Chapter 5

Uncertainty Quantification, Model Calibration and Sensitivity

Jakub Bijak and Jason Hilton

Better understanding of the behaviour of agent-based models, aimed at embedding them in the broader, model-based line of scientific enquiry, requires a comprehensive framework for analysing their results. Seeing models as tools for experimenting *in silico*, this chapter discusses the basic tenets and techniques of uncertainty quantification and experimental design, both of which can help shed light on the workings of complex systems embedded in computational models. In particular, we look at: relationships between model inputs and outputs, various types of experimental design, methods of analysis of simulation results, assessment of model uncertainty and sensitivity, which helps identify the parts of the model that matter in the experiments, as well as statistical tools for calibrating models to the available data. We focus on the role of emulators, or meta-models – high-level statistical models approximating the behaviour of the agent-based models under study – and in particular, on Gaussian processes (GPs). The theoretical discussion is illustrated by applications to the Routes and Rumours model of migrant route formation introduced before.

5.1 Bayesian Uncertainty Quantification: Key Principles

Computational simulation models can be conceptualised as tools for carrying out “opaque thought experiments” (Di Paolo et al., 2000), where the links between model specification, inputs and outputs are not obvious. Many different sources of uncertainty contribute to this opaqueness, some of which are related to the uncertain state of the world – the reality being modelled – and our imperfect knowledge about it, while others relate to the different elements of the models. In the context of computational modelling, Kennedy and O’Hagan (2001) proposed a taxonomy of sources of error and uncertainty, the key elements of which encompass: model inadequacy – discrepancy between the model and the reality it represents; uncertainty in observations (including measurement errors); uncertainty related to the unknown model parameters; pre-specified parametric variability, explicitly included in the

model via probability distributions; errors in the computer code; and residual variability, left after accounting for every other source.

The tools of probability and statistics, and in particular Bayesian statistics, offer a natural way of describing these different sources of uncertainty, by expressing every modelled quantity as a random variable with a probability distribution. The mechanism of Bayesian inference, by which the prior quantities (distributions) are combined with the likelihood of the data to yield posterior quantities, helps bring together the different sources of knowledge – data and *prior knowledge*, the latter for example elicited from experts in a given domain.

There is a long history of mutual relationships between Bayesian statistics and social sciences, including demography, dating back to the seminal work of Thomas Bayes and Pierre-Simon de Laplace in the late eighteenth century (Courgeau, 2012, see also Foreword to this book). A thorough introduction to Bayesian statistics is beyond the scope of this book, but more specific details on Bayesian inference and applications in social sciences can be found in some of the excellent textbooks and reference works (Lynch, 2007; Gelman et al., 2013; Bryant & Zhang, 2018), while the use of Bayesian methods in demography was reviewed in Bijak and Bryant (2016).

The Bayesian approach is especially well-suited for carrying out a comprehensive analysis of uncertainty in complex computational models, as it can cover various sources and forms of error in a coherent way, from the estimation of the models, to prediction, and ultimately to offering tools for supporting decision making under uncertainty. In this way, Bayesian inference offers an explicit, coherent description of uncertainty at various levels of analysis (parameters, models, decisions), allows the expert judgement to play an important role, especially given deficiencies of data (which are commonplace in such areas as migration), and can potentially offer more realistic assessment of uncertainty than traditional methods (Bijak, 2010).

Uncertainty quantification (UQ) as a research area looking into uncertainty and inference in large, and possibly analytically intractable, computational models, spanning statistics, applied mathematics and computing, has seen rapid development since the early twenty-first century (O'Hagan, 2013; Smith, 2013; Ghanem et al., 2019). The two key aspects of UQ include propagating the uncertainty through the model and learning about model parameters from the data (calibration), with the ultimate aim of quantifying and ideally reducing the uncertainty of model predictions (*idem*). The rapid development of UQ as a separate area of research, with distinct methodology, has been primarily motivated by the increase in the number and importance of studies involving large-scale computational models, mainly in physical and engineering applications, from astronomy, to weather and climate, biology, hydrology, aeronautics, geology and nuclear fusion (Smith, 2013), although with social science applications lagging behind. A recent overview of UQ was offered by Smith (2013), and a selection of specific topics were given detailed treatment in the living reference collection of Ghanem et al. (2019). For the reasons mentioned before, Bayesian methods, with their coherent probabilistic language for describing all unknowns, offer natural tools for UQ applications.

The main principles of UQ include a comprehensive description of different sources of uncertainty (error) in computational models of the complex systems

under study, and inference about the properties of these systems on that basis. To do that, it relies on specific methods from other areas of statistics, mathematics and computing, which are tailored to the UQ problems. These methods, to a large extent, rely on the use of *meta-models* (or *emulators*, sometimes also referred to as surrogate models) to approximate the dynamics of the complex computational models, and facilitate other uses. Specific methods that have an important place in UQ include uncertainty analysis, which looks at how uncertainty is propagated through the model, and sensitivity analysis, which aims to assess which elements of the model and, in particular, which parameters matter for the model outputs (Oakley & O'Hagan, 2002). Besides, for models with predictive ambitions, methods for calibrating them to the observed data become of crucial importance (Kennedy & O'Hagan, 2001). We discuss these different groups of methods in more detail in the remainder of this chapter, starting from a general introduction to the area of statistical experimental design, which is underpinning the construction and calibration of meta-models, and therefore provides foundations for many of the UQ tools and their applications.

