are comma separated.

‘species’ commands

1) **species name="People" type="population"**
   Adds a species to a model with the name ‘People’, which is defined to be a population-based model (see §2.1.1).

2) **species name="People" type="individual"**
   Adds a species to a model with the name ‘People’, which is defined to be an individual-based model.

3) **species name="People" type="individual" trans-tree="on"**
   As (2), but defines a transmission tree that determines who acquires infection from whom. This requires one or more of the compartmental states in one of the classifications to be set as ‘infected’. Note, this option slows down simulation and inference but is necessary for incorporating pathogen genetic data into analyses.

‘class’ / ‘classification’ command

1) **class name="DS" index="a"**
   This sets up a new classification with name ‘DS’ and mathematical index a.

2) **class name="DS" index="a" coord="latlng"**

As (1), but here coordinates of compartments are defined by longitude and latitude (by default they are Cartesian). Coordinates are important if the distance matrix is used within the model.

3) **class name="DS" clone="People"**
This copies the classification with name 'DS' from another species (such that it contains the same set of compartments, although any transitions may be different).

'comp' / 'compartment' command

1) **comp name="S" x=0 y=1 color="#ff0000"**
Adds a compartment with name 'S' in location x=0, y=1 with colour "#ff0000".

2) **comp name="S" lat=47.442 lng=-4.753 color="#ff0000"**
As (1), but here longitude and latitude are specified.

3) **comp name="S" x=0 y=1 color="#ff0000" branch-prob="true"**
As (1), but this specifies that a branching probability exists for transitions leaving this compartment (only applicable when there are two or more exponentially distributed transitions leaving the compartment).

4) **comp name="S" x=0 y=1 color="#ff0000" infected="true"**
As (1), but this specifies that the compartment is 'infected'. This is used to inform transmission trees.

5) **comp-all color="#00ff00" file="comp-all-data.csv"**
This specifies multiple compartments within a file (useful if there are many). Tags can be specified which apply to all compartments ('color' in this case). The file contains a table with headings 'name', 'x', and 'y'.

'trans' / 'transition' command

1) **trans name="S->I" value="exp(rate:β*{I})"**
Sets up a transition between compartments S and I. The time between an individual entering S and leaving to go to I is sampled from an exponential distribution with rate β times the population in compartment I.

2) **trans name="I->R" value="gamma(mean:m,cv:c)"**
Sets up a transition between compartments I and R. The time between entering I and leaving to R is sampled from a gamma distribution with mean m and coefficient of variation c.

3) **trans name="+->P" value="exp(rate:a)"**
Sets up a source transition that allows for individual to enter the system into compartment P (*e.g.* this could represent births in an ecological model). The value specifies a global rate of individual entering the system.

4) **trans name="I->-" value="log-normal(mean:m,cv:c)"**
Sets up a sink transition that allows for individual to leave the system from compartment I (*e.g.* this could represent deaths).

5) **trans-all file="trans-all-data.csv"**
Sets up multiple transition through a file. This contains a data table with headings 'name' and 'value'.

'param' command

1) **param name="β" value="0.001" prior="uniform(0,0.01)"**
A parameter β has a value 0.001 under simulation and a uniform prior between 0 and 0.01 under inference (§4.2.2 provides details for other prior specifications).

2) **param name="N" constant="10000"**
A parameter N has a constant value 10000 (under simulation or inference).

3) **param name="β_s" value="value-beta.csv" prior="uniform(0,1)"**
A parameter vector β_s has its value set in the file 'value-beta.csv'. This file may look something like this (as displayed in Excel):

	A	B	C	D	E	F	G	H
1	s	Value						
2	M	0.6						
3	F	0.4						

4) **param name="β_s" value="[[**
s,Value
M,0.6
F,0.4
]]" prior="uniform(0,1)"

This is the same as (3), but with the data table stored inline.
For brevity, this format is subsequently written as:
param name="β_s" value="[[s,Value | M,0.6 | F,0.4]]" prior="uniform(0,1)"

5) **param name="K_s,I" value="[[s,,I.Value | M,A,0.1 | M,B,0.2 | F,A,0.15 | F,B,0.22]]" prior="uniform(0,1)"**
A parameter matrix $K_{s,I}$ has a specified value under simulation and a uniform prior under inference.

6) **param name="β(t)" value="[[t,Value | start,1 | 50, 1.5 | end,1.2" prior="uniform(0,2)" knot-times="start,50,end"**
Parameter $\beta(\mathrm{t})$ represents a piece-wise linear spline with knot points at the start, 50 and end times (where "start" and "end" are determined from the time range of the simulation or inference). The vector determined in 'value' sets the values for $\beta(\mathrm{start})$, $\beta(50)$ and $\beta(\mathrm{end})$. Under inference a uniform prior between 0 and 2 is used for these parameters.

7) **param name="β(t)" value="[[t,Value | start,1 | 50, 1.5 | end,1.2" prior="uniform(0,2)" knot-times="start,50,end" smooth="log-normal(0.5)"**
The same as (6), but with a smoothing prior applied under inference (see §2.2.4).

8) **param name="f(t)" constant="[[t,Value | start,0.25 | 50, 0.21 | end,0.22" knot-times="start,50,end"**
Parameter f(t) represents a constant piece-wise linear spline (*e.g.* this could represent some time-varying data).

9) **param name="q^D_s" value="[[s,Value | M,1.2 | F,1.5]]" prior="uniform(0,1)"**
Like (4), but here parameter q_s^D has a superscript (superscripts can be used to differentiate related parameters).

10) **param name="M_l,l'" reparam="1/(1+pow(D_l,l'/Δ|α))"**
Here the matrix $M_{l,l'}$ is reparameterised as $1/\left(1+\left(\frac{D_{l,l'}}{\Delta}\right)^{\alpha}\right)$, which represents is a spatial kernel between different locations separated by distance $D_{l,l'}$. Reparameterisations can involve any number of indices, but they cannot contain populations or splines.

11) **param name="G_g" dist="normal(0,σ^G)"**
Parameter vector G_g is sampled from a normal distribution with zero mean and standard deviation σ^G (distributions take the same options as the priors in §4.2.2, but can contain hyper-parameters).

Appendix D: Derived functions for epidemiological problems

This appendix outlines some functions that are particularly useful when BICI is applied to epidemiological problems. These functions can be set as the right-hand-side of a derived equation. For example,

$$R(t) = RN(E, I) \tag{4}$$

sets the parameter $R(t)$ to be the time-varying reproduction number (see below for a definition).

All the infected (though not necessarily infectious) compartments are specified within the brackets of the function. The example in Eq.(4) is appropriate for an SEIR model, because here E and I are infected compartments and S and R are uninfected. Such a definition is only valid provided at least one transition with a population-dependent rate causes individuals to become infected and none of the infected compartments are terminal[29].

Below are further functions that can be derived:

Function	Description	Requirement
RN(…)	Time-varying reproduction number R(t). This is the expected number of cases from an infected individual in an otherwise susceptible population.	Cannot account for individual or fixed effects. Approximate when non-Markovian.

[29] If an individual indefinitely remains in an infectious state its reproduction number becomes infinite. Consequently, a reproduction number cannot be defined for the SI model.

RNE(…)	Effective reproduction number $R^e(t)$. This characterises the number of cases from an infected individual in the current population.	As RN(…).
RNC(…)	Computational reproduction number $R^c(t)$. This numerically calculates the average number of infections caused by an infected individual.	Must be an individual-based model.
GT(…)	Generation time $G(t)$. This is the mean time between when an individual becomes infected and when it infects other individuals.	Cannot account for individual or fixed effects. Approximate when non-Markovian.
GTE(…)	Effective generation time $G^e(t)$. As GT, but uses the current demographic composition.	As GT(…).
GTC(…)	Computational generation time $G^c(t)$. This numerically calculates mean time between when an individual becomes infected and when it infects others.	Must be an individual-based model.

Definitions

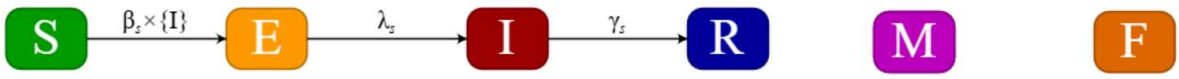
Generation – An epidemic starts with a single infected individual. We refer to this as "generation 1". That individual then goes on to infect other secondary infections which make up generation 2. Those individuals subsequently infect generation 3, and so on and so forth.

Basic reproduction number R_0 – This is often defined to be the expected number of cases directly caused by the individual in generation 1 (*i.e.* in contact with an otherwise completely susceptible population). However, for models that include different demographic classifications, *e.g.* age and/or sex, or complex compartmental structures, careful consideration needs to be given to precisely how R_0 is calculated. In particular, just averaging over the demographic possibilities for the initially infected individual is not enough. One must also consider what happens in subsequent generations in the early phase of an epidemic to get a meaningful estimate for R_0 (because it typically takes several generations for the distribution in demographic groups that cause the bulk of disease transmission to manifest).

Calculations

RN(…) Time-varying reproduction number *R(t)* – This sets the value R_0 would have taken if the initial rates of mixing between individuals in the population are taken to be the same as that at time t. As such, $R(t)$ should be interpreted as a quantity proportional to the rate at which individuals come into contact with each other (with each contact allowing for the possibility of disease transmission). So, for example, when $R(t)$ goes down it indicates that either individuals are meeting less frequently, or disease controls are blocking transmission somehow (*e.g.* mask wearing). Note, disease transmission only actually occurs from contacts between infected and susceptible individuals. It is important to remember, therefore, that R_t *does not* account for the reduction in the susceptible fraction of the population as the epidemic progresses (this is incorporated into the effective reproduction number, as discussed below).

We outline the approach taken by Diekmann *et* al. to calculate $R(t)$. First, a set of compartments Ω for which individuals are infected (but not necessarily infectious) is identified. For the purposes of explanation, we refer to the example of an SEIR compartmental model with sex:



Here $\Omega=\{E_d,I_d\}$, where d goes over demographic possibilities (male M and female F).

A vector $\mathbf{v}=(E_M,E_F,I_M,I_F)^T$ is defined[30] to be the number of individuals in each of the infected compartments in Ω. In the deterministic case, the time evolution in $\mathbf{v}$ is given by

$$\frac{d\mathbf{v}}{dt}=(\mathbf{F}-\mathbf{\Sigma})\mathbf{v}, \tag{5}$$

where $\mathbf{F}$ is a matrix accounting for the rate of individuals *entering* compartments contained in Ω, and $\mathbf{\Sigma}$ is a matrix accounting for transitions *between* and *leaving* compartments within Ω.

From the model above we see that individuals enter Ω through the exposed E_M and E_F compartments, caused by infectious individuals in the I state:

$$\mathbf{F}=\begin{bmatrix} 0 & 0 & N_{S,M}\beta_M & N_{S,M}\beta_M \\ 0 & 0 & N_{S,F}\beta_F & N_{S,F}\beta_F \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \tag{6}$$

where $N_{S,M}$ and $N_{S,F}$ give the number of susceptible males and females in the population at the start time.

The matrix $\mathbf{\Sigma}$, representing transitions between and leaving the infected compartments Ω, is given by:

$$\mathbf{\Sigma}=\begin{bmatrix} \lambda_M & 0 & 0 & 0 \\ 0 & \lambda_F & 0 & 0 \\ -\lambda_M & 0 & \gamma_M & 0 \\ 0 & -\lambda_F & 0 & \gamma_F \end{bmatrix}, \tag{7}$$

where λ_s and γ_s are transition rates going between the E and I compartments for the two sexes. This matrix is constructed by considering each transition in the model in turn. Denoting the initial and final infected compartments to be i and j, matrix element Σ_{ii} gets a positive contribution given by the individual rate (because individuals are leaving compartment i) and Σ_{ji} gets a corresponding negative contribution (because those individuals are entering compartment j). Note, if j is not one of the infected compartments in Ω, this second contribution is ignored.

The inverse of the matrix in Eq.(7) is given by

$$\mathbf{\Sigma}^{-1}=\begin{bmatrix} \frac{1}{\lambda_M} & 0 & 0 & 0 \\ 0 & \frac{1}{\lambda_F} & 0 & 0 \\ \frac{1}{\gamma_M} & 0 & \frac{1}{\gamma_M} & 0 \\ 0 & \frac{1}{\gamma_F} & 0 & \frac{1}{\gamma_F} \end{bmatrix}. \tag{8}$$

This has a simple interpretation: If an individual enters compartment i, Σ_{ji}^{-1} gives the time, on average, that individual is expected to (eventually) spend in compartment j. For example, inspecting the first column in

[30] The superscript "T" strands from transpose, and this converts a row vector into a column vector.

Eq.(8) we see that if an individual enters compartment $\mathrm{E_M}$ (at the top) it will stay in $\mathrm{E_M}$ for time $1/\lambda_M$, spend no time in $\mathrm{E_F}$, time $1/\gamma_M$ in compartment $\mathrm{I_M}$, and no time in compartment $\mathrm{I_F}$.

We now construct the next generation matrix $\mathbf{K}$. We denote $\mathbf{w}_g$ to be a vector giving the number of individuals entering the different compartments in Ω in generation g. The equivalent vector for the next generation is given by[31]

$$\mathbf{w}_{g+1} = \mathbf{K}\mathbf{w}_g \quad \text{where} \quad \mathbf{K} = \mathbf{F}\boldsymbol{\Sigma}^{-1}. \tag{9}$$

The reasoning behind this expression is as follows: Suppose we consider an individual in generation g that enters compartment i. As described above, during its infected lifetime this individual spends on average Σ_{ji}^{-1} in each compartment j. But in doing so, Eq.(6) tells us that through its own infectiousness it will generate F_{kj} new infected individuals in each of the compartments k. This means that $w_{g+1,k}$ gains a contribution $F_{kj}\Sigma_{ji}^{-1}$ for each of the $w_{g,i}$ individuals in generation g that enter compartment i. Summing over all the possible values for i and j gives the relationship in Eq.(9).

Next, we ask the question what happens when we iterate Eq.(9) over many generations? It turns out that this equation can be solved[32], as shown in Appendix E. In summary, as the generations increase, so $\mathbf{w}_g$ becomes proportional to the normalised eigenvector $\mathbf{e}$ of the matrix $\mathbf{K}$ which has the largest eigenvalue. The value of this eigenvalue provides an estimate for $R(t)$ (by definition this is the average number of infections an individual in one generation causes in the next).

