No free lunch when estimating simulation parameters

true

2019-05-08

Abstract

We estimate the parameters of 21 simulation models using 9 estimation algorithms to discover which one is better at matching simulations to data. Unfortunately no single algorithm is best at minimizing estimation error for all or even most the simulations; the best algorithm differs for each simulation, and sometimes for each parameter of each simulation. Cross-validation is necessary to pair each simulation's parameter to its appropriate estimation algorithm. This is computationally feasible for reference table algorithms. In terms of confidence intervals the results are more clear: bootstrap generates more precise prediction intervals than either quantiles or Approximate Bayesian Computation.

Contents

Keywords

Agent-based models; Individual-based models; Estimation; Calibration; Approximate Bayesian Computation; Random Forest; Generalized Additive Model; Bootstrap;

1 Introduction

Models take parameters as input to describe and predict a phenomenon. When we gather real data we can search for the input parameters that most likely generated it. This is difficult for simulation models whose likelihood function is often unknown or intractable. Fortunately, many likelihood-free estimation algorithms are available(Hartig et al. 2011). We would like to know if any of them estimate parameters better, particularly when data is high dimensional, simulating is computationally expensive and in the presence of identification issues.

In this paper we estimate the parameters of 21 simulations with nine different algorithms. We focus on "reference table" algorithms: rejection ABC (Approximate Bayesian Computation as in Beaumont, Zhang, and Balding 2002), regression-adjusted ABC and regression-only methods. We rank them by their estimation error and the quality of their confidence intervals. Superficially, there is a winner: GAM, generalized additive models, regressions (Hastie and Tibshirani 1986; Wood and Augustin 2002) produce most often the lowest estimation errors, the best confidence intervals and no identification failures. Even so, GAMs produce the best estimates for only 30% of the parameters. In fact, each of the algorithms tested is the best estimator in at least one instance.

Since best estimation algorithm changes between simulations and even between parameters of the same simulation, there is no alternative to test multiple algorithms against each parameter of a simulation. Fortunately this plays to the advantage of reference table algorithms since the data used to train one algorithm can be recycled to train all the others as well as rank them by cross-validation.

2 Materials and Methods

We define here a simulation model as any function that depends on a set of parameter θ to generate a set of summary statistic $S(\theta)$. We are interested in the estimation problem where we observe summary statistics S^* and we want to know which parameter θ^* most likely generated them.

We parametrize 21 simulation models (described in section 2.1). We have nine candidate algorithms to do so (described in section 2.2). All are "reference table" algorithms: algorithms whose only input for estimation is a table of simulation parameters θ, selected by random sampling, and the summary statistics $S(\theta)$ they generate. We split this reference table into training and testing sets and ask each algorithm to estimate the parameters of the testing set observing only the training portion of the reference table.

We measure algorithm performance with two indicators: predictivity and coverage. Predictivity (Salle and Yıldızoğlu 2014; modelling efficiency in Stow et al. 2009) is the out of sample mean square error when using a particular algorithm normalized by the mean square error of using the average parameter instead:

$$1 - \frac{\sum \left(\theta_i - \hat{\theta}_i\right)^2}{\sum \left(\theta_i - \bar{\theta}\right)^2} \tag{1}$$

Where $\hat{\theta}$ is the estimated parameter, θ is the real simulation parameter and $\bar{\theta}$ is the average parameter value. Predictivity ranges from 1 (perfectly estimated) to 0 (unidentified) to negative values (misidentified).

We define coverage as in Raynal et al. (2018) as the percentage of times the real parameter falls within the 95% prediction intervals suggested by the estimating algorithm. The best coverage is 95%: higher generates type I errors, lower generates type II errors.

2.1 Models

We estimate the parameters of 21 separate simulations, repeating some with different amount of data or summary statistics. We can roughly categorize the simulations into three groups: simple, ill posed and complicated. Table 1 lists them all.