5.2 Preliminaries of Statistical Experimental Design

The use of tools of statistical experimental design in the analysis of the results of agent-based models starts from the premise that agent-based models, no matter how opaque, are indeed experiments. By running the model at different parameter values and with different settings – that is, experimenting by repeated execution of the model *in silico* (Epstein & Axtell, 1996) – we learn about the behaviour of the model, and hopefully the underlying system, more than would be possible otherwise. This is especially important given the sometimes very complex, non-transparent and analytically intractable nature of many computational simulations.

Throughout this chapter, we will define an experiment as a process of measuring a “stochastic response corresponding to a set of ... input variables” (Santner et al., 2003, p. 2). A computer experiment is a special case, based on a mathematical theory, implemented by using numerical methods with appropriate computer hardware and software (*idem*). Potential advantages of computer experiments include their built-in features, such as replicability, relatively high speed and low cost, as well as their ability to analyse large-scale complex systems. Whereas the quality standards of natural experiments are primarily linked to the questions of randomisation (as in randomised control trials), blocking of similar objects to ensure homogeneity, and replication of experimental conditions, computer experiments typically rely on deterministic or stochastic simulations, and require transparency and thorough documentation as minimum quality standards (*idem*).

Computer experiments also differ from traditional, largely natural experiments thanks to their wider applicability, also to social and policy questions, with different ethical implications than experiments requiring direct human participation. In some social contexts, other experiments would not be possible or ethical. For example,

analysing optimal ways of evacuating people facing immediate danger (such as fire or flood), very important for tailoring operational response, cannot involve live experiments in actual dangerous conditions. In such cases, computer experiments can provide invaluable insights into the underlying processes, possibly coupled with ethically sound natural experiments carried out in safe conditions, for example on the ways large groups of people navigate unknown landscapes.

To make the most of the computer experiments, their appropriate planning and design becomes of key importance. To maximise our information gains from experimentation, which typically comes at a considerable computational cost (as measured in computing time), we need to know at which parameter values and with which settings the models need to be run. The modern statistical theory and practice of experimental design dates back to the agricultural work of Sir Ronald Fisher (1926), with the methodological foundations fully laid out, for example, in the much-cited works of Fisher (1935/1958) and Cox (1958/1992). Since then, the design of experiments has been the subject of many refinements and extensions, with applications specifically relevant for analysing computer models discussed in Santner et al. (2003) and Fang et al. (2006), among others.

The key objectives of the statistical design of experiments are to help understand the relationship between the inputs and the outcome (response), and to maximise information gain from the experiments – or to minimise the error – under computational constraints, such as time and cost of conducting the experiments. The additional objectives may include aiding the analytical aims listed before, such as the uncertainty or sensitivity analysis, or model-based prediction.

As for the terminology, throughout this chapter we use the following definitions, based on the established literature conventions. Most of these definitions follow the conventions presented in the *Managing Uncertainty in Complex Models* online compendium (MUCM, 2021).

Model (simulator) “A representation of some real-world system, usually implemented as a computer program” (MUCM, 2021), which is transforming inputs into outputs;

Factor (input) “A controllable variable of interest” (Fang et al., 2006, p. 4), which can include model parameters or other characteristics of model specification.

Response (output) A variable representing “specific properties of the real system” (Fang et al., 2006, p. 4), which are of interest to the analyst. The output is a result of an individual run (implementation) of a model for a given set of inputs.

Calibration The analytical process of “adjusting the inputs so as to make the simulator predict as closely as possible the actual observation points” (MUCM, 2021);

Calibration parameter “An input which has ... a single best value” with respect to the match between the model output and the data (reality), and can be therefore used for calibration (MUCM, 2021);

Model discrepancy (inadequacy) The residual difference between the observed reality and the output calibrated at the best inputs (calibration parameters);

Meta-model (emulator, surrogate) A statistical or mathematical model of the underlying complex computational model. In this chapter, we will mainly look at statistical emulators.

Fig. 5.1 Concepts of the model discrepancy (left), design (middle) and training sample (right). For the discrepancy example, the real process (solid line) is $f(x) = 1.2 \sin(8\pi x)$, and the model (dashed line) is a polynomial of order 6, fitted by using ordinary least squares. The calibration parameters are then the coefficients of the polynomial, and the model discrepancy is the difference between the values of the two functions. (Source: own elaboration)

Design “A choice of the set of points in the space of simulator inputs at which the simulator is run” (MUCM, 2021), and which then serve as the basis for model analysis;

Training sample Data comprising inputs from the design space, as well as the related outputs, which are used to build and calibrate an emulator for subsequent use in the analysis.

The diagrams in Fig. 5.1 illustrate the concepts of model discrepancy, design and training sample.

There are different types of design spaces, which are briefly presented here following their standard description in the selected reference works (Cox, 1958/1992; Santner et al., 2003; Fang et al., 2006). To start with, a **factorial design** is based on combinations of design points at different levels of various inputs, which in practice means being a subset of a hyper-grid in the full parameter space, conventionally with equidistant spacing between the grid points for continuous variables. As a special case, the **full factorial design** includes *all* combinations of *all* possible levels of *all* inputs, whereas a **fractional factorial design** can be any subset of the full design. Due to practical considerations, and the ‘combinatorial explosion’ of the number of possible design points with the increasing number of parameters, limiting the analysis to a fractional factorial design, for the sake of efficiency, is a pragmatic necessity.

There are many ways in which fractional factorial designs can be constructed. One option involves random design, with design points randomly selected from the full hyper-grid, e.g. by using simple random sampling, or – more efficiently – stratified sampling, with the hyper-grid divided into several strata in order to ensure good coverage of different parts of the parameter space. An extension of the stratified design is the Latin Hypercube design – a multidimensional generalisation of a two-dimensional idea of a Latin Square, where only one item can be sampled from each row and each column, similarly to a Sudoku puzzle. In the multidimensional case, only one item can be sampled for each level in every dimension; that is, for every input (*idem*).