The eigenvector $\mathbf{e}$ itself has an intuitive explanation. Supposing that certain demographic groups come into contact more often, *e.g.* younger age groups. Irrespective of the age of the person who initiates an epidemic, eventually those driving disease progression will be the young and so $\mathbf{e}$ will have proportionately larger values associated with entering the exposed compartment for young individuals.

RNE(…) Effective reproduction number $R^e(t)$ – The effective reproduction number is the expected number of cases directly caused by an infected individual as a function of time t. Note, this *does* take into account the fact that as the epidemic progresses the fraction of susceptible individuals reduces (causing effective transmission upon contact of individuals to become less and less common). $R^e(t)$ is always less than $R(t)$ and if it reduces below 1, herd immunity in reached (*i.e.* the disease will naturally die out over time).

$R^e(t)$ is calculated using exactly the same method as above, except the population of susceptible individuals (*i.e.* $N_{S,M}$ and $N_{S,F}$ in Eq.(6)) are taken at the given point in time t (rather than at the start time).

RNC(…) Computational reproduction number $R^c(t)$ – This numerical approach takes each infection event in turn and apportions it between all the individuals that could have caused that infection[33]. This leads to each individual potentially having a period of time over which it is infected[34] and an estimate for the number of

[31] In fact, because individuals usually only enter Ω through a limited number of states ($\mathrm{E_M}$ and $\mathrm{E_F}$ in our example), then, without loss of generality, $\mathbf{w}$ and $\mathbf{K}$ can be defined for just these states. This is a commonly used technique to increase computational efficiency.

[32] We assume the timescale of the initial phase of the epidemic is slow compared to time variation in other model parameters.

[33] In cases in which individual or fixed effects are used for the infectious population, this apportioning is done in proportion to the relative infectivity of individuals.

[34] Or in cases in which individuals can become reinfected, there may be more than one infected period for an individual.

infections it "caused". Consequently, a reproduction number can be calculated on an individual basis. This is then converted onto a continuous distribution by means of averaging over all individuals infected at a given time. When no individuals are infected, a nominal value of zero is used to represent undefined.

GT(…) Generation time *G(t)* – This is the time interval between when an individual becomes infected and the average time that individuals transmits their infection to others. This quantity is important because it specifies how fast infections are spreading within the system with the passing of each generation.

Using the SEIR example above, the generation time is calculated as follows. The fraction of individuals entering compartment i is given by the corresponding element in the eigenvector e_i. Equation (8) shows a matrix Σ_{ji}^{-1} giving the average time those individuals then go on to spend in compartment j. Whilst in compartment j they initiate new infections in each of the compartments k at a rate given by F_{kj} (see Eq.(6)). The average timing of those new infections is given by the elements of the following vector:

$$\boldsymbol{\tau} = \left(\tfrac{1}{2}\tfrac{1}{\lambda_M}, \tfrac{1}{2}\tfrac{1}{\lambda_F}, \tfrac{1}{\lambda_M} + \tfrac{1}{2}\tfrac{1}{\gamma_M}, \tfrac{1}{\lambda_F} + \tfrac{1}{2}\tfrac{1}{\gamma_F}\right)^T. \tag{10}$$

Combining these contributions yields the final result:

$$G(t) = \frac{\sum_{k,j,i} F_{kj}\tau_j \Sigma_{ji}^{-1} e_i}{\sum_{k,j,i} F_{kj}\Sigma_{ji}^{-1} e_i}. \tag{11}$$

GTE(…) Effective generation time *G^e(t)* – This is the same as above, except here the population of susceptible individuals (*i.e.* $N_{S,M}$ and $N_{S,F}$ in Eq.(6)) is taken at the given point in time t (rather than at the start time).

GTC(…) Computational generation time *G^c(t)* – This numerical approach takes each infection event in turn and apportions it between all the individuals that could have caused that infection along with the infection time. This leads to each individual potentially having a period of time over which it is infected and an estimate for the mean time it "caused" other infections. Consequently, a generation time can be calculated on an individual basis. This is then converted onto a continuous distribution by means of averaging over all individuals infected at a given time. When no individuals are infected, a nominal value of zero is used to represent undefined.

Appendix E: Solving iterative matrix equations

This appendix outlines the solution to the matrix equation in Eq.(9).

We refer to $\mathbf{e}$ as an "eigenvector" and λ as an "eigenvalue" of a matrix $\mathbf{K}$ if it solves the equation

$$\mathbf{K}\mathbf{e} = \lambda\mathbf{e}. \tag{12}$$

In general, a square matrix of size N will actually have N eigenvectors $\mathbf{e}_j$ and eigenvalues λ_j, ordered such that λ_1 is the highest and λ_N is the lowest. These can collectively be written as

$$\mathbf{KU} = \mathbf{U}\begin{bmatrix} \lambda_1 & 0 & 0 & \cdots \\ 0 & \lambda_2 & 0 & \cdots \\ 0 & 0 & \lambda_3 & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix}, \tag{13}$$

where $\mathbf{U}=[\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3,\ldots]$ is a matrix made up of the eigenvectors. Multiplying both sides of Eq.(13) by the inverse matrix $\mathbf{U}^{-1}$ and substituting this expression for $\mathbf{K}$ into Eq.(9) gives

$$\begin{aligned} \mathbf{w}_g &= \mathbf{K}^{g-1}\mathbf{w}_1 \\ &= \left(\mathbf{U}\begin{bmatrix} \lambda_1 & 0 & 0 & \cdots \\ 0 & \lambda_2 & 0 & \cdots \\ 0 & 0 & \lambda_3 & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix}\mathbf{U}^{-1} \right)^{g-1} \mathbf{w}_1 \\ &= \mathbf{U}\begin{bmatrix} \lambda_1^{g-1} & 0 & 0 & \cdots \\ 0 & \lambda_2^{g-1} & 0 & \cdots \\ 0 & 0 & \lambda_3^{g-1} & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix}\mathbf{U}^{-1}\mathbf{w}_1. \end{aligned} \tag{14}$$

A key feature of this relationship is that as the generation number g increases, so the diagonal element that corresponds to the largest eigenvalue λ_1 dominates over all the others (which also means that $\mathbf{w}_g$ become proportional to the eigenvector $\mathbf{e}_1$). Because the overall number of individuals in $\mathbf{w}_g$ goes up by a proportion λ_1 each generation, so λ_1 provides an approximation to $R(t)$.

Appendix F: Further information about priors

This appendix provides some further details about the use of priors in BICI.

Uninformative priors

In many cases nothing *a priori* is known about a model parameter. In such cases an "uninformative" prior would want to be used. There are, however, several different candidate distributions that can be used for this purpose:

Flat prior – The most intuitively obvious approach is to use an entirely flat prior spanning all of parameter space[35]. However, because such a distribution is improper it can't be sampled from, and so isn't implemented in BICI.

Uniform prior – Here the prior distribution takes a constant value over all realistically possible values the parameter can take.

Jeffreys prior – This prior is relatively weak and, by construction, is invariant under reparameterisation. It has a different functional form, depending of the distribution used in the model, that must be derived (see below).

Example

Consider an exponential distribution with rate r. One simple option would be to give this a uniform prior in the range between $r_{\min}$ and $r_{\max}$:

$$\pi(r) = \begin{matrix} c \\ 0 \end{matrix} \Bigg\} \begin{matrix} \text{if } r_{\min} < r < r_{\max} \\ \text{otherwise} \end{matrix}, \tag{15}$$

where $c = 1/(r_{\max} - r_{\min})$ is a constant value. However, an exponential distribution can also be parameterised in terms of a mean μ, which is functional related to the rate through $\mu = 1/r$. Reparameterising Eq.(15) leads to a new prior:

$$\pi(\mu) = \begin{matrix} d/\mu^2 \\ 0 \end{matrix} \Bigg\} \begin{matrix} \text{if } \mu_{\min} < \mu < \mu_{\max} \\ \text{otherwise} \end{matrix} \tag{16}$$

where d is a constant and $\mu_{\min} = 1/r_{\max}$ and $\mu_{\max} = 1/r_{\min}$. The important point to notice about Eq.(16) is that this prior is no longer uniform. What appear to be uninformative when thought of as a rate, has suddenly become informative in a mean. This discrepancy is somewhat unsatisfactory, given the original intention was to be "uninformative".

An alternative is to use the Jeffreys prior (see the next section for how this is derived). When the exponential distribution is defined by a rate, the Jeffreys prior is given by:

$$\pi(r) = \begin{matrix} c/r \\ 0 \end{matrix} \Bigg\} \begin{matrix} \text{if } r_{\min} < r < r_{\max} \\ \text{otherwise} \end{matrix}, \tag{17}$$

[35] That is from 0 to ∞ for positive parameter, otherwise from -∞ to ∞.

with constant $c = 1/\log(r_{\max}/r_{\min})$.

When the exponential distribution is defined by a mean the Jeffreys prior is given by:

$$\pi(\mu) = \left.\begin{matrix} d/\mu \\ 0 \end{matrix}\right\} \begin{matrix} \text{if } \mu_{\min} < \mu < \mu_{\max} \\ \text{otherwise} \end{matrix} \tag{18}$$

with constant $d = 1/\log(\mu_{\max}/\mu_{\min})$.

A simple change of variable shows that Eqs.(17) and (18) are identical (so an analysis performed with one would give exactly the same result as with the other). Because of this independence of reparameterisation, it is recommended that Jeffreys priors are used for key parameters associated with transition distributions.

Deriving Jeffreys priors

This section shows Jeffreys priors for a variety of different distributions. In Bayesian statistics, the Jeffreys prior is a non-informative prior distribution with density function proportional to the square root of the determinant of the Fisher information matrix:

$$\pi(\theta) \propto \sqrt{|I(\theta)|}. \tag{19}$$

Given some probability distribution $f(x|\theta)$, the Fisher information matrix is defined by

$$I(\theta)_{i,j} = \mathrm{E}\left[\frac{\partial^2 \log(f)}{\partial\theta_i \partial\theta_j}\right], \tag{20}$$

where $\mathrm{E}[\ldots]$ is the expectation[36]. Below we go through the derivation of the Jeffreys prior for an exponential distribution with a rate.

Exponential distribution with rate

The probability distribution function for sampling a time t given a rate r is given by

$$f(t \mid r) = \tfrac{1}{r} e^{-rt}. \tag{21}$$

The log of this is given by:

$$\log(f) = -\log(r) - rt. \tag{22}$$

The first derivative is:

$$\frac{\partial \log(f)}{\partial r} = -\frac{1}{r} - t. \tag{23}$$

The second derivative is:

$$\frac{\partial^2 \log(f)}{\partial r^2} = \frac{1}{r^2}. \tag{24}$$

[36] The expectation is calculated by integrated the function multiplied by the distribution $f(x|\theta)$) over all x.

The Fisher information matrix is given by:

$$I(r) = \mathrm{E}\left[\frac{\partial^2 \log(f)}{\partial r^2}\right] = \int_0^\infty \frac{1}{r^2}\frac{1}{r}e^{-rt}dt = \frac{1}{r^2}. \quad (25)$$

By using Eq.(19), the prior is:

$$\pi(r) \propto \frac{1}{r}. \quad (26)$$

Note, this prior is improper, so in practice limits must be placed on the range over which the rate r can extend (*i.e.* a plausible range), as shown in Eq.(17). This is implemented in BICI using the "inverse" prior option.

Jeffreys priors for transition distributions

Jeffreys priors for distributions with one parameter are relatively easy to derive. When two (or more) parameters are involved, the results depend on whether one of the parameters is being fixed or not. Table F1 goes through all the possibilities for the different transition distributions used in BICI.

Note, some of the analytical expressions for the exact Jeffreys prior are somewhat complicated. In these instances, we have also simplified in the limit of small coefficient of variation (using the ≈ symbol). To test this approximation, Fig. F1 shows a comparison, for each of the distributions, between a numerical evaluation of Eq.(19) (red lines), the analytical expressions (black lines) and any potential approximations (red lines). Because distributions are only known up to a constant factor, so any displacement between the curves in unimportant[37]. The key point to note is whether the lines have the same slope or not. We find that generally there is very good agreement (apart from for the Weibull distribution, but this may be as a result of numerical inaccuracies in evaluating the Weibull function). This justifies us in making use of the approximations in any analysis in BICI.

Jeffreys priors are implemented in BICI either using the "inverse" option, typically for means and rates, or the "power" option, *e.g.* for non-inverse priors such as cv^{-2}.

Distribution	**pdf**	**Jeffreys prior**
exp(r)	$\frac{1}{r}e^{-rt}$	$\pi(r) \propto \frac{1}{r}$
exp(μ)	$\mu e^{-t/\mu}$	$\pi(\mu) \propto \frac{1}{\mu}$
gamma(μ,cv)	$\frac{1}{\Gamma(k)}\mu^{-k}t^{k-1}e^{-t/\mu}$	$\pi(\mu, cv) \propto \frac{1}{\mu}\frac{\sqrt{\psi'(\frac{1}{cv^2}) - cv^2}}{cv^4} \approx \frac{1}{\mu}\frac{1}{cv^2}$
gamma(μ,cv) (fixed cv)	$\frac{1}{\Gamma(k)}\mu^{-k}t^{k-1}e^{-t/\mu}$	$\pi(\mu) \propto \frac{1}{\mu}$
gamma(μ,cv) (fixed μ)	$\frac{1}{\Gamma(k)}\mu^{-k}t^{k-1}e^{-t/\mu}$	$\pi(cv) \propto \frac{\sqrt{\psi'(\frac{1}{cv^2}) - cv^2}}{cv^3} \approx \frac{1}{cv}$

[37] Note, these plots are logarithmic in the x and y axes.