Table 1: List of all the models parametrized

Experiment	No. of parameters	No. of summary statistics	No. of simulations	Testing
α-stable	3	11	1,250 or 5,000	5-fold CV
Birds ABM	2	2 or 105	5,000	5-fold CV
Broken Line	1	10	1,250 or 5,000	5-fold CV
Coalescence	2	7	100,000	Single testing set
Earthworm	11	160	100,000	Single testing set
g-and-k distribution	4	11	1,250 or 5,000	5-fold CV

Experiment	No. of parameters	No. of summary statistics	No. of simulations	Testing
Hierarchical Normal Mean	2	61	1,250 or 5,000	5-fold CV
Lotke-Volterra	2	16 (noisy or non-noisy)	100,000	Single testing set
Locally Identifiable	2	2	1,250 or 5,000	5-fold CV
Moving Average (2)	2	2	1,250 or 5,000	5-fold CV
Median and MAD	2	2 or 4	1,250 or 5,000	5-fold CV
μ-σ^2	2	2	10,000	5-fold CV
Normal 25	2	25	1,250 or 5,000	5-fold CV
Scale	2	1	1,250 or 5,000	5-fold CV
Unidentifiable	2	1	1,250 or 5,000	5-fold CV
Partially Identifiable	2	2	1,250 or 5,000	5-fold CV
Real Business Cycle	6	44 or 48	2,944 or 2,961	5-fold CV
Pathogen	4	11	200,000	Single testing set
Toy Model	2	2	1,250 or 5,000	5-fold CV
Ecological Traits	4	4	1,250 or 5,000	5-fold CV
Wilkinson	1	1	1,250 or 5,000	5-fold CV

2.1.1 Simple simulations

Simple simulations, few parameters and summary statistics, feature prominently in the ABC literature both as a teaching tool and to compare different techniques. They are useful because they run quickly but they may bias comparisons towards simpler estimation algorithms. We compute predictivity and coverage for all the experiments in this section by 5-fold cross-validation: keeping one fifth of the data out of sample, using the remaining portion to train our algorithms and doing this five times, rotating each time the portion of data used for testing. We run all the experiments in this section twice: once the total data is 1,250 simulation runs and once it is 5,000 simulation runs.

α-stable: Rubio and Johansen (2013) uses ABC to recover the parameters of an α-stable distribution by looking at sample of 1096 independent observations from it. We replicate this here using the original priors for the three parameters ($\alpha \sim U(1, 2)$, $\mu \sim U(-0.1, 0.1)$, $\sigma \sim U(0.0035, 0.0125)$). We use 11 summary statistics representing the 0%,10%,. . .,100% deciles of each sample generated.

g-and-k distribution: Karabatsos and Leisen (2017) uses ABC to estimate the parameters of the g-and-k distribution (an extension of the normal distribution whose density function has no analytical expression). We replicate this here using the `gk` package in R (Prangle 2017). We want to retrieve the 4 parameters of the distribution $A, B, g, k \sim U[0, 10]$ given the 11 deciles (0%,10%,. . .,100%) of a sample of 1,000 observations from that distribution.

Normal 25: Sometimes sufficient summary statistics exist but the modeller may miss them and use others of lower quality. In this example 25 i.i.d observations from the same normal distribution $\sim N(\mu, \sigma^2)|\mu \sim U(-5, 5); \sigma \sim U(1, 10)$ are used directly as summary statistics to retrieve the two distribution parameters μ, σ^2.

Moving Average(2): Creel (2017) used neural networks to recover the parameters of the MA(2) process with $\beta_1 \sim U(-2, 2); \beta_2 \sim U(-1, 1)$. We observe generated a time series of size 100 and we summarise it with the coefficients a AR(10) regression.

Median and MAD: As a simple experiment we sample 100 observations from a normal distribution $\mu \sim U(-5, 5)$ and $\sigma \sim U(0.1, 10)$ and we collect as summary statistics their median and median absolute deviation, using them to retrieve the original distributions. We run this experiment twice, the second time adding two useless summary statistics $S_3 \sim N(3, 1)$ and $S_4 \sim N(100, .01)$.

μ-σ^2: The `abc` package in R (Csilléry, François, and Blum 2012) provides a simple dataset example connecting two observed statistics: "mean"" and "variance" as" generated by the parameters μ and σ^2. The posterior that connects the two derives from the Iris setosa observation (Anderson 1935). The data set contains 10,000 observations and we log-transform σ^2 when estimating.

Toy Model: A simple toy model suggested by the `EasyABC` R package(Jabot, Faure, and Dumoulin 2013) involves retrieving two parameters, $a \sim U[0,1]; b \sim U[1,2]$, observing two summary statistics $S_1 = a + b + \epsilon_1; S_2 = ab + \epsilon_2 | \epsilon_1, \epsilon_2 \sim N(0, .1^2)$.