Fig. 5.2 Examples of a full factorial (left), fractional factorial (middle), and a space-filling Latin Hypercube design (right). (Source: own elaboration)

More formally, with a discrete Latin Hypercube design we ideally want to cover the whole range of the distribution of each of the K input variables, X_i . For each i , let this input range be divided into N equal parts (bins), from which N elements satisfying the Latin Hypercube rule can be sampled. This can be done in $[N^K (N-1)^K \dots 1^K] / [N(N-1) \dots 1] = (N!)^{K-1}$ different ways. Among those, some designs can be space filling, with points spread out more evenly in the multidimensional space, while some others are non-space filling, leaving large ‘gaps’ without sampling points, which is undesirable. In practice, the available algorithms try ensuring that the design is as much space filling as possible, for example by maximising the minimum distances between the design points, or minimising correlations between factors (Ranjan & Spencer, 2014). Examples of a full factorial, fractional factorial, and a space-filling Latin Hypercube design spaces for a 6×6 grid are shown in Fig. 5.2.

Generally, Latin Hypercube samples have desirable statistical properties, and are considered more efficient than both random and stratified sampling (see the examples given by McKay et al., 1979). One alternative approach involves model-based design, which requires a model for the results that we expect to observe based on any design – for example an emulator – as well as an optimality criterion, such as minimising the variance, maximising the information content, or optimising a certain decision based on the design, in the presence of some loss (cost) function. The optimal model-based design is then an outcome of optimising the criterion over the design space, and a typical example involves design that will minimise the variance of an emulator built for a given model.

If the parameter space is high-dimensional, it is advisable to reduce the dimensionality first, to limit the analysis to those parameters that matter the most for a given output. This can be achieved by carrying out pre-screening, or sequential design, based on sparse fractional factorial principles, which date back to the work of Davies and Hay (1950). Among the different methods that have been proposed for that purpose, Definitive Screening Design (Jones & Nachtsheim, 2011, 2013) is relatively parsimonious, and yet allows for identifying the impact of the main effects of the parameters in question, as well as their second-order interactions.

Fig. 5.3 Visualisation of a transposed Definite Screening Design matrix D' for 17 parameters. Black squares correspond to high parameter values (+1), white to low ones (-1), and grey to middle ones (0). (Source: own elaboration)

The Definitive Screening Design approach is based on so-called conference matrices $C_{m \times m}$, such that $1/(m-1) C'C = I_{m \times m}$, where m is either the number of parameters (if m is even), or the number of parameters plus 1 (if m is odd). The elements of matrix C can take three values: +1 for the ‘high’ values of the respective parameters, 0 for the ‘middle’ values, and -1 for the ‘low’ values, where the specifics are set by the analyst after looking at the possible range of each parameter. The design matrix D is then obtained by stacking the matrices C , $-C$ and a vector of middle values, 0 , so that $D' = [C', -C', 0']'$ (Jones & Nachtsheim, 2011, 2013). The rows of matrix D' represent parameters (if m is odd, the last row can be omitted), and the columns represent the design points, at which the pre-screening experiments are to be run: $2m + 1$ if m is even, and $2m + 3$ if m is odd. An example of a design matrix D' for $m = 17$ parameters, implying 37 design points, is illustrated in Fig. 5.3.

Once the model is run, either a descriptive exploration of the output, or a formal sensitivity analysis (see Sect. 5.4) can indicate which parameters can be dropped without much information loss. In Box 5.1, we present an illustration of the proposed approach for the Routes and Rumours migration model, which was introduced in Chap. 3, with some detailed results reported in Appendix C.

Other methods that can be used for pre-screening of the model parameter space include Automatic Relevance Determination (ARD), and Sparse Bayesian Learning (SBL), dating back to the work of MacKay (1992), which both use Bayesian inference to reduce the dimensionality of the parameter space by ‘pruning’ the less relevant dimensions (for an overview, see e.g. Wipf & Nagarajan, 2008). From the statistical side, these methods link with Bayesian model selection (Hoeting et al., 1999) and the Occam’s razor principle, which favours simpler models (in this case, models with fewer parameters) over more complex ones. From the machine

Box 5.1: Designing Experiments on the Routes and Rumours Model

This running example illustrates the process of experimental design and analysis for the model of migrant routes and information exchange introduced in Chap. 3. In this case, the pre-screening was run on $m = 17$ parameters: six related to information exchange and establishing or retaining contacts between the agents; four related to the way in which the agents explore their environment, with focus on the speed and efficiency; four describing the quality of the routes, resources and the environment; and three related to the resource economy: resources and costs.

The Definitive Screening Design was applied to the initial 17 parameters, with 37 design points as shown in Fig. 5.3, with the low, medium and high values corresponding to $\frac{1}{4}$, $\frac{1}{2}$ and $\frac{3}{4}$ of the respective parameter ranges. At these points, four model outputs were generated: *mean_freq_plan*, related to agent behaviour, describing the proportion of time the agents were following their route plan; *stddev_link_c*, describing route concentration, measuring the standard deviation of the number of visits over all links; *corr_opt_links*, linked to route optimality, operationalised as the correlation of the number of passages over links with the optimal scenario; and *prop_stddev*, measuring replicability, here approximated by the standard deviation of traffic *between* replicate runs (see also Bijak et al., 2020). For the first three outputs, 10 samples were taken at each point, to allow for the cross-replication error in the computer code, while the fourth one already summarised cross-replicate information.