Erlang(μ,k) (fixed k)	$\frac{1}{\Gamma(k)}\mu^{-k}t^{k-1}e^{-t/\mu}$	$\pi(\mu) \propto \frac{1}{\mu}$
log-normal(μ,cv)	$\frac{1}{x\sqrt{2\pi\sigma^2}}e^{-(\log(x)-\log(\mu)+\frac{1}{2}\sigma^2)^2/(2\sigma^2)}$ where $\sigma^2 = \log(1+cv^2)$	$\pi(\mu,cv) \propto \frac{1}{\mu}\frac{cv}{\left(\log(1+cv^2\right)^{3/2}(1+cv^2)} \approx \frac{1}{\mu}\frac{1}{cv^2}$
log-normal(μ,cv) (fixed cv)	$\frac{1}{x\sqrt{2\pi\sigma^2}}e^{-(\log(x)-\log(\mu)+\frac{1}{2}\sigma^2)^2/(2\sigma^2)}$ where $\sigma^2 = \log(1+cv^2)$	$\pi(\mu) \propto \frac{1}{\mu}$
log-normal(μ,cv) (fixed μ)	$\frac{1}{x\sqrt{2\pi\sigma^2}}e^{-(\log(x)-\log(\mu)+\frac{1}{2}\sigma^2)^2/(2\sigma^2)}$ where $\sigma^2 = \log(1+cv^2)$	$\pi(cv) \propto \frac{cv}{\log(1+cv^2)(1+cv^2)} \approx \frac{1}{\mu}\frac{1}{cv}$
weibull(λ,k)	$\left(\frac{k}{\lambda}\right)\left(\frac{x}{\lambda}\right)^{k-1}e^{-\left(\frac{x}{\lambda}\right)^k}$	$\pi(\lambda,k) \propto \text{compilated} \approx \frac{1}{\lambda}\frac{1}{k^2}$
weibull(λ,k) (fixed k)	$\left(\frac{k}{\lambda}\right)\left(\frac{x}{\lambda}\right)^{k-1}e^{-\left(\frac{x}{\lambda}\right)^k}$	$\pi(\lambda) \propto \frac{1}{\lambda}$
weibull(λ,k) (fixed λ)	$\left(\frac{k}{\lambda}\right)\left(\frac{x}{\lambda}\right)^{k-1}e^{-\left(\frac{x}{\lambda}\right)^k}$	$\pi(k) \propto \text{compilated} \approx \frac{1}{k}$

Table F1: Jeffreys priors for transition distributions –The following notation is used: t=time, r=rate, μ =mean, k=shape, λ=scale, σ=standard deviation, cv=coefficient of variation. The coefficient of variation cv is related to the shape parameter through $k = 1/cv^2$.

Distribution	**pdf**	**Jeffreys prior**
normal(μ,σ)	$\frac{1}{\sqrt{2\pi\sigma^2}}e^{-(x-\mu)^2/(2\sigma^2)}$	$\pi(\mu,\sigma) \propto \frac{1}{\sigma^2}$
normal(μ,σ) (fixed σ)	$\frac{1}{\sqrt{2\pi\sigma^2}}e^{-(x-\mu)^2/(2\sigma^2)}$	$\pi(\mu) \propto \text{constant}$
normal(μ,σ) (fixed μ)	$\frac{1}{\sqrt{2\pi\sigma^2}}e^{-(x-\mu)^2/(2\sigma^2)}$	$\pi(\sigma) \propto \frac{1}{\sigma}$
mvn($\boldsymbol{\mu}$,$\boldsymbol{\Sigma}$)	$\frac{1}{(2\pi)^{d/2}\lvert\boldsymbol{\Sigma}\rvert^{1/2}}e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})}$	$\pi(\boldsymbol{\mu},\boldsymbol{\Sigma}) \propto \lvert\boldsymbol{\Sigma}\rvert^{-\frac{d}{2}-1}$
mvn($\boldsymbol{\mu}$,$\boldsymbol{\Sigma}$) (fixed $\boldsymbol{\Sigma}$)	$\frac{1}{(2\pi)^{d/2}\lvert\boldsymbol{\Sigma}\rvert^{1/2}}e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})}$	$\pi(\mu) \propto \text{constant}$
mvn($\boldsymbol{\mu}$,$\boldsymbol{\Sigma}$) (fixed $\boldsymbol{\mu}$)	$\frac{1}{(2\pi)^{d/2}\lvert\boldsymbol{\Sigma}\rvert^{1/2}}e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})}$	$\pi(\boldsymbol{\Sigma}) \propto \lvert\boldsymbol{\Sigma}\rvert^{-\frac{d}{2}-\frac{1}{2}}$

Table F2: Jeffreys priors for other distributions –The following notation is used: x=value, μ =mean, σ=standard deviation, $\mathbf{x}$=vector value, $\boldsymbol{\mu}$ =vector mean, $\boldsymbol{\Sigma}$=covariance matrix, $|\boldsymbol{\Sigma}|$=determinant, d=dimension.

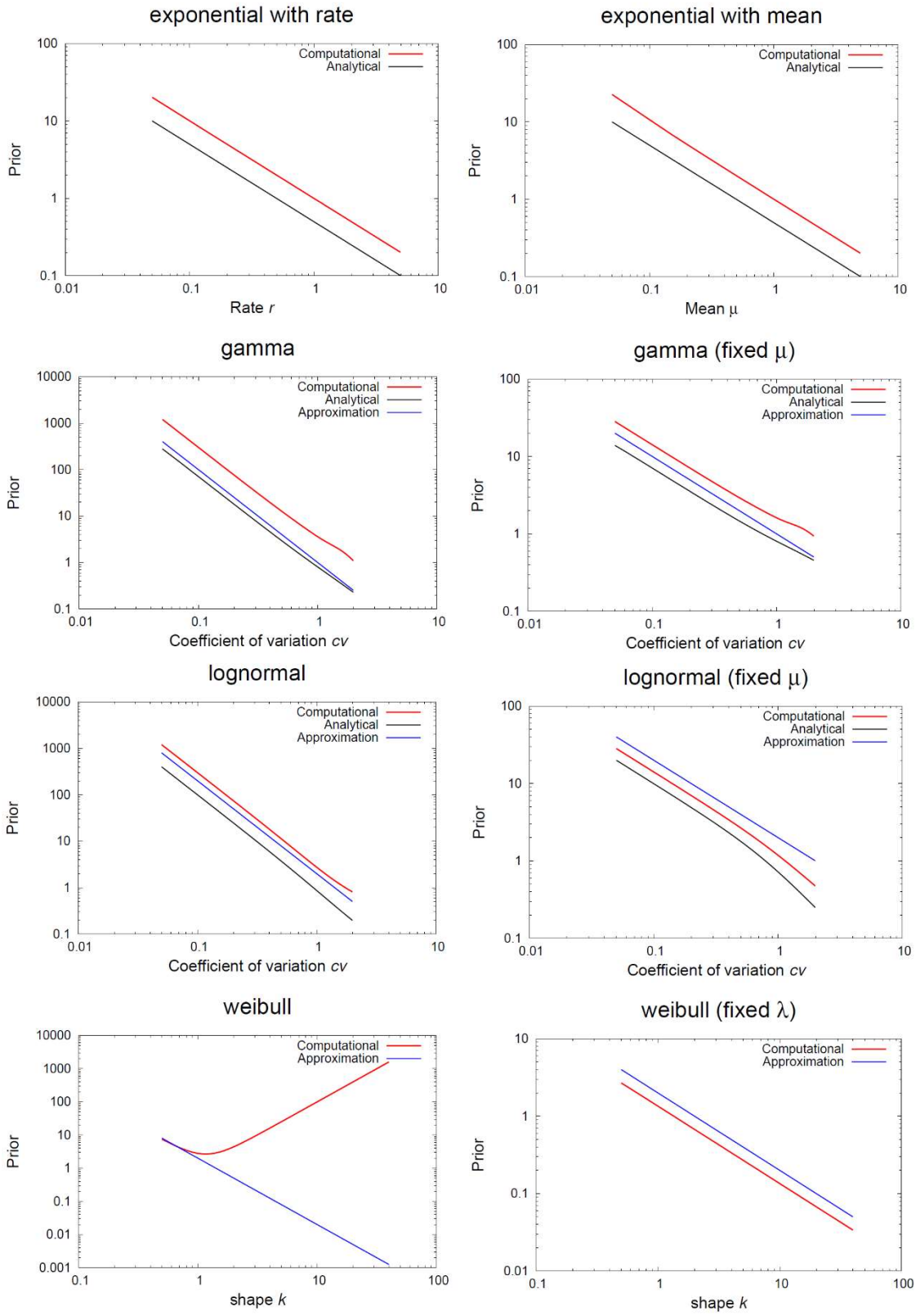


Figure F1 – *Jeffreys priors for transition distributions*. This graph shows Jeffreys priors for the distributions in Table F1. The red lines are generated by numerically integrating Eq.(19). The black curves represent the analytically derived expressions in the right-hand column of Table F1. The blue curves show any approximations (denoted by ≈). Note, because curves are only known up to a constant value, so vertical displacement is unimportant.

Jeffreys priors for normal distribution

Table F2 shows Jeffreys priors for normal and multivariate normal distributions. In contrast to those shown in Table F1, we find that these are independent of the mean. Numerical verification of the scaling relationships is given in Fig. F2.

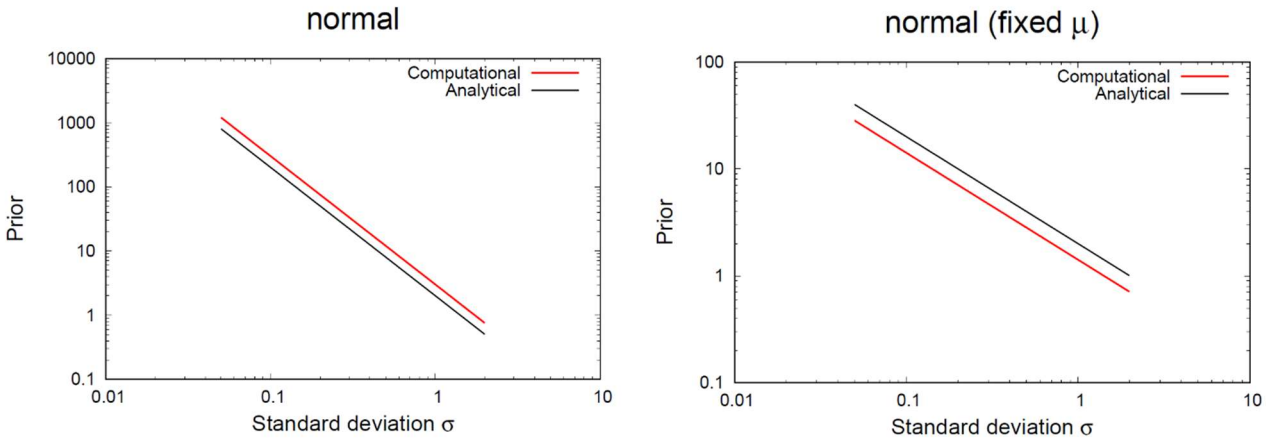


Figure F2 – *Jeffreys priors for normal distributions*. This graph shows Jeffreys priors for the distributions in Table F2. The red lines are generated by numerically integrating Eq.(19). The black curves represent the analytically derived expressions in the right-hand column of Table F2. The blue curves show any approximations (denoted by ≈). Note, because curves are only known up to a constant, so vertical displacement is unimportant.

Jeffreys prior for covariance matrices for individual effects

Two possible prior distributions can be applied to individual effects:

- "mvn-jeffreys(min,max)". The minimum sets a lower limit on the determinant of the covariance matrix and the maximum sets the upper limit for the diagonals of the covariance matrix (to ensure prior is not improper). The functional form is based on the MVN Jeffreys prior in Table 2 for a fixed mean:

$$\pi(\Omega) \propto \begin{cases} |\Omega|^{-\frac{d}{2}-\frac{1}{2}} \\ 0 \end{cases} \quad \begin{array}{l} \text{if } |\Omega| > \min^d,\ \Omega_{i,i} < \max,\ -c < \omega_{i \neq j} < c \ \text{ for all } i,j \\ \text{otherwise} \end{array} \qquad (27)$$

  Here the correlations $\omega_{i,j} = \Omega_{i,j} \Big/ \sqrt{\Omega_{i,i}\Omega_{j,j}}$ is constrained by parameter c=0.9 (this was introduced to ensure numerical stability of the algorithm).

  Numerical analysis has shown that suitable values for 'min' are: 0.1 (d=1), 0.2 (d=2), 0.5 (d=3). Going below these limits can lead to MCMC mixing problems. Note, however, that Eq.(27) does not restrict the minimum for the variances themselves, which are free to go to zero.

- "mvn-uniform(min,max)" simply specifies minimum and maximum values for variances:

$$\pi(\Omega) \propto \left.\begin{matrix} 1 \\ 0 \end{matrix}\right\} \begin{matrix} \text{if } \min < \Omega_{i,i} < \max\,,\ -c < \omega_{i \neq j} < c \ \text{ for all } \ i,j \\ \text{otherwise} \end{matrix} \tag{28}$$

Modified Dirichlet prior for factors

The modified Dirichlet prior "mdir(σ)" is used for parameter factors (see §2.2.9). We take f_i to represent the value of a parameter element i [38]. This element can also have a weight w_i associated with it (by default set to one). For a factor the weighted average of the parameter value must be one:

$$\sum_i \left(f_i w_i\right) \Big/ \sum_i w_i = \sum_i f_i \phi_i = 1, \tag{29}$$

where

$$\phi_i = w_i \Big/ \sum_i w_i \,. \tag{30}$$

To ensure the condition in Eq.(29), each terms $f_i \phi_i$ is set to have a Dirichlet prior Dir(α_i) with

$$\alpha_i = \left(\frac{N-1}{\sigma^2} - 1 \right) \phi_i, \tag{31}$$

where N is the total number of elements in f. This choice leads to the prior for f_i having a mean of one and a variance

$$\sigma^2 \frac{\frac{1}{\phi_i} - 1}{N-1}. \tag{32}$$

Note, when all elements have equal weight, *i.e.* $\phi_i = 1/N$, this simplifies to a variance of σ^2 (or equivalently a standard deviation of σ).

In the general case, those elements with a smaller weight will have a larger variance than average, and *vice-versa*. The choice σ=0.5 represents a relatively uninformative prior choice (allowing for a large ~50% variation in factor value).

Appendix G: Run-time warnings

This appendix outlines various run-time warnings can be generated after BICI has been executed. These do not prevent results from being generated, but rather are designed to inform the user about the potential unreliability of the results.

Time-step – This warning is generated if the time-step is either chosen to be too large (leading to discretisation errors) or too small (becoming computationally inefficient).

Typical messages are as follows:

[38] Note, parameter factors are usually vectors and so i represents elements of the vector. In principle, however, parameter factors can also be matrices or tensors, and here i simply iterates though each element.

"High transition rates mean that there is a potential for a finite time-step discretisation error. It is recommended that this model is run with a time-step below '0.5'. The following transitions are affected: S→I."

"This is being run with a relatively small time-step. Analysis suggests it could be reliably run up to a time-step '10'."

Execution time – These warnings are used to indicate when inference hasn't been run for sufficiently long.

This estimates how many MCMC update would be needed for convergence:

"MCMC diagnostics suggests it has not been run for long enough. An estimated 16000 updates are required."

Note, if this estimated number of updates is much larger than that used to run the analysis (for example if here only 1000 updates had been used), this value may not be particularly reliable.