Ecological Traits: The `EasyABC` R package(Jabot, Faure, and Dumoulin 2013) provides a replication of Jabot (2010), a trait-based ecological simulator. Here we fix the number of individuals to 500 and the number of traits to 1, leaving four free parameters: $I \sim U(3,5), A \sim U(0.1,5), h \sim U(-25,125), \sigma \sim U(0.5,25)$. We want to estimate these with four summary statistics: richness of community S, shannon index H, mean and skewness of traiv values in the community.

Wilkinson: Wilkinson (2013) suggested a simple toy model with one parameter, $\theta \sim U(-10,10)$, and one summary statistic $S_1 \sim N(2(\theta+2)\theta(\theta-2), 0.1+\theta^2)$. We run this experiment twice, once where the total data is 1,250 sets of summary statistics and one where the total data is 5,000 sets of summary statistics.

2.1.2 Ill-posed Models

We often face identification issues: the inability to recover parameters given the information we have. Because these issues take many forms, we produce a series of experiments to test each. As with simple simulations, we test all the experiments with 5-fold cross validation and run each twice: once where the total reference table has 1,250 total rows, and once where it has 5,000.

Ideally two behaviours should emerge from an estimation algorithm under these circumstances. First we would like to maximize the quality of our estimated parameters when the information is noisy (the lesser problem of "weak" identification). Second we would like our estimation algorithm to recognize when the model cannot be identified and not be fooled into still producing an arbitrary estimate and a small confidence interval around it.

Broken Line: we observe 10 summary statistics $S = (S_0, \ldots, S_9)$ generated by:

$$S_i = \begin{cases} \epsilon & i < 5 \\ \beta i + \epsilon & i \geq 5 \end{cases}$$

and where $\beta \sim U(0,2)$

Hierarchical Normal Mean: Raynal et al. (2018) compares ABC to direct random forest estimation in a "toy" hierarchical normal mean model:

$$\begin{aligned} y_i | \theta_1, \theta_2 &\sim N(\theta_1, \theta_2) \\ \theta_1 | \theta_2 &\sim N(0, \theta_2) \\ \theta_2 &\sim IG(\kappa, \lambda) \end{aligned}$$

Where $IG(\cdot)$ is the inverse gamma distribution. We want to estimate θ_1, θ_2 given a sampled vecor y of size 10 which is described by 61 summary statistics: the mean, the variance, the median absolute deviation of the sample, all possible combinations of their products and sums as well as 50 noise summary statistics $\sim U(0,1)$.

Locally Identifiable: macroeconomics often deals with structural models that are only locally identifiable (see Fernández-Villaverde, Rubio Ramírez, and Schorfheide 2016). These are models where the true parameter is only present in the data for some of its possible values. Here we use the example:

$$S_i = \begin{cases} y \sim N(\theta_1, \theta_2) & \theta_1 > 2, \theta_2 > 2 \\ y \sim N(0,1) & \text{Otherwise} \end{cases}$$

Where $\theta_1, \theta_2 \sim U[0.1, 5]$, each simulation we sample the vector y of size 100 and we collect its mean and standard deviation as summary statistics.

Scale: a common source of under-identification in economics occurs when "when two structural parameters enter the objective function only proportionally, making them separately unrecoverable"(Canova and Sala

2009). In this example, two people of weight $w_1, w_2 \sim U[80, 150]$ step together on a scale whose reading $S_1 = w_1 + w_2 + \epsilon | \epsilon \sim N(0, 1)$ is the only summary statistic we can use. This problem is locally identifiable to an extent: very low readings means both people are light (and viceversa).

Unidentifiable: in some cases the model parameters are just unrecoverable and we hope that our estimation algorithm does not tell us otherwise. In this example the three summary statistics $S_1, S_2, S_3 \sim N(x, 1) | x \sim U[0, 50]$ provide no information regarding the two parameters we are interested in: $\mu \sim U(0, 50), \sigma \sim U(0, 25)$.

Partially Identifiable: Fernández-Villaverde, Rubio Ramírez, and Schorfheide (2016) mention how partial identification can occur when a model is the real data generating process conditional on some other unobserved parameter. This makes the model identifiable in some samples but not others. The example we use is a slight modification of the original: we try to retrieve parameter $\theta \sim U[1, 5]$ when we observe mean and standard deviation of a size 10 vector y generated as follows:

$$y \sim N(\theta \cdot x, 1), \; x = \begin{cases} 0 & \text{with probability } \frac{1}{2} \\ \sim N(1, 1) & \text{Otherwise} \end{cases}$$

2.1.3 Complicated Models

Simulations, particularly agent-based models, tend to be large. They involve many input parameters and summary statistics. Inference in a high-dimensional space is difficult not just for ABC methods but for non-parametric smoothing as well (see section 4.5 in Wasserman 2006). In principle one could solve this by just simulating more data but complicated simulation models tend also to be slow. This puts a premium on the quality of the algorithms to extrapolate from the data they have.