The results of the model were analysed by using Gaussian process emulators fitted in the GEM-SA package and used for conducting a preliminary sensitivity analysis (Kennedy & Petropoulos, 2016, see also Sects. 5.3 and 5.4). Across the four outputs, five parameters related to information exchange proved to be of primary importance: the probabilities of losing a contact (*p_drop_contact*), communicating with local agents (*p_info_mingle*), communicating with contacts (*p_info_contacts*), and exchanging information through communication (*p_transfer_info*), as well as the information noise (*error*). The sensitivity analysis indicated that these five parameters were jointly responsible for explaining between 30% and 83% of the variation of the four outputs, and almost universally included the top three most influential parameters for each output. For further experiments, two parameters related to exploration were also manually included, to make sure that the role of this part of the model was not overlooked. These were the speed of learning about the environment (*speed_expl*), and probability of finding routes and connecting links during the local exploration (*p_find*). Detailed results in terms of shares of variances attributed to individual inputs are reported in Appendix C.

The results proved largely robust to changes in the random seed, especially when a separate variance term for the error in computer code (the ‘nugget’ variance) was included, and also when comparing them with the outcome of a standard ANOVA procedure. For the further steps of the analysis, a Latin Hypercube sample design was generated in GEM-SA, with $N = 65$ design points, and six replicates of the model run at each point, so with 390 samples in total. This sample was used to build and test emulators and carry out uncertainty and sensitivity analysis, as discussed in the next section.

learning side, these approaches also have common features with support vector machines (Tipping, 2001). As the ARD and SBL methods are quite involved, we do not discuss them here in more detail, but a fuller treatment of some of the related approaches can be found, for example, in Neal (1996).

5.3 Analysis of Experiments: Response Surfaces and Meta-Modelling

There are several ways in which the results of complex computational experiments can be analysed. The two main types of analysis, linking to different research objectives, include *explanation* of the behaviour of the systems being modelled, as well as the *prediction* of this behaviour outside of the set of observed data points. In this chapter, broadly following the framework of Kennedy and O'Hagan (2001), we look specifically at four types of explanations:

- Response of the model output to changes in inputs, both descriptive and model-based.
- Sensitivity analysis, aimed at identifying the inputs which influence the changes in output.
- Uncertainty analysis, describing the output uncertainty induced by the uncertain inputs.
- Calibration, aimed at identifying a combination of inputs, for which the model fits the observed data best, by optimising a set of calibration parameters (see Sect. 5.2).

Notably, Kleijnen (1995) argued that these types of analysis (or equivalent ones) also serve an internal modelling purpose, which is model *validation*, here understood as ensuring “a satisfactory range of accuracy consistent with the intended application of the model” (Sargent, 2013: 12). This is an additional model quality requirement beyond a pure code *verification*, which is aimed at ensuring that “the computer program of the computerized model and its implementation are correct” (*idem*). In other words, carrying out different types of explanatory analysis, ideally together, helps validate the model internally – in terms of inputs and outputs – as well as externally, in relation to the data. Different aspects of model validation are reviewed in a comprehensive paper by Sargent (2013).

At the same time, throughout this book we interpret *prediction* as a type of analysis involving both *interpolation* between the observed sample points, as well as *extrapolation* beyond the domain delimited by the training sample. Extrapolation comes with obvious caveats related to going beyond the range of training data, especially in a multidimensional input space. Predictions can also serve the purpose of model validation, both out-of-sample, by assessing model errors on new data points, outside of the training sample, as well as in-sample (cross-validation), on the same

Fig. 5.4 Examples of piecewise-linear response surfaces: a 3D graph (left) and contour plot (right). (Source: own elaboration)

data points, by using such well-known statistical techniques as leave-one-out, jack-knife, or bootstrap.

In all these cases, mainly because of computational constraints – chiefly the time it takes the complex computer models to run – it is much easier to carry out the explanatory and predictive analysis based on the surrogate meta-models. To that end, we begin the overview of the methods of analysis by discussing response surfaces and other meta-models in this section, before moving to the uncertainty and sensitivity analysis in Sect. 5.4, and calibration in Sect. 5.5.

The first step in analysing the relationships between model inputs and outputs is a simple, usually graphical description of a *response surface*, which shows how model output (response) varies with changes in the input parameters (for a stylised example, see Fig. 5.4). This is useful mainly as a first approximation of the underlying relationships, although even at this stage the description can be formalised, for example by using a regression meta-model, either parametric or non-parametric. Such a simple meta-model can be estimated from the data and allows the inclusion of some – although not all – measures of error and uncertainty of estimation, mainly those related to the random term and parameter estimates. The typical choices for regression-based approximations of the response surfaces include models including just the main (first-order) effects for the individual parameters, as well as those additionally involving quadratic effects, and possible interaction terms (Kleijnen, 1995). Other options include local regression models and spline-based non-parametric approaches.

The uses of emulators based on Gaussian processes date back to approaches that later became known as Kriging, named after South African geostatistician, Danie G Krige, who developed them in early 1950s¹ (Cressie, 1990). The more recent developments, specifically tailored for the meta-analysis of complex computational models, are largely rooted in the methodology proposed in the seminal papers of

¹It is worth noting that, according to Cressie (1990), similar methods were independently proposed already in the 1940s by Herman Wold, Andrey Nikolaevich Kolmogorov and Norbert Wiener.

Kennedy and O'Hagan (2001) and Oakley and O'Hagan (2002), presenting the construction and estimation of Bayesian GP emulators.