This indicates which parameters are not converging based on being less than an effective sample size statistic of 200 (see §4.4.7 for details):

"Parameter below the 200 ESS threshold: β"

This indicates which parameters are not converging based on exceeding a Gelman-Rubin statistic of 1.01 (see §4.4.7 for details):

"Parameter above the 1.01 GR threshold: β"

Data outside range – Sometimes it is convenient to only run inference on a subset of the data. In particular setting the time range as smaller than provided by the data. This means that some of the data is not used, which generates the warning:

"Data outside of time range is ignored in source 'Infections'."

Unexpected added individuals – In individual-based models typically the user specifies individuals which are added to the system (through the 'Add Ind.' data type) and then various observations made on those individuals. It's important to note, however, that BICI will automatically create individuals which are observed but not added. There are many circumstances when this is valid (such as in the ecological setting where trapped animals are *a priori* unknown). Sometimes, however, the IDs on the observations are actually accidental errors, and the should really reference the added individuals (*e.g.* resulting from a spelling mistake). To guard against this possibility a warning message is generated:

"The following individuals are not explicitly added to the system (is this correct?): 'Ind-1', 'Ind-2'…"

Derived functions – Derived functions allows for the estimation of important quantities such as the reproduction number (see Appendix D). This warning shows that the calculation of the derived quantity is only approximate, because it replaces any non-Markovian transitions with exponential distributions with an equivalent rate:

"Calculation of 'R' is approximate due to non-Markovian transitions"

Internal BICI warnings – These warnings indicate potential problems with the code execution. They are of a technical nature and are aimed at improving code reliability. The message given is:

> "This run has generated algorithm warnings. Please check the diagnostic file(s) for details."

Appendix H: Parameter visualisations

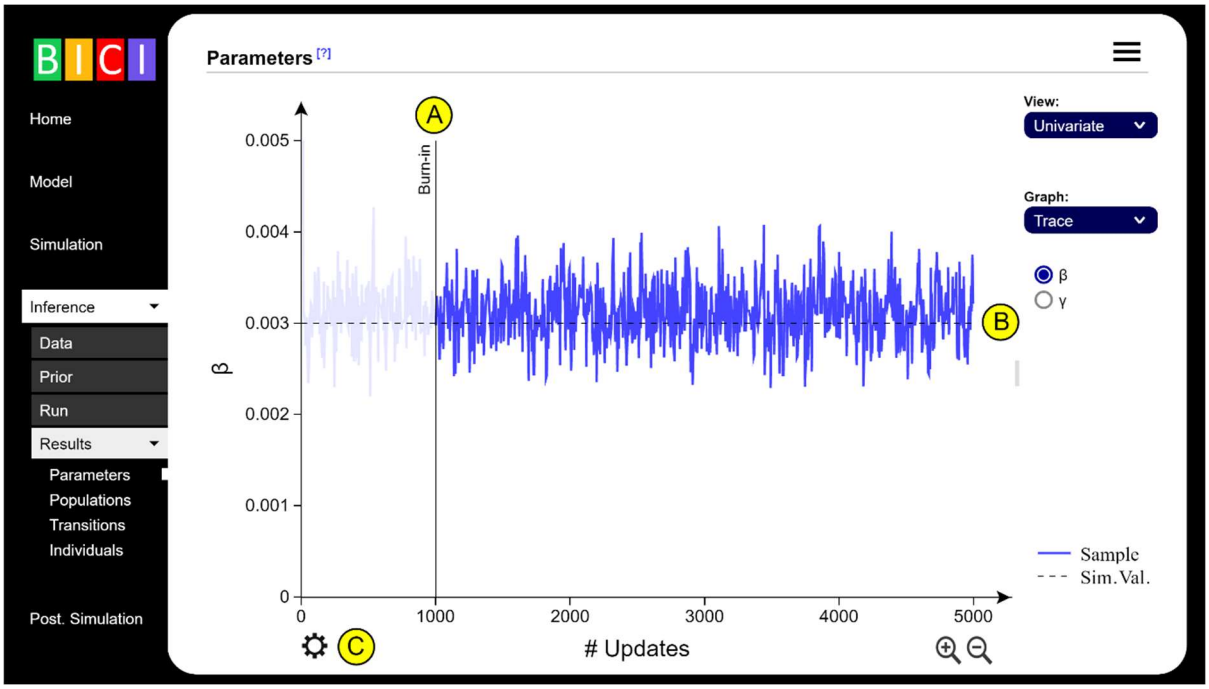


Figure H1 – *Trace plot*. MCMC works by iteratively performing "updates" and drawing posterior samples. The number of updates is shown on the *x*-axis and a selected parameter on the *y*-axis. Ideally, samples would be randomly sampled from the posterior, but in reality they are correlated (which manifests itself as structure within these trace plots). This example is for just a single MCMC chain, but multiple chains can be represented by lines with different colours. A: This vertical line represents the end of the burn-in period, B: The horizontal dashed line indicates the parameter value used to simulate the data (if applicable), C: Under settings, the burn-in period can be altered or the dashed line removed.

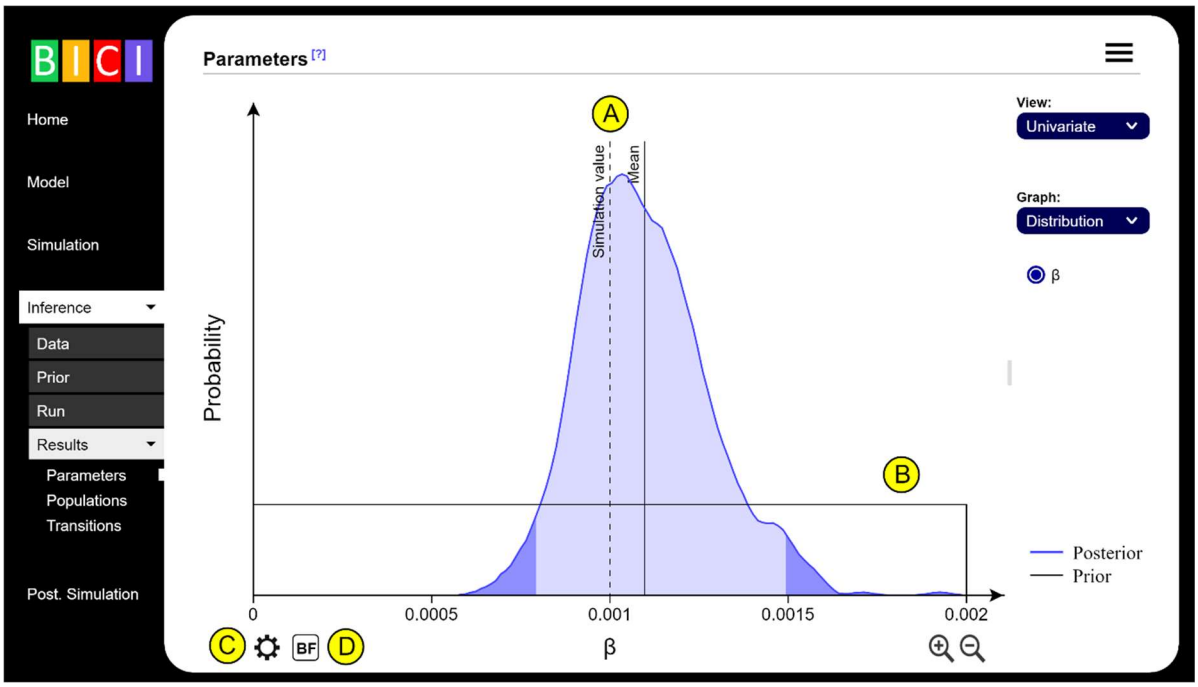


Figure H2 – *Distribution plot.* This shows a posterior distribution plot for a selected parameter (the darker blue areas indicate the 2.5% tails). A: The value used to simulate the data (if applicable), B: The prior distribution (solid black line), C: Under settings, either kernel density estimation (with a triangular kernel of specified size), or binning (with specified bin number) can be selected, D: Calculate the Bayes factor (see §4.4.6).

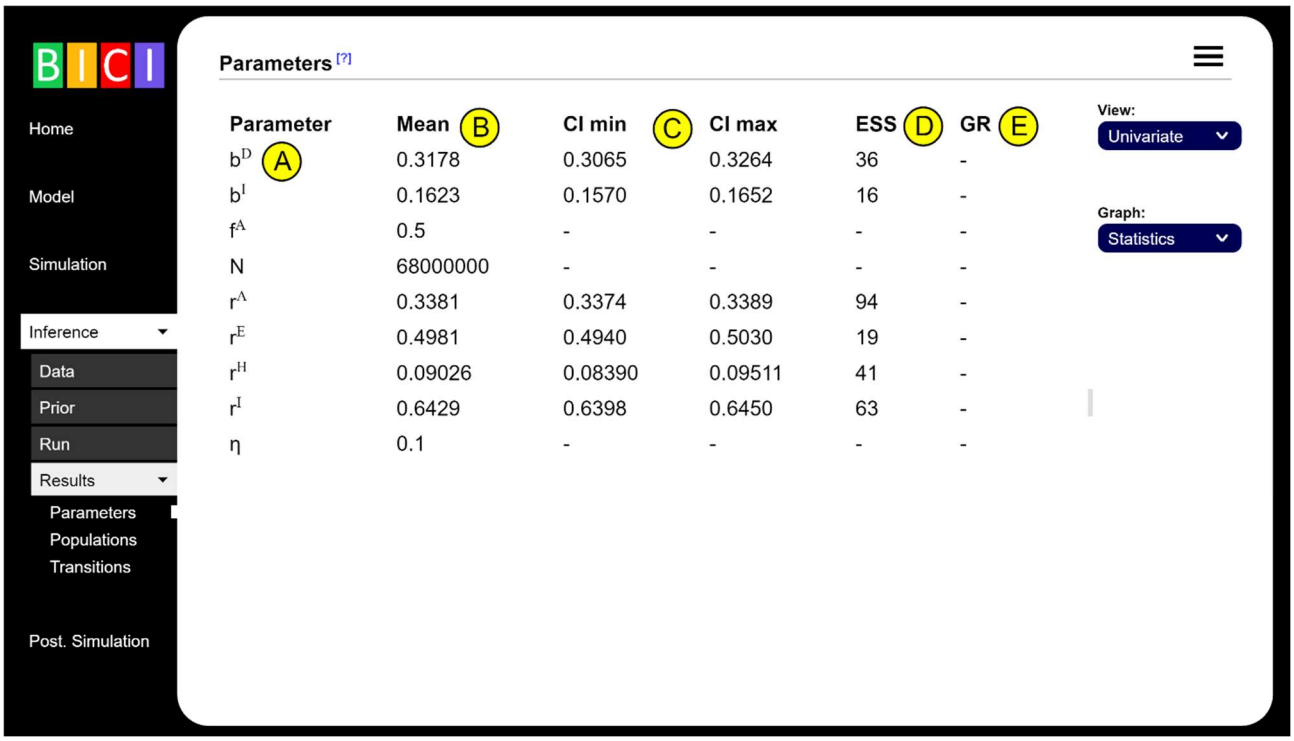


Figure H3 – *Statistics*. Provides posterior statistics for model parameters. A: Parameter name (this example shows univariate parameters, but for multivariate parameters different rows correspond to different elements), B: Posterior mean, C: 95% credible interval (note, this is unspecified for constant parameters), D: Effective sample size (ESS), E: Gelman-Rubin statistic (GR), not shown in this particular case because only a single MCMC chain was run. See §4.4.7 for details on ESS and GR.

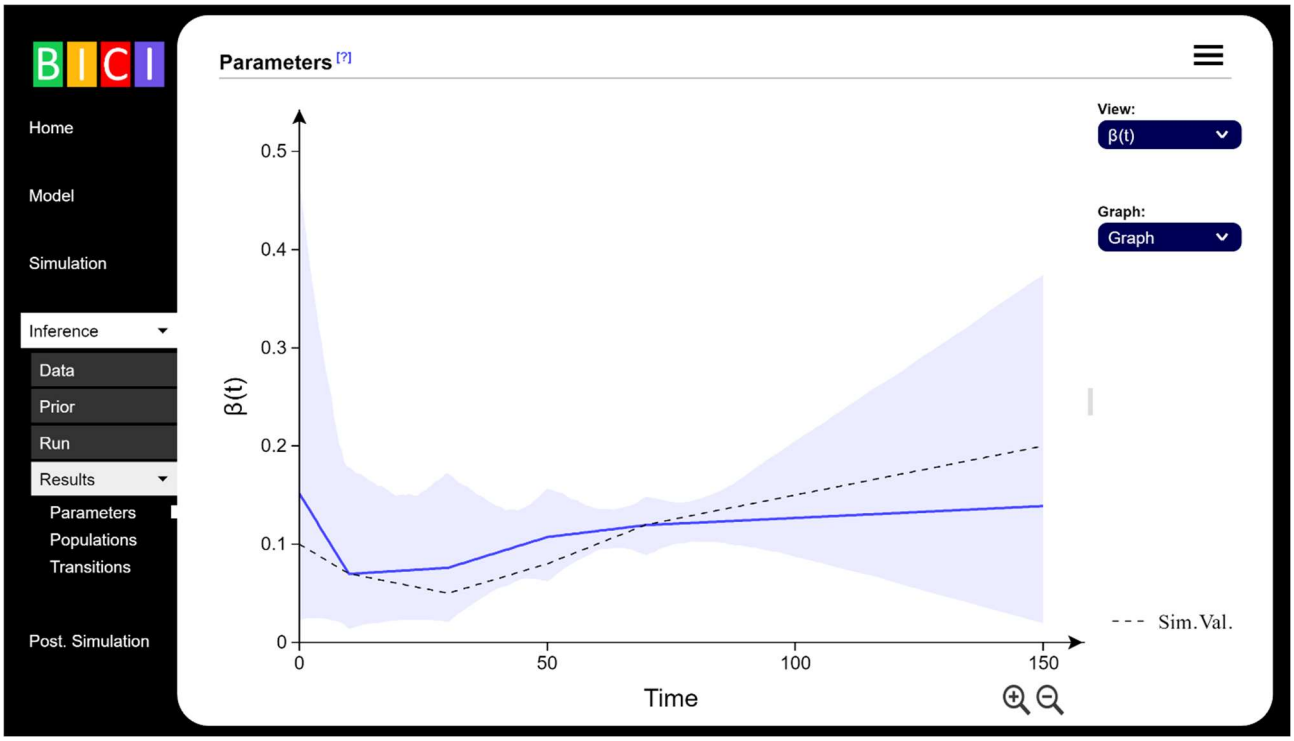


Figure H4 – *Graph*. This shows a graph representing time variation in spline parameter $\beta(t)$. The dashed line gives the values used for simulation. The blue line provides the posterior mean (with shading representing 95% credible intervals).