Birds ABM: Thiele, Kurth, and Grimm (2014) estimated the parameters of a simple agent-based bird population model (originally in Railsback and Grimm (2012)) with ABC. Their paper provided an open source NETLOGO implementation of the model. The model depends on two parameters: `scout-prob`$\sim U[0, 0.5]$ and `survival-prob`$\sim U[0.95, 1]$. We ran this experiment twice, once where there are only 2 summary statistics: mean abundance and mean variation over 20 years, and one where are 105 (comprising the average, last value, standard deviation, range and the coefficients of fitting an AR(5) regression to the time series of abundance, variation, months spent foraging and average age within bird population). This experiment is useful because in the original specification (with 2 summary statistics) the `scout-prob` parameter is unidentifiable. For each experiment we ran the model 5000 times.

Coalescence: the `abctools` package (Nunes and Prangle 2015) provides 100,000 observations of 7 summary statistics from a DNA coalescent model depending on two parameters $\theta \sim u[2, 10]$ and $\rho \sim U[0, 10]$. Blum et al. (2013) in particular used this dataset to compare the quality of ABC dimensionality reduction schemes to better estimate the two parameters. This data-set is too big for cross-validation so in this experiment we simply used 1,250 observation as the testing data-set and the rest for training.

Lotke-Volterra: Toni et al. (2009) showcases SMC-ABC with a 2 species deterministic Lotke-Volterra model with 2 parameters: a, b.

$$\begin{cases} \frac{dx}{dt} = ax - yx \\ \frac{dy}{dt} = bxy - y \end{cases}$$

Here we assume $a, b \sim U(0, 10)$ (avoiding the negative values in the original paper). For each simulation we sample 8 observations for predator and prey at time $t = 1, 1.2, 2.4, 3.9, 5.7, 7.5, 9.6, 11.9, 14.5$ (as in the original paper). We run this experiment twice, once where data is observed perfectly and one where to each observation we add noise $\sim N(0, 0.5)$. In both experiments we do not perform 5-fold cross validation, rather we generate 100,000 sets of summary statistics for training and another 1,250 sets of summary statistics to test the parametrization.

Real Business Cycle: we want to parametrize the default Real Business Cycle model (a simple but outdated class of macro-economics models) implemented in the `gEcon` R package(Klima, Podemski, and Retkiewicz-Wijtiwiak 2018). It has 6 parameters $(\beta, \delta, \eta, \mu, \phi, \sigma)$ and we try to parametrize them in two separate experiments. In the first, we use as summary statistics the -10,+10 cross-correlation table between output Y, consumption C, investment I, interest rates r and employment L (44 summary statistics in total). For this experiment we have 2,944 distinct observations. In the second experiment we follow Carrella, Bailey, and Madsen (2018) using as summary statistics (i) coefficients of regressing Y on Y_{t-1}, I_t, I_{t-1}, (ii) coefficients of regressing Y on Y_{t-1}, C_t, C_{t-1}, (iii) coefficients of regressing Y on Y_{t-1}, r_t, r_{t-1}, (iv) coefficients of regressing Y on Y_{t-1}, L_t, L_{t-1}, (v) coefficients of regressing Y on C, r (vi) coefficients of fitting AR(5) on Y, (vii) the (lower triangular) covariance matrix of Y, I, C, r, L. 48 summary statistics in total. For this experiment we have 2,961 distinct observations.

Pathogen: another dataset used by Blum et al. (2013) to test dimensionality reduction methods for ABC concerns the ability to predict pathogens' fitness changes due to antibiotic resistance (the original model and data is from Francis et al. 2009). The model has four free parameters and 11 summary statistics. While the original data-set contains 1,000,000 separate observations, we only sample 200,000 at random for training the algorithms and 1,250 more for testing.

Earthworm: Vaart et al. (2015) calibrated an agent-based model of earthworms with rejection ABC. The simplified version of the model contains 11 parameters and 160 summary statistics. The original paper already carried out cross-validation proving under-identification: the model contains a mixture of unidentified, weakly identified and well identified parameters. We use 100,000 runs from the original paper, setting 1,250 aside for out of sample testing and using the rest for training.