The basic description of the GP emulation approach, presented here after Kennedy and O'Hagan (2001, 431–434), is as follows. Let the (multidimensional) model inputs $\mathbf{x}$ from the input (parameter) space $\mathbf{X}$, $\mathbf{x} \in \mathbf{X}$, be mapped onto a one-dimensional output $y \in \mathbf{Y}$, by the means of a function f , such that $y = f(\mathbf{x})$. The function f follows a GP distribution, if “for every $n = 1, 2, 3, \dots$, the joint distribution of $f(\mathbf{x}_1), \dots, f(\mathbf{x}_n)$ is multivariate normal for all $\mathbf{x}_1, \dots, \mathbf{x}_n \in \mathbf{X}$ ” (*idem*: 432). This distribution has a mean m , typically operationalised as a linear regression function of inputs or their transformations $\mathbf{h}(\cdot)$, such that $m(\mathbf{x}) = \mathbf{h}(\mathbf{x})' \boldsymbol{\beta}$, with some regression hyperparameters $\boldsymbol{\beta}$. The GP covariance function includes a common variance term across all inputs, σ^2 , as well as a non-negative definite correlation matrix between inputs, $\mathbf{c}(\cdot, \cdot)$. The GP model can be therefore formally written as:

$$f(\cdot) | \boldsymbol{\beta}, \sigma^2, \mathbf{R} \sim \text{MVN}(m(\cdot); \sigma^2 \mathbf{c}(\cdot, \cdot)) \quad (5.1)$$

The correlation matrix $\mathbf{c}(\cdot, \cdot)$ can be parameterised, for example, based on the distances between the input points, with a common choice of $\mathbf{c}(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{c}(\mathbf{x}_1 - \mathbf{x}_2) = \exp(-(\mathbf{x}_1 - \mathbf{x}_2)' \mathbf{R} (\mathbf{x}_1 - \mathbf{x}_2))$, with a *roughness matrix* $\mathbf{R} = \text{diag}(r_1, \dots, r_n)$, indicating the strength of response of the emulator to particular inputs. To reflect the uncertainty of the computer code, the matrix $\mathbf{c}(\cdot, \cdot)$ can additionally include a separate variance term, called a *nugget*. Kennedy and O'Hagan (2001) discuss in more detail different options of model parameterisation, choices of priors for model parameters, as well as the derivation of the joint posterior, which then serves to calibrate the model given the data. We come back to some of these properties in Sect. 5.5, devoted to model calibration.

In addition to the basic approach presented above, many extensions and generalisations have been developed as well. One such extension concerns GP meta-models with heteroskedastic covariance matrices, allowing emulator variance to differ across the parameter space. This is especially important in the presence of phase transitions in the model domain, whereby model behaviour can be different, depending on the parameter combinations. This property can be modelled for example by fitting two GPs at the same time: one for the mean, and one for the (log) variance of the output of interest. Examples of such models can be found in Kersting et al. (2007) and Hilton (2017), while the underpinning design principles are discussed in more detail in Tack et al. (2002).

Another extension concerns multidimensional outputs, where we need to look at several output variables at the same time, but cannot assume independence between them. Among the ideas that were proposed to tackle that, there are natural generalisations, such as the use of multivariate emulators, notably multivariate GPs (e.g. Fricker et al., 2013). Alternative approaches include dimensionality reduction of the output, for example through carrying out the Principal Components Analysis (PCA), producing orthogonal transformations of the initial output, or Independent Component Analysis (ICA), producing statistically independent transformations

(Boukouvalas & Cornford, 2008). One of their generalisations involves methods like Gaussian Process Latent Variable Models, which use GPs to flexibly map the latent space of orthogonal output factors onto the space of observed data (*idem*).

Given that GP emulators offer a very convenient way of describing complex models and their various features, including response surfaces, uncertainty and sensitivity, they have recently become a default approach for carrying out a meta-analysis of complex computational models. Still, the advances in machine learning and increase of computational power have led to the development of meta-modelling methods based on such algorithms, as classification and regression trees (CART), random forests, neural networks, or support vector machines (for a review, see Angione et al., 2020). Such methods can perform more efficiently than GPs in computational terms and accuracy of estimation (*idem*), although at the price of losing analytical transparency, which is an important advantage of GP emulators. In other words, there appear to be some trade-offs between different meta-models in terms of their computational and statistical efficiency on the one hand, and interpretability and transparency on the other. The choice of a meta-model for analysis in a given application needs therefore to correspond to specific research needs and constraints. Box 5.2 below continues with the example of a migration route model introduced in Chap. 4, where a GP emulator is fitted to the model inputs and outputs, with further details offered in Appendix C.

Box 5.2: Gaussian Process Emulator Construction for the Routes and Rumours Model

The design space with seven parameters of interest, described in Box 5.1 was used to train and fit a set of four GP emulators, one for each output. The emulation was done twice, assuming that the parameters are either uniformly or normally distributed. The emulators for all four output variables (*mean_freq_plan*, *std_link_c*, *corr_opt_links* and *prop_std*) additionally included code uncertainty, described by the ‘nugget’ variance term. The fitting was done in GEM-SA (Kennedy & Petropoulos, 2016). In terms of the quality of fit, the root mean square standardised errors (RMSSE) were found to be in the range between 1.59 for *mean_freq_plan* and 1.95 for *std_link_c*, based on a leave-20%-out cross-validation exercise, which, compared with the ideal value of 1, indicated a reasonable fit quality. Figure 5.5 shows an example analysis of a response surface and its error for one selected output, *mean_freq_plan*, and two inputs, *p_transfer_info* and *p_info_contacts*, based on the fitted emulator. Similar figures for the other outputs are included in Appendix C. For this piece of analysis, all the input and output variables have been standardised.