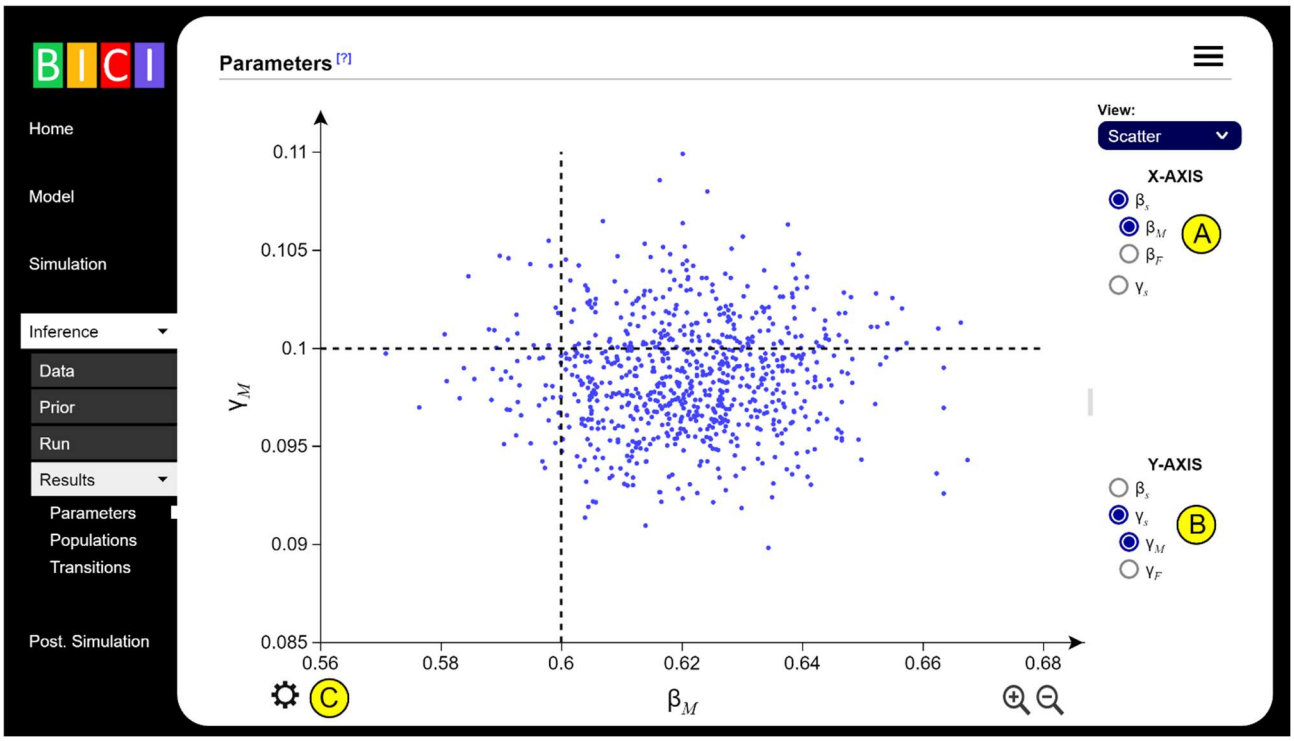


Figure H5 – *Scatter plot*. This shows a scatter plot for two model parameters, based on different posterior samples. A: Select one parameter on the *x*-axis, B: Select another parameter on the *y*-axis, C: Under settings, the simulated values (shown by the dashed lines) can be switch on or off.

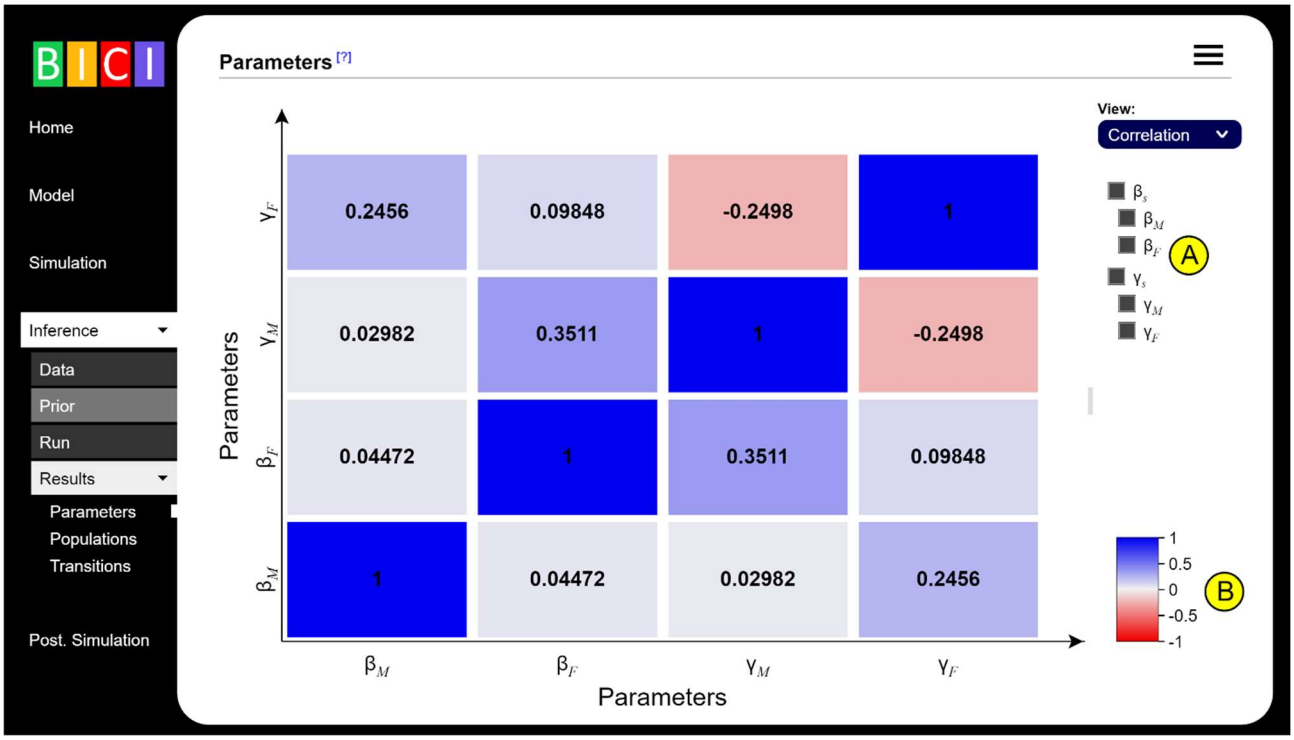


Figure H6 – *Correlation matrix*. This shows posterior correlations between different parameters in the model (as measured by the Pearson correlation coefficient) 1=completely correlated, -1=completely anti-correlated. A: Select which parameters appear in the matrix, B: Key indicating level of correlation (blue for correlated and red for anti-correlated).

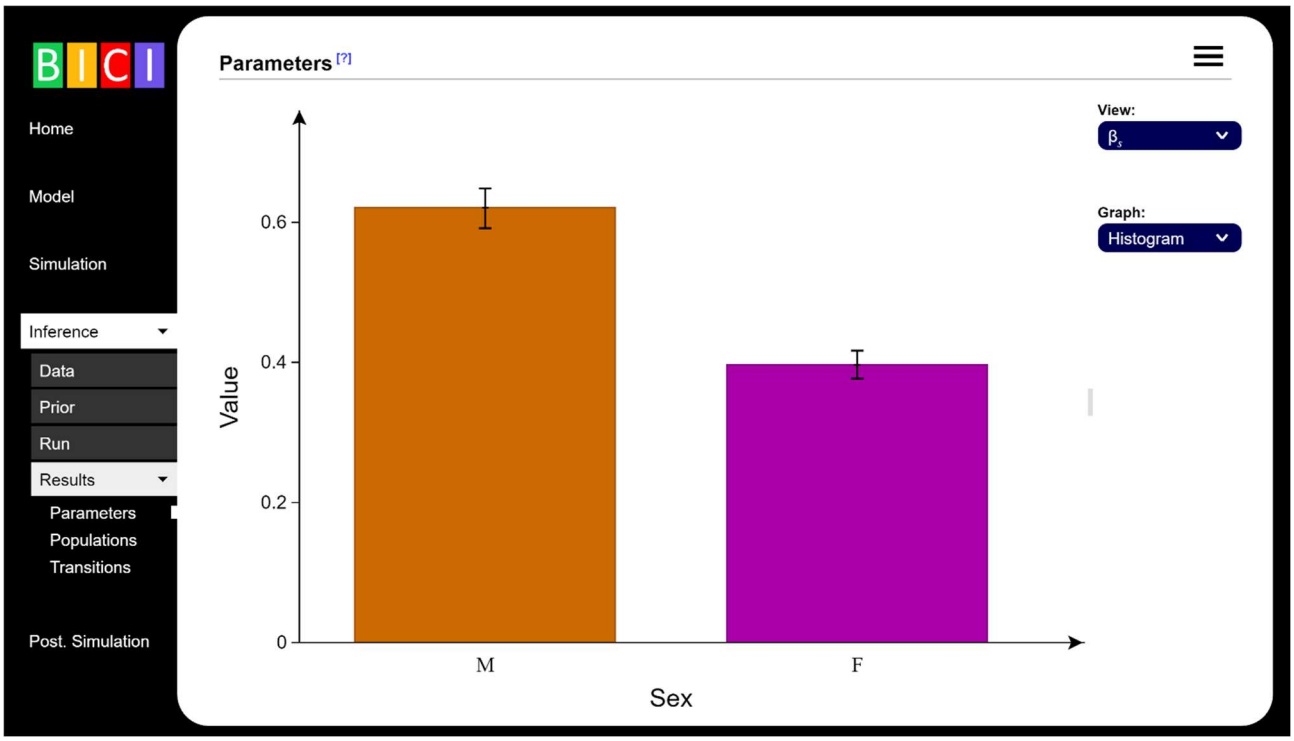


Figure H7 – *Histogram*. This can be used to visualise vectors, where each bar corresponds to a compartment within a classification. The error bars represent 95% credible intervals in the posterior. Time-varying histograms can also be plotted, with animation controls as in Fig. D3.

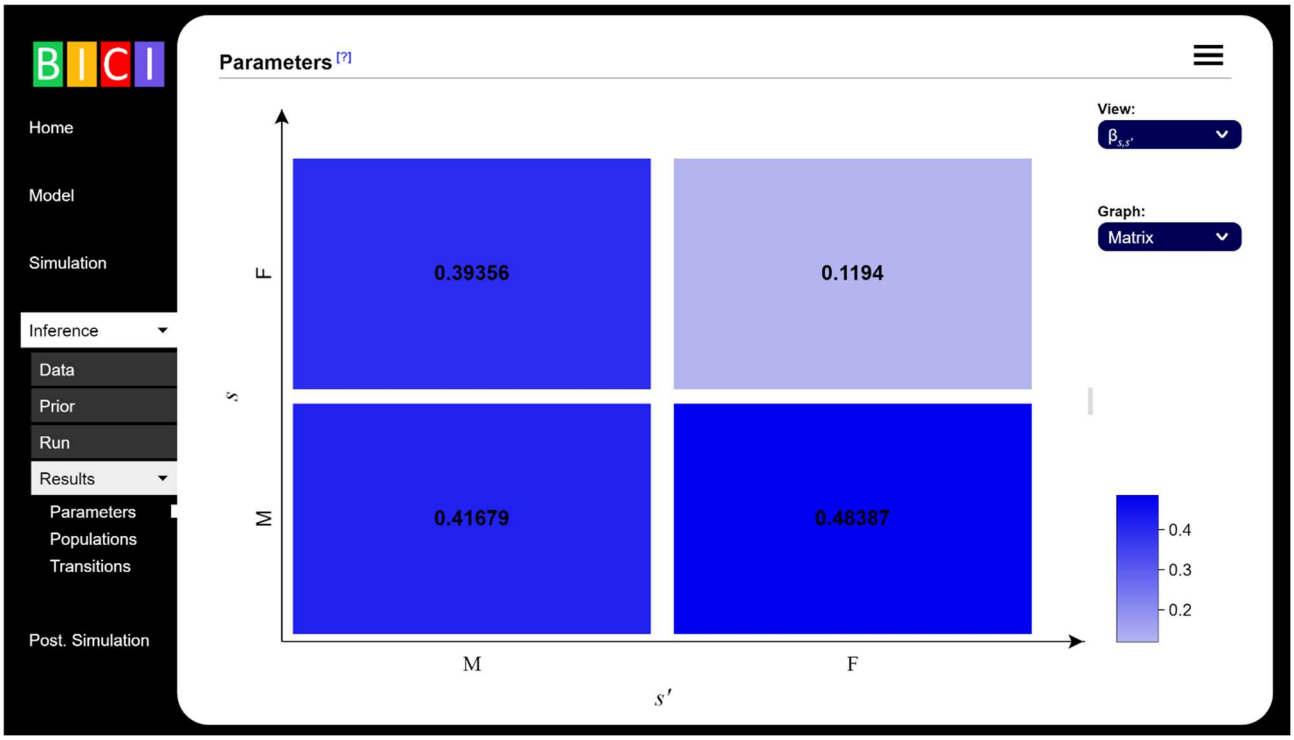


Figure H8 – *Matrix*. This shows the posterior mean for elements of a matrix. Time-varying matrices can also be plotted, with animation controls as in Fig. D3.