Fig. 5.5 Estimated response surface of the proportion of time the agents follow a plan vs two input parameters, probabilities of information transfer and of communication with contacts: mean proportion (top) and its standard deviation (bottom). (Source: own elaboration)

5.4 Uncertainty and Sensitivity Analysis

Once fitted, emulators can serve a range of analytical purposes. The most immediate ones consider the impact of various model inputs on the output (response). Questions concerning the uncertainty of the output and its susceptibility to the changes in inputs are common. To address these questions, *uncertainty analysis* looks at how much error gets propagated from the model inputs into the output, and *sensitivity analysis* deals with how changes in individual inputs and their different combinations affect the response variable.

Of the two types of analysis, uncertainty analysis is more straightforward, especially when it is based on a fitted emulator such as a GP (5.1), or another meta-model. Here, establishing the output uncertainty typically requires simulating from the assumed distributions for the inputs and from posterior distributions of the emulator parameters, which then get propagated into the output, allowing a Monte Carlo-type assessment of the resulting uncertainty. For simpler models, it may be also possible to derive the output uncertainty distributions analytically.

On the other hand, the sensitivity analysis involves several options, which need to be considered by the analyst to ascertain the relative influence of input variables. Specifically for agent-based models, ten Broeke et al. (2016) discussed three lines of enquiry, to which sensitivity analysis can contribute. These include insights into mechanisms generating the emergent properties of models, robustness of these insights, and quantification of the output uncertainty depending on the model inputs (ten Broeke et al., 2016: 2.1).

Sensitivity analysis can also come in many guises. Depending on the subset of the parameter space under study, one can distinguish *local* and *global* sensitivity analysis. Intuitively, the local sensitivity analysis looks at the changes of the response surfaces in the neighbourhoods of specific points in the input space, while the global analysis examines the reactions of the output across the whole space (as long as an appropriate, ideally space-filling design is selected). Furthermore, sensitivity analysis can be either *descriptive* or *variance-based*, and either *model-free* or *model-based*, the latter involving approaches based on regression and other meta-models, such as GP emulators.

The descriptive approaches to evaluating output sensitivity typically involve graphical methods: the visual assessment ('eyeballing') of response surface plots (such as in Fig. 5.4), correlations and scatterplots can provide first insights into the responsiveness of the output to changes in individual inputs. In addition, some of the simple descriptive methods can be also model-based, for example those using standardised regression coefficients (Saltelli et al., 2000, 2008). This approach relies on estimating a linear regression model of an output variable y based on all standardised inputs, $z_{ij} = (x_{ij} - x_i)/\sigma_i$, where x_i and σ_i are the mean and standard deviation of the i th input calculated for all design points j . Having estimated a regression

model on the whole design space $\mathbf{Z} = \{(z_{ij}, y_j)\}$, we can subsequently compare the absolute values of the estimated coefficients to infer about the relative influence of their corresponding inputs on the model output.

Variance-based approaches, in turn, aim at assessing how much of the output variance is due to the variation in individual inputs and their combinations. Here again, both model-free and model-based approaches exist, which differ in terms of whether the variance decomposition is analysed directly, based on model inputs and outputs, or whether it is based on some meta-model that is fitted to the data first. As observed by Ginot et al. (2006), one of the simplest, although seldom used methods here is the analysis of variance (ANOVA), coupled with the factorial design. Here, as in the classical ANOVA approach, the overall sum of squared differences between individual outputs and their mean value can be decomposed into the sums of squares related to all individual effects (inputs), plus a residual sum of squares (Ginot et al., 2006). This approach offers a quick approximation of the relative importance of the various inputs.

The state-of-the-art approaches, however, are typically based on the decomposition of variance and on so-called Sobol' indices. Both in model-free and model-based approaches, the template for the analysis is the same. Formally, let overall output variance in a model with K inputs be denoted by $V = \text{Var}[f(\mathbf{x})]$. Let us then define the *sensitivity variances* for individual inputs i and all their multi-way combinations, denoted by $V_i, V_{ij}, \dots, V_{12\dots K}$. These sensitivity variances measure by how much the overall variance V would reduce if we observed particular sets of inputs, $x_i, \{x_i, x_j\} \dots \{x_1, x_2 \dots x_K\}$, respectively. Formally, the sensitivity variances can be defined as $V_S = V - E\{\text{Var}[f(\mathbf{x})|\mathbf{x}_S = \mathbf{x}_S^*]\}$, where $\mathbf{S}$ denotes any non-empty set of individual inputs and their combinations. The overall variance V can then be additively decomposed into terms corresponding to the inputs and their respective combinations (e.g. Saltelli et al., 2000: 381):

$$V = \sum_i V_i + \sum_{i < j} V_{ij} + \dots + V_{12\dots K} \quad (5.2)$$

Based on (5.2), the sensitivity indices (or Sobol' indices) S can be calculated, which are defined as shares of individual sensitivity variances in the total V , $S_i = V_i/V$, $S_{ij} = V_{ij}/V$, $\dots$, $S_{12\dots K} = V_{12\dots K}/V$ (e.g. Sobol', 2001; Saltelli et al., 2008). These indices, adding up to one, have clear interpretations in terms of variance shares that can be attributed to each input and each combination of inputs.

The model-based variant of the variance-based approach is based on some meta-model fitted to the experimental data; such a meta-model can involve, for example, a Bayesian version of the GP, which was given a fully probabilistic treatment by Oakley and O'Hagan (2004). Another special case of the sensitivity analysis is *decision-based*: it looks at the effect of varying the inputs on the decision based on the output, rather than the output as such. Again, this can involve model-based

approaches, which can be embedded within the Bayesian decision analysis, coupling the estimates with loss functions related to specific outputs (*idem*).

In addition to the methods for global sensitivity analysis, local methods may include evaluating partial derivatives of the output function $f(\cdot)$ – or its emulator – in the interesting areas of the parameter space (Oakley & O’Hagan, 2004). In practice, this is often done by the means of a ‘one-factor-at-a-time’ method, where one of the model inputs is varied, while others are kept fixed (ten Broeke et al., 2016). This approach can help identify the type and shape of one-way relationships (*idem*). In terms of a comprehensive treatment of the various aspects of sensitivity analysis, a detailed overview and discussion can be found in Saltelli et al. (2008), while a fully probabilistic treatment, involving Bayesian GP emulators, can be found in Oakley and O’Hagan (2004). In the context of agent-based models, ten Broeke et al. (2016) have provided additional discussion and interpretations, while applications to demographic simulations can be found for example in Bijak et al. (2013) and Silverman et al. (2013).

To illustrate some of the key concepts, the example of the model of migration routes is continued in Box 5.3 (with further details in Appendix C). This example summarises results of the uncertainty and global variance-based sensitivity analysis, based on the fitted GP emulators.

Box 5.3: Uncertainty and Sensitivity of the Routes and Rumours Model

In terms of the uncertainty of the emulators presented in Box 5.2, the fitted variance of the GPs for standardised outputs, representing the uncertainty induced by the input variables and the intrinsic randomness (nugget) of the stochastic model code, ranged from 1.14 for *mean_freq_plan*, to 1.50 for *std_link_c*, to 1.65 *corr_opt_links*. The nugget terms were respectively equal 0.009, 0.020 and 0.019. For the cross-replicate output variable, *prop_std*, the variances were visibly higher, with 4.15 overall and 0.23 attributed to the code error.

As for the sensitivity analysis, for all four outputs the parameters related to information exchange proved most relevant, especially the probability of exchanging information through communication, as well as the information error – a finding that was largely independent of the priors assumed for the parameters (Fig. 5.6). In neither case did parameters related to exploration matter much.

Fig. 5.6 Variance-based sensitivity analysis: variance proportions associated with individual variables and their interactions, under different priors. (Source: own elaboration)

5.5 Bayesian Methods for Model Calibration

Emulators, such as the GPs introduced in Sect. 5.3, can serve as tools for calibrating the underlying complex models. There are many ways in which this objective can be achieved. Given that the emulators can be built and fitted by using Bayesian methods, a natural option for calibration is to utilise full Bayesian inference about the distributions of inputs and outputs based on data (Kennedy & O’Hagan 2001; Oakley & O’Hagan, 2002; MUCM, 2021). Specifically in the context of agent-based models, various statistical methods and aspects of model analysis are also reviewed in Banks and Norton (2014) and Heard et al. (2015).

The fully Bayesian approach proposed by Kennedy and O’Hagan (2001) focuses on learning about the calibration parameters θ of the model or, for complex models, its emulator, based on data. Such parameters are given prior assumptions, which are subsequently updated based on observed data to yield calibrated posterior distributions. However, as mentioned in Sect. 5.3, even at the calibrated values of the input parameters, model discrepancy – a difference between the model outcomes and observations – remains, and needs to be formally acknowledged too. Hence, the general version of the calibration model for the underlying computational model (or meta-model) f based on the training sample $\mathbf{x}$ and the corresponding observed data $z(\mathbf{x})$, has the following form (Kennedy & O’Hagan, 2001: 435; notation after Hilton, 2017):

$$z(\mathbf{x}) = \rho f(\mathbf{x}, \theta) + \delta(\mathbf{x}) + \varepsilon(\mathbf{x}). \quad (5.3)$$

In this model, $\delta(\mathbf{x})$ represents the discrepancy term, $\varepsilon(\mathbf{x})$ is the residual observation error, and ρ is the scaling constant. GPs are the conventional choices of priors both for $f(\mathbf{x}, \theta)$ and $\delta(\mathbf{x})$. For the latter term, the informative priors for the relevant parameters typically need to be elicited from domain experts in a subjective Bayesian fashion, to avoid problems with the non-identifiability of both GPs (*idem*).

The calibrated model (5.3) can be subsequently used for prediction, and also for carrying out additional uncertainty and sensitivity checks, as described before. Existing applications to agent-based models of demographic or other social processes are scarce, with the notable exception of the analysis of a demographic micro-simulation model of population dynamics in the United Kingdom, presented by Hilton (2017), and, more recently, an analysis of ecological demographic models, as well as epidemiological ‘compartment’ models discussed by Hooten et al. (2021).

Emulator-based and other more involved statistical approaches are especially applicable wherever the models are too complex and their parameter spaces have too many dimensions to be treated, for example, by using simple Monte Carlo algorithms. In such cases, besides GPs or other similar emulators, several other approaches can be used as alternative or complementary to the fully Bayesian inference. We briefly discuss these next. Detailed explanations of these methods are beyond the scope of this chapter, but can be explored further in the references (see also Hooten et al., 2020 for a high-level overview, with a slightly different emphasis).