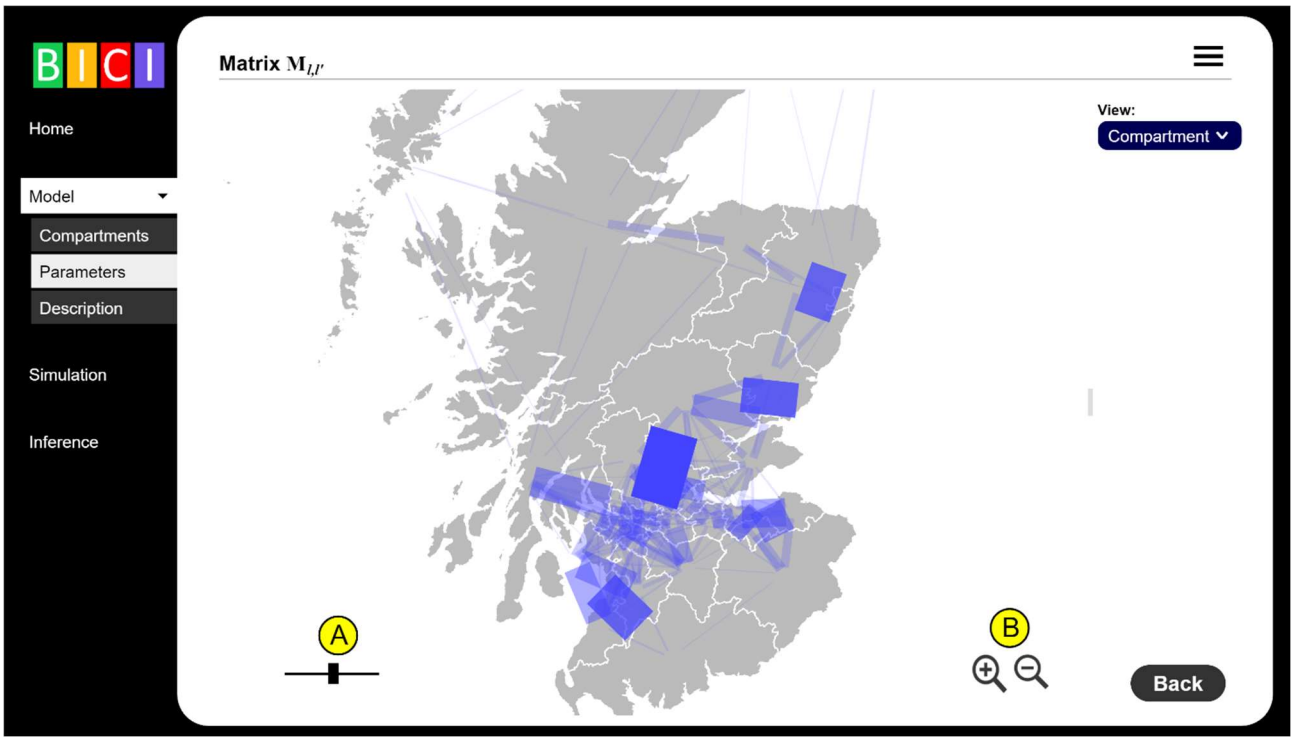


Figure H9 – *Geographical matrix*. When both matrix indices correspond to a geographical classification, the matrix can be plotted on the map. Here the strength of the bars indicates the size of the corresponding matrix elements. A: A slider can be used to vary the size of the bars (to help clarity) , B: The map can be zoomed in or out to see fine details. Time-varying matrices can also be plotted, with animation controls as in Fig. D3.

Appendix I: Population visualisations

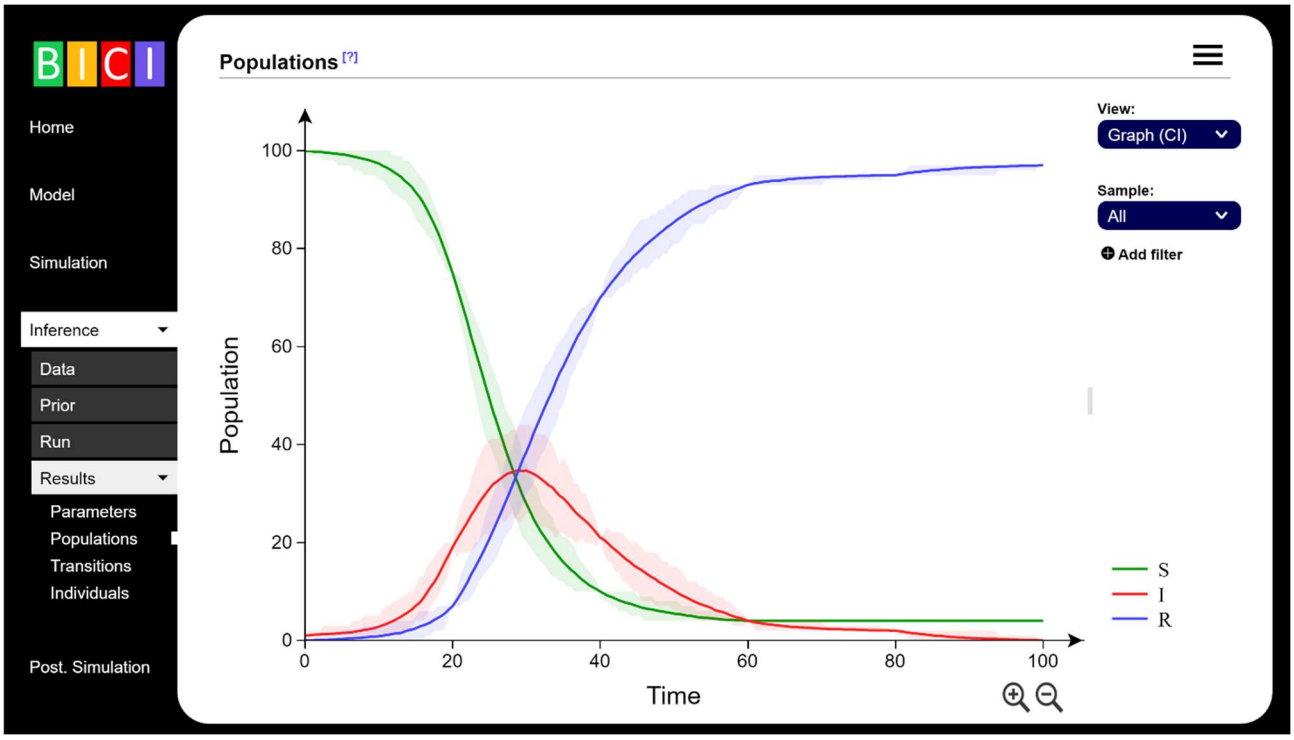


Figure I1 – *Population graph*. Shows the posterior distribution for the populations in different compartments as a function of time (solid lines give posterior means and shading gives 95% credible intervals). Various options on the right-hand menu are described in §4.4.2.

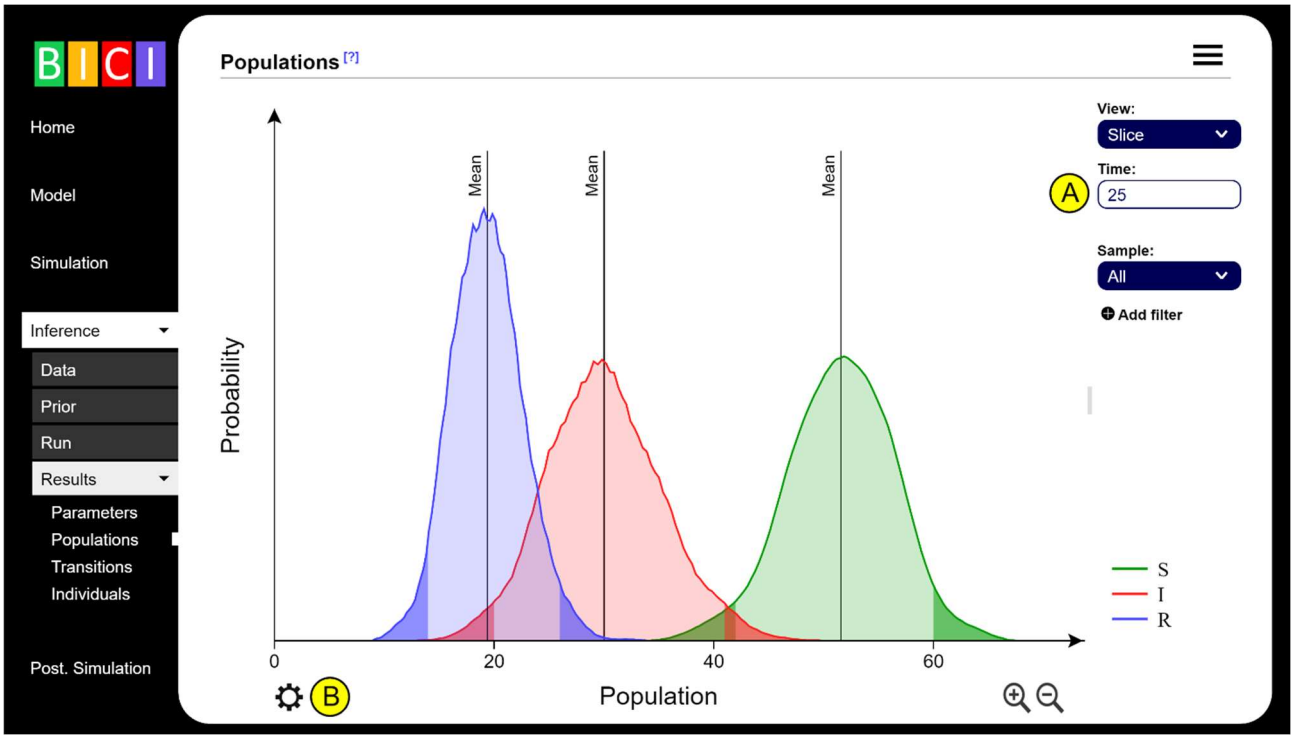


Figure I2 – *Time slice*. Provides the posterior distribution for populations at a specified time slice. A: Select the time, B: Under settings, either kernel density estimation (with a triangular kernel of specified size), or binning (with specified bin number) can be selected. Various options on the right-hand menu are described in §4.4.2.

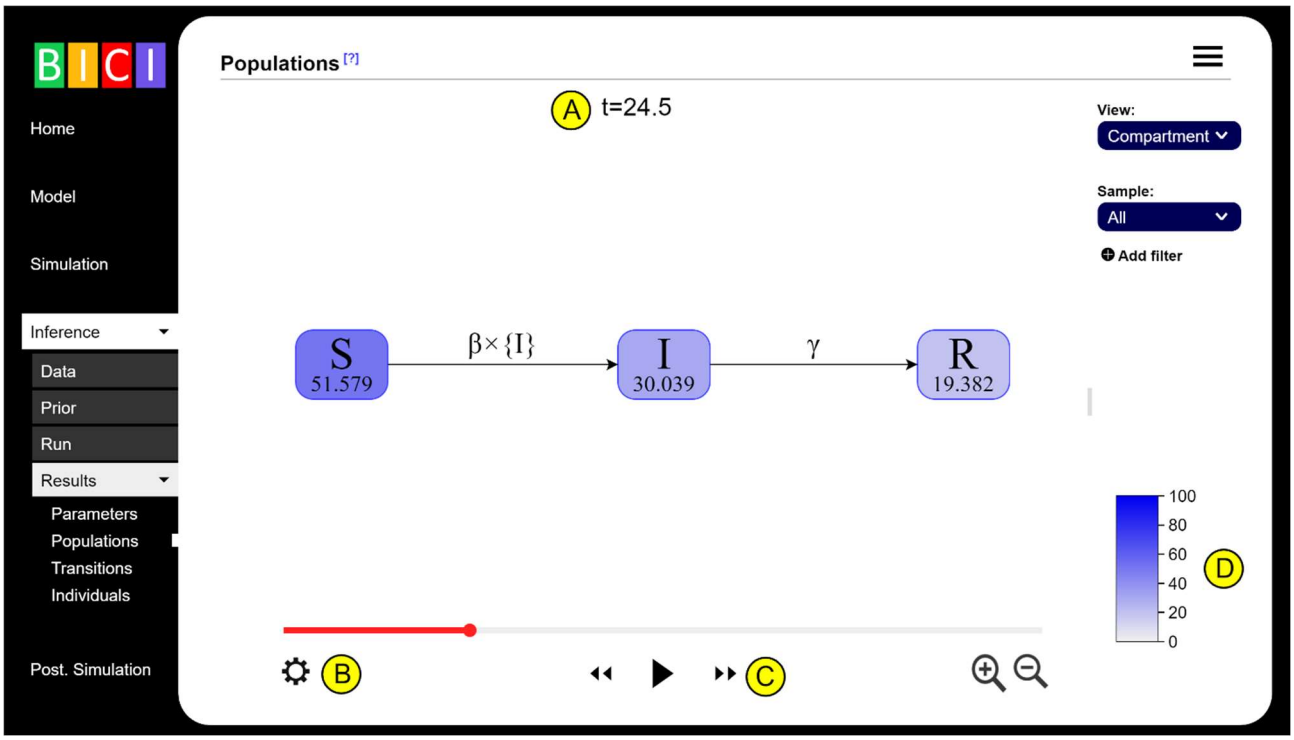


Figure I3 – *Compartment plot*. Shows an animation giving the posterior mean for the compartmental populations as a function of time. A: Shows the current time, B: Settings allow for the animation speed to be altered, C: The animation can be played, or stepped forwards or backwards, one frame at a time, D: Key for colour on compartments. Various options on the right-hand menu are described in §4.4.2.

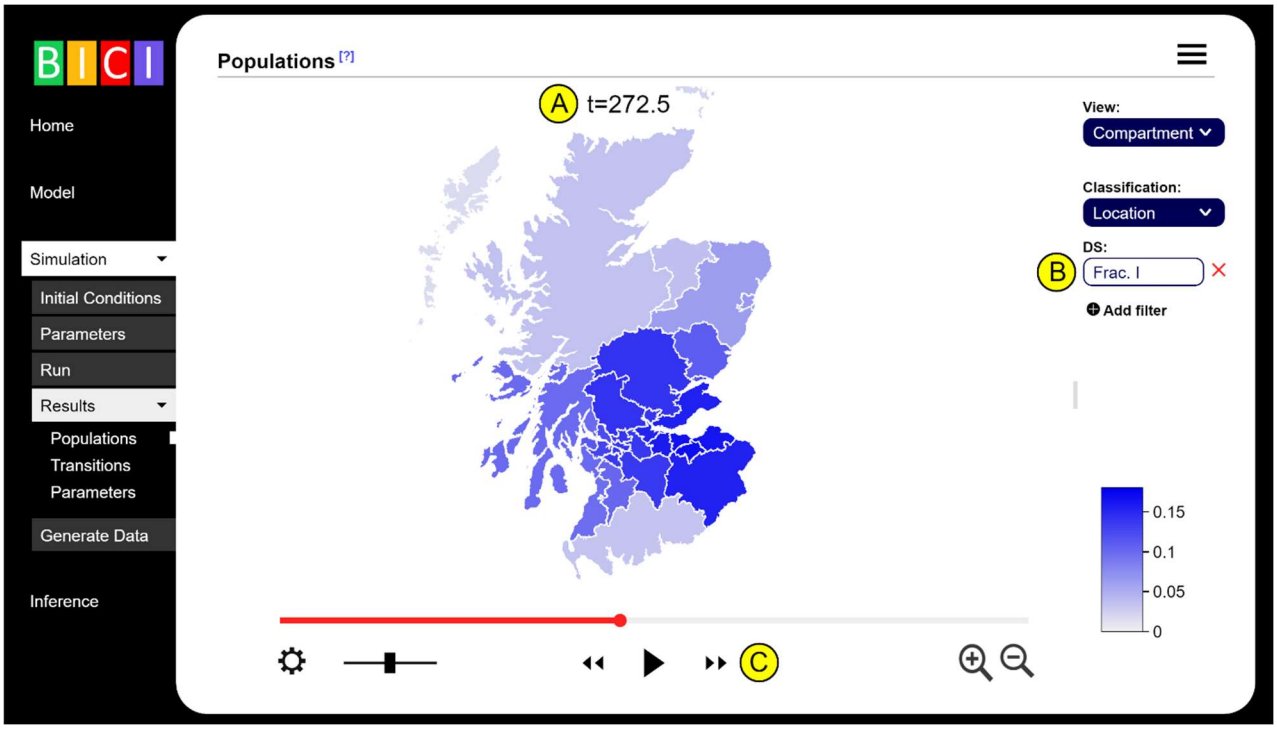


Figure I4 – *Geographical compartment plot*. When compartmental plots are made using geographical coordinates, they can be shown on maps. This shows an animation giving the posterior mean for the compartmental populations as a function of time. A: Shows the current time, B: This filter shows the fraction of infected individuals (*i.e.* the prevalence), C: The animation can be played, or stepped forwards or backwards, one frame at a time. Various options on the right-hand menu are described in §4.4.2.

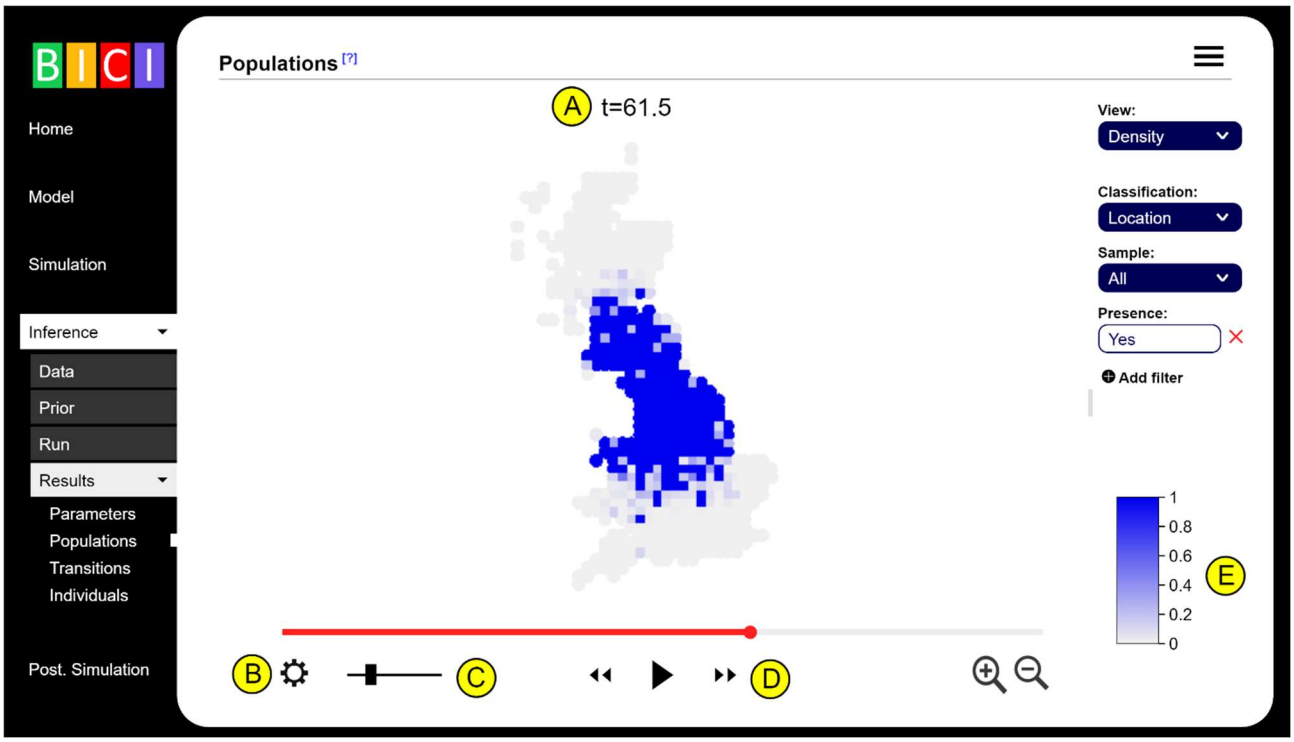


Figure I5 – *Density plot*. A density plot is like a compartment plot, but here circles are used (when overlapping they are clipped using Voronoi tessellation). This type of plot can be good for viewing the spread of disease over geographical points (*e.g.* farms). A: Shows the current time, B: Under settings the animation speed can be changed, C: This slider changes the size of the circles (which can be useful to improve the spatiotemporal visualisation), D: The animation can be played, or stepped forwards or backwards, one frame at a time, E: Key which relates the circle colours to population size. Various options on the right-hand menu are described in §4.4.2.

Appendix J: Transition visualisations

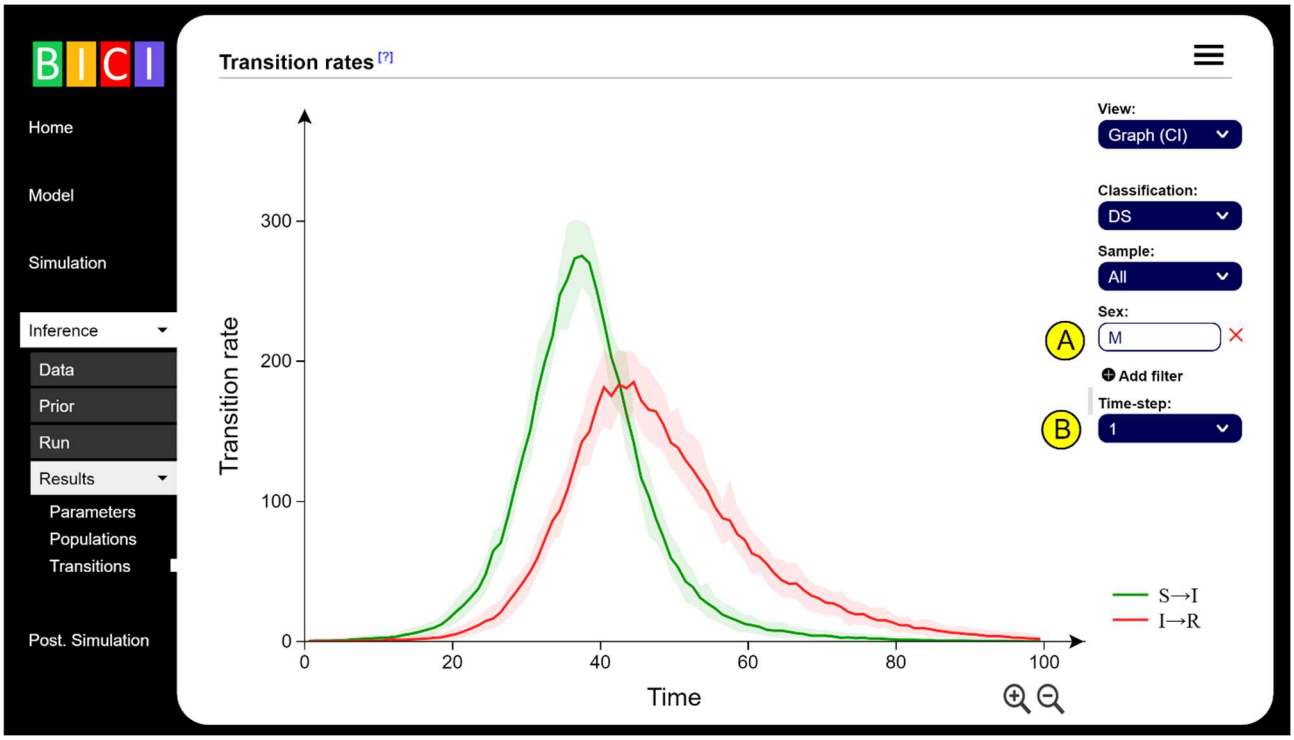


Figure J1 – *Transition rate plot*. This plots transition rates as a function of time (solid lines represent posterior means and shading represents 95% credible intervals). A: Population filters can be applied, in this case specifying males only (see §4.4.2 for details), B: The time-step can be altered (making this larger results in smoother plots, but reduces temporal resolution).

Appendix K: Individual visualisations

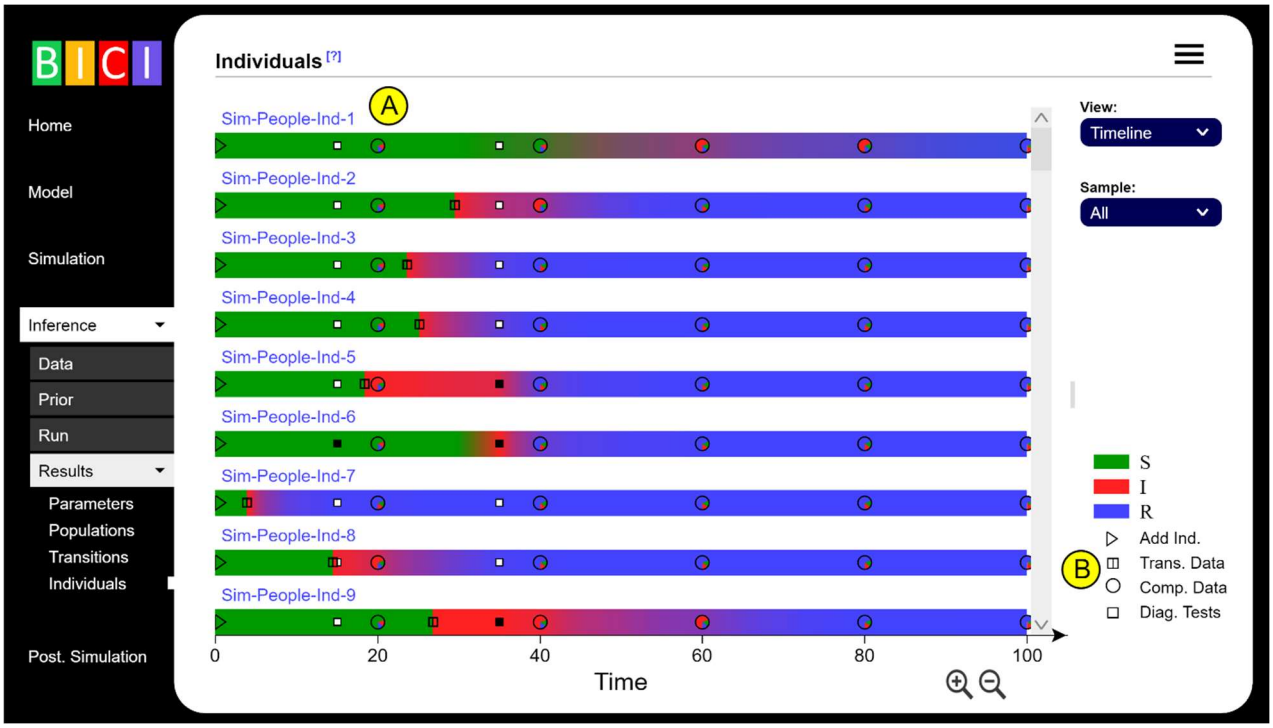


Figure K1 – *Individual timelines*. This summarises individual posterior state information. Each individual in the system has its own timeline. Compartments are colour coded (S=green, I=red and R=blue) according to a key. The gradations in colour reflect the mixing of compartmental colours in proportion to the posterior mean (*e.g.* a strongly red colour indicates that the individual is highly likely to be in the 'I' compartment). A: Timeline for individual with ID "Sim-People-Ind-1", B: Different symbols represent different individual data types: right triangles show the addition of an individual, double rectangles denote transition data, circles represent compartmental data (where the small pie chart indicates the probability of being in the different compartments) and squares represent diagnostic test results (black/white indicating +ve/-ve).

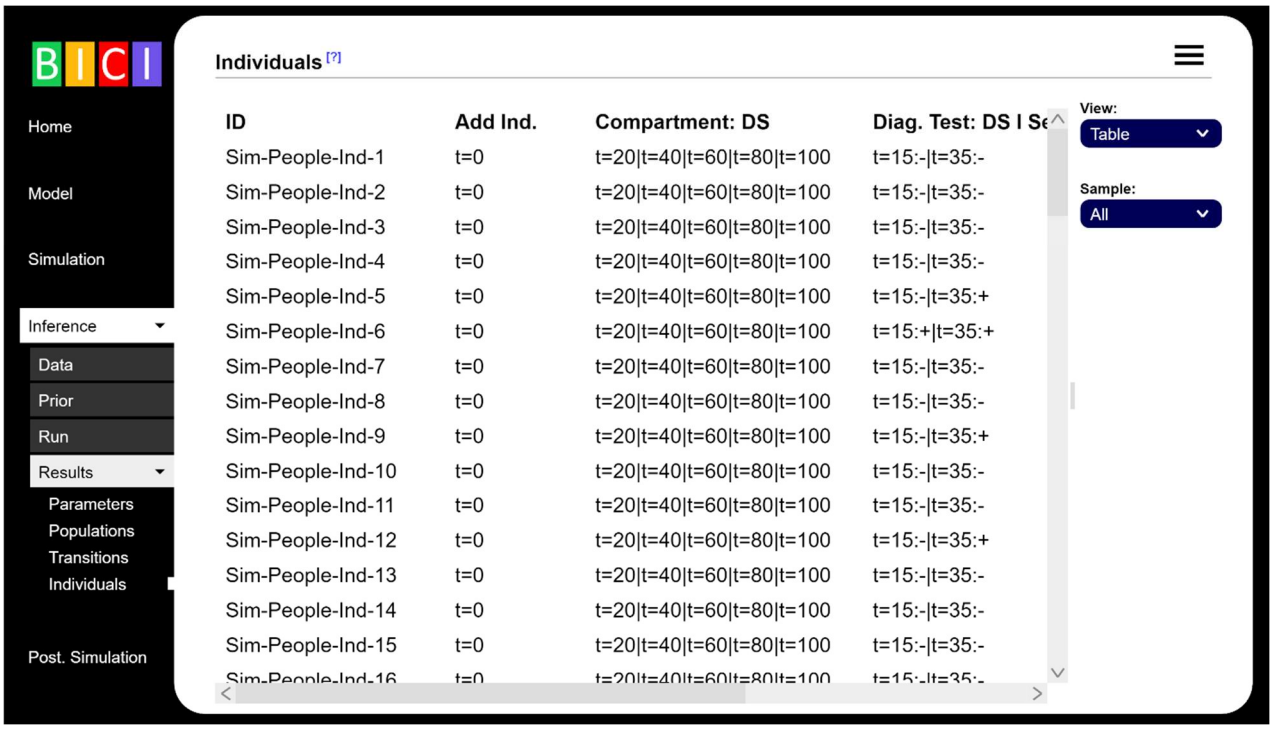


Figure K2 – *Table*. All individual data and properties are collected together into a table.