- **Approximate Bayesian Computation (ABC).** This method relies on sampling from the prior distributions for the parameters of a complex model, comparing the resulting model outputs with actual data, and rejecting those samples for which the difference between the outputs and the data exceeds a pre-defined threshold. As the method does not involve evaluating the likelihood function, it can be computationally less costly than alternative approaches, although it can very quickly become inefficient in many-dimensional parameter spaces. The theory underpinning this approach dates to Tavaré et al. (1997), with more recent overviews offered in Marin et al. (2012) and Sisson et al. (2018). Applications to calibrating agent-based models in the ecological context were discussed by van der Vaart et al. (2015).
- **Bayes linear methods, and history matching.** In this approach, the emulator is specified in terms of the two first moments (mean and covariance function) of the

output function, and a simplified (linear) Bayesian updating is used to derive the expected posterior moments given the model inputs and outputs from the training sample, under the squared error loss (Vernon et al., 2010). Once built, the emulator is fitted to the observed empirical data by comparing them with the model outputs by using measures of *implausibility*, in an iterative process known as history *matching* (*idem*). For many practical applications, especially those involving highly-dimensional parameter spaces, the history matching approach is computationally more efficient than the fully Bayesian approach of Kennedy and O’Hagan (2001), although at the expense of providing an approximate solution (for more detailed arguments, see e.g. the discussion of Vernon et al., 2010, or Hilton, 2017). Examples of applying these methods to agent-based approaches include a model of HIV epidemics by Andrianakis et al. (2015), as well as models of a demographic simulation and fertility developments in response to labour market changes (the so-called Easterlin effect) by Hilton (2017).

- **Bayesian melding.** This approach ‘melds’ two types of prior distributions for the model output variable: ‘pre-model’, set for individual model inputs and parameters and propagated into the output, and ‘post-model’, set directly at the level of the output. The two resulting prior distributions for the output are weighted (linearly or logarithmically) by being assigned weights a and $(1-a)$, respectively, and the posterior distribution is calculated based on such a weighted prior. The underpinning theory was proposed by Raftery et al. (1995) and Poole and Raftery (2000). In a recent extension, Yang and Gua (2019) proposed treating the pooling parameter a as another hyper-parameter of the model, which is also subject to estimation through the means of Bayesian inference. An example of an application of Bayesian melding to an agent-based modelling of transportation can be found in Ševčíková et al. (2007).
- **Polynomial chaos.** This method, originally stemming from applied mathematics (see O’Hagan, 2013), uses polynomial approximations to model the mapping between model inputs and outputs. In other words, the output is modelled as a function of inputs by using a series of polynomials with individual and mixed terms, up to a specified degree. The method was explained in more detail from the point of view of uncertainty quantification in O’Hagan (2013), where it was also compared with GP-based emulators. The conclusion of the comparison was that, albeit computationally promising, polynomial chaos does not (yet) account for all different sources of uncertainty, which calls for closer communication between the applied mathematics and statistics/uncertainty quantification communities. A relevant example, using polynomial chaos in an agent-based model of a fire evacuation, was offered by Xie et al. (2014).
- **Recursive Bayesian approach.** This method, designed by Hooten et al. (2019, 2020), aims to make full use of the natural Bayesian mechanism for sequential updating in the context of time series or similar processes, whereby the posterior distributions of the parameters of interest are updated one observation at a time. The approach relies on a recursive partition of the posterior for the whole series into a sequence of sub-series of different lengths (Hooten et al. 2020), which can be computed iteratively. The computational details and the choice of appropriate sampling algorithms were discussed in more detail in Hooten et al. (2019).

We conclude this chapter by providing an example of calibrating the migration route formation model, which is presented in Box 5.4.

Box 5.4: Calibration of the Routes and Rumours Model

In order to demonstrate the use of calibration techniques, a set of representative values from the previous set of experimental samples was treated as ‘observed data’ against which to calibrate. Principal components were taken from a normalised matrix of samples of the output variables *mean_freq_plan*, *corr_opt_links*, and *stdd_link_c* to transform to a set of orthogonal coordinates. The variable *prop_stdd* was not used because it refers to summaries of repeated simulations; these cannot even theoretically be observed, as they would correspond outcomes from many different possible histories. Following Higdon (2008), separate GP emulators were then fitted to the mean of the principal component scores at each design point, with the variation over repetitions added as a variance term that is allowed to vary over the design. The DiceKriging R package was used to fit all emulators (Roustant et al., 2012), and k -fold cross validation indicated that the emulators captured the variation in the simulator reasonably well. A simplified but multivariate version of the model discussed in Sect. 5.3 was employed for the purposes of calibration, with ρ set to 1 and with the discrepancy and observation error terms assumed to independently and identically (normally) distributed. Posterior distributions for the unknown calibration parameters θ were obtained from this model using the *stan* Bayesian modelling package (Stan Development Team, 2021). Non-informative Beta(1,1) priors were used for the calibration parameters.

Figure 5.7 shows the resultant calibrated posterior distributions. As the sensitivity analysis showed, *p_transfer_info* has the greatest effect on simulator outputs, and therefore we gain more information about this parameter during the calibration process, while the posteriors indicate that a wide range of values of other parameters could replicate the observed values, given our uncertainty about the simulator and about reality, and taking into account the stochasticity of the simulator itself. Still, the wide uncertainty in the posterior distributions for the most parameter values is not surprising: it reflects the high uncertainty of the process itself. In a general case, such high residual errors remaining after calibration could illuminate the areas where the uncertainty might be either irreducible (aleatory), or at least difficult to reduce given the available set of calibration data that was used for that purpose.

Figure 5.8 shows that the resulting calibrated predicted emulator outputs are close to the target values (red dotted lines). This means that running the simulator on samples from the calibrated posterior of the input parameters is expected to produce a multivariate distribution of output values centred on our observed values.

Fig. 5.7 Calibrated posterior distributions for Routes and Rumours model parameters

Fig. 5.8 Posterior calibrated emulator output distributions

Open Access This chapter is licensed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.