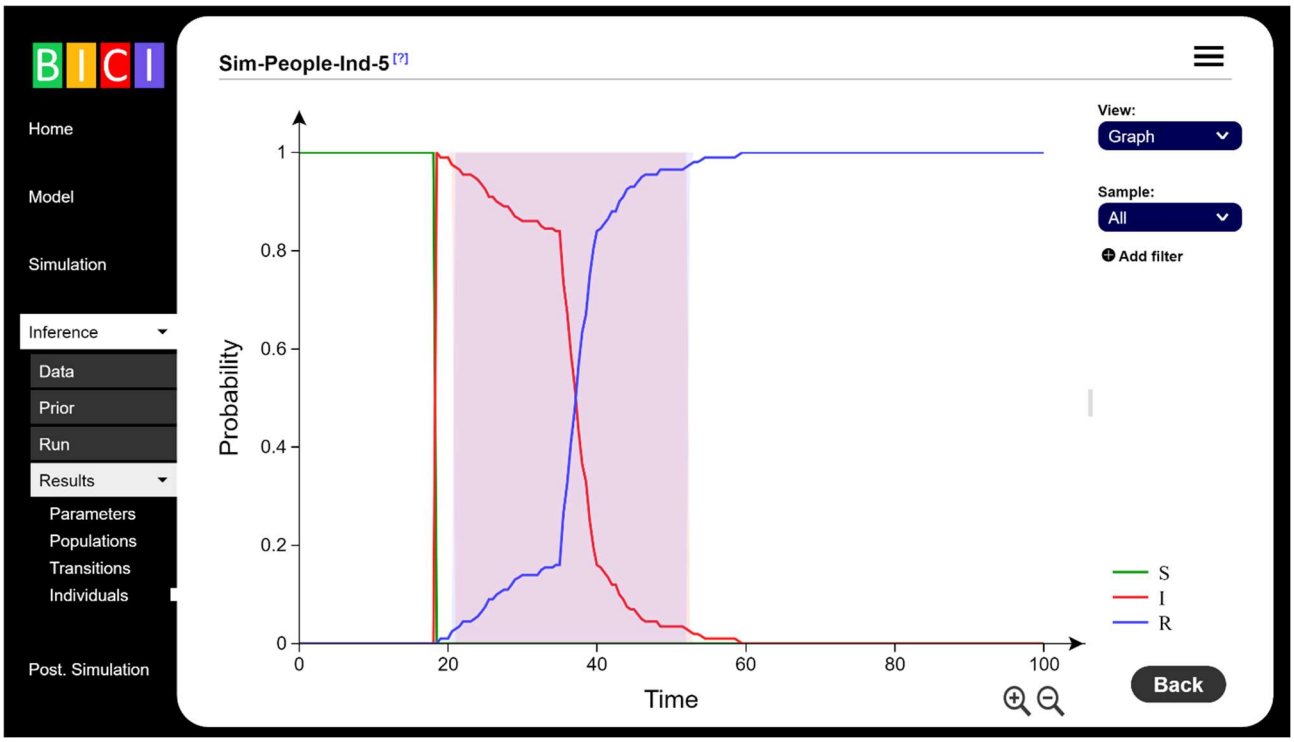


Figure K3 – *Graph of individual compartmental probabilities*. After an individual has been selected (Fig. F1A), this graph shows how the probability the individual is in the different compartments changes as a function of time (solid lines represent posterior means and shading represent 95% credible intervals).

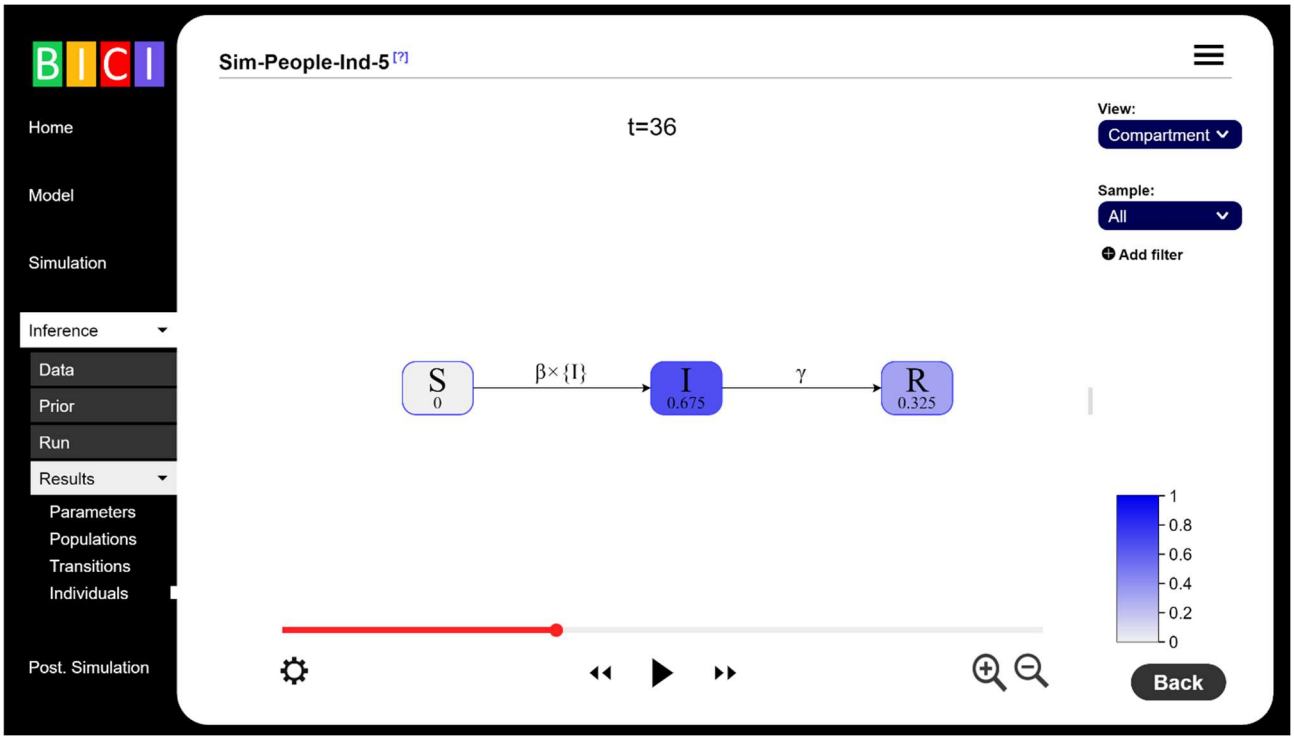


Figure K4 – *Individual compartment probabilities*. After an individual has been selected (Fig. F1A), this animation shows how the mean posterior probability that the individual is in the various compartments changes as a function of time. An analogous "density" plot is also available (see Fig. D5).

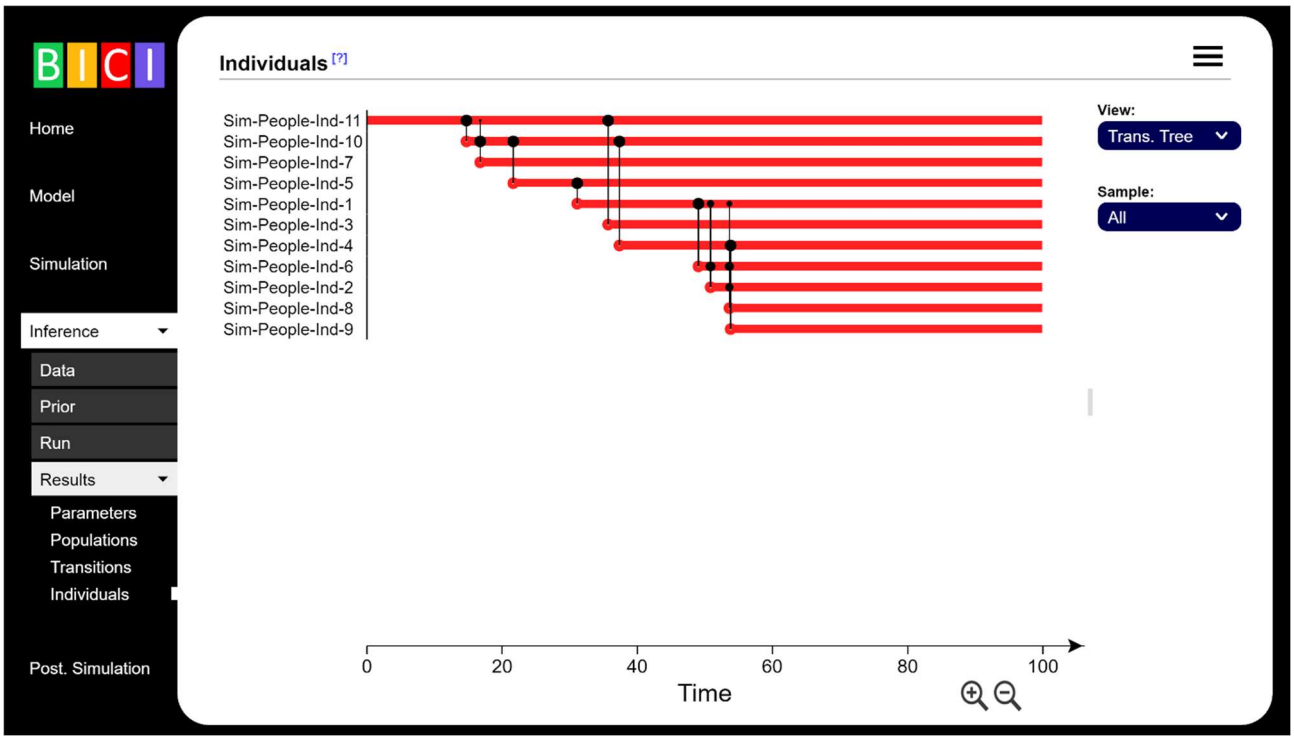


Figure K5 – *Transmission tree*. This shows who infects whom. Each individual has a time line. The red lines indicate the time period over which individual are infected. The black bars indicate the most likely individual and timing that caused the infection (with the size of the black circles indicating certainty).

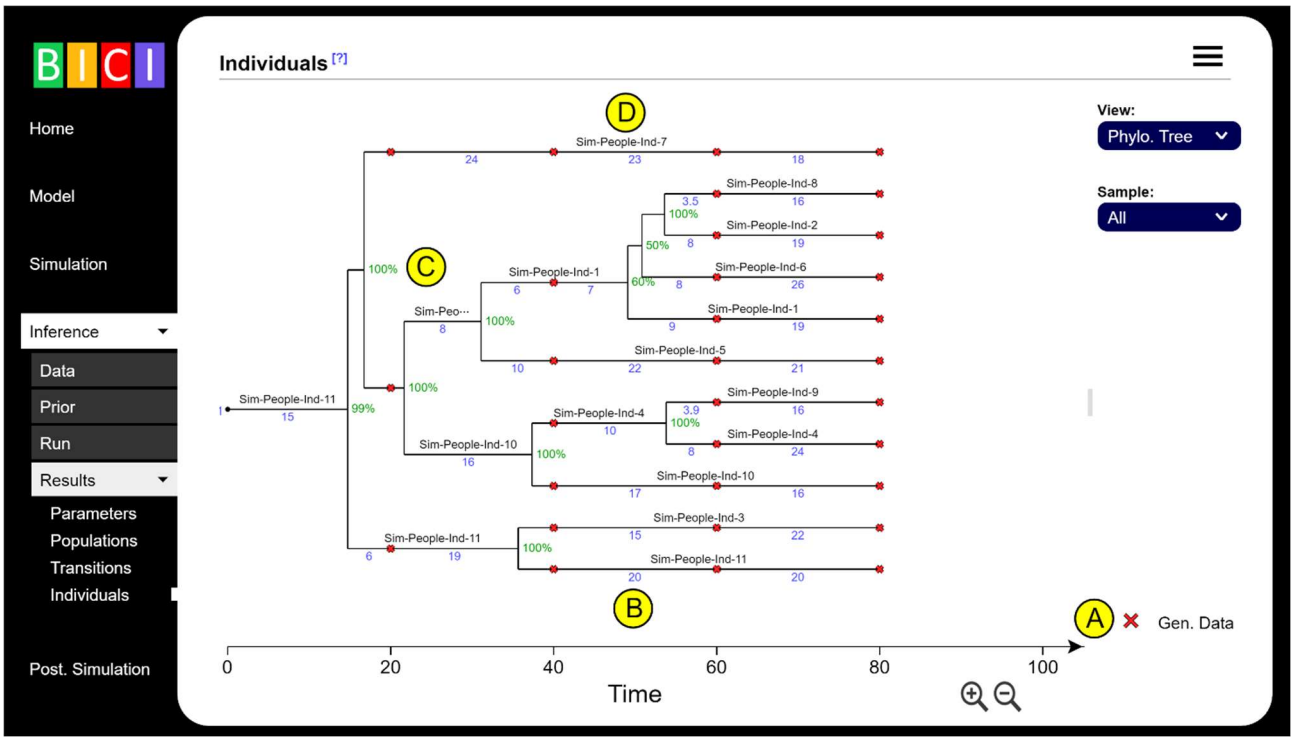


Figure K6 – *Phylogenetic tree*. This shows the relationship between the observed genetic data. A: The red crosses indicate the genetic observations of the pathogen, B: The blue figures represent the mean posterior number of mutations down a line segment, C: Green percentages show the posterior probability that this particular branching point occurs, D: Labels indicate the most likely pathogen host individual.