2.2 Algorithms

We test nine algorithms to parametrize simulations: five are ABC and four are regressions-only. In this paper we focused exclusively on reference table algorithms. We thus ignored methods that combine search and estimation, such as synthetic likelihood (Wood 2010; Fasiolo and Wood 2014), ABC-MCMC (Hartig et al. 2011) and Bayesian optimization (Snoek, Larochelle, and Adams 2012). We also ignored regression-only techniques that do not generate prediction intervals such as the deep neural networks proposed in Creel (2017) and the elastic nets proposed in Carrella, Bailey, and Madsen (2018).

2.2.1 ABC

The first algorithm we use is the simple rejection ABC (Pritchard et al. 1999; Beaumont, Zhang, and Balding 2002). Start by ranking all training observations by their euclidean distance to the testing summary statistics $\left(\sum_i S_i(\theta) - S_i(\theta^*)\right)^2$. Ignore all training observations except the closest 10%. Use the θ parameters of the accepted observations to generate the posterior estimate for the testing parameter θ^*.

The second algorithm is the local-linear regression adjusted ABC (Beaumont, Zhang, and Balding 2002). Weigh all training observations by an Epanechnikov kernel with bandwidth equal to euclidean the distance between the testing summary statistics $S(\theta^*)$ and the furthest $S(\theta)$ we would have accepted using simple rejection. Then run a local-linear regression on the weighted training set to predict $E[\theta|S(\theta)]$ and use the residuals of that regression to estimate the posterior distribution for the testing parameter θ^*.

The third algorithm, neural network ABC, feeds the same weighted training set to a feed forward neural network (Blum and Francois 2010). The approach is similar to the local-linear regression above but the residuals are also weighted by a second regression (on the log squared residuals) to correct for heteroskedasticity.

These three algorithms are implemented in the `abc` package(Csilléry, François, and Blum 2012) in R. We used the package default settings for its neural networks (10 networks, 5 units in the hidden layer and weight decay randomly chosen for each network between 0.0001,0.001 and 0.01).

The fourth and fifth algorithm are semi-automatic ABC methods which "pre-process" summary statistics before applying rejection ABC(Prangle et al. 2014). More precisely, the original summary statistics $S(\theta)$ are fed into a set linear regressions estimating $r_i = E[\theta_i|S(\theta)]$ (one for each parameter of the model) and then regression fitted values are used as summary statistics for the simple rejection ABC. The rationale is that these regressions will project the summary statistics into a space where rejection ABC performs better. We do this in two different ways here: by running first or fourth degree linear regressions in the pre-processing phase. This is done using the R package `abctools`(Nunes and Prangle 2015) and their default parameters: using half of the training set to run the regression and the other half to run the rejection ABC.

A feature of all ABC methods is that they are local: they remove or weight training observations differently depending on the θ^* we want to estimate. This means that during cross-validation we need to retrain each ABC for each row of the testing set.

2.2.2 Regression Only

Estimating parameters by regression is a straightforward process. We build a separate regression r for each θ in the training set as dependent variable using the summary statistics $S(\theta)$ as the independent variables. We then plug in these regressions the testing summary statistic $S(\theta^*)$ to predict the testing parameter θ^*. When a simulation depends on multiple parameters we build a separate regression for each parameter.

The simplest algorithm of this class is linear regression of degree one. It is linear, its output is understandable and is fast to compute. This speed allows us to estimate the prediction interval of θ^* by resampling bootstrap(Davison and Hinkley 1997): we produce 200 training data sets by resampling with replacement from the original one and run the same linear regression on on each new data set. From each regression i we collect their prediction $\beta_i S(\theta^*)$ and sample one standardized residual e (a residual divided by the square root of one minus the hat value associated with that residual). This produces a set of 200 $\beta_i S(\theta) + e_i$. The 95% prediction interval is then defined by 2.5 and 97.5 percentile of this set.

A more complex algorithm that is not linear but still additive is the generalized additive model(GAM), where we regress:

$$\theta = \sum s_i(S_i(\theta))$$

s_i is a smooth spline transformation (see chapter 9 in Hastie, Tibshirani, and Friedman 2009; also Wood and Augustin 2002). We use the `mgcv` R package(Wood 2017, 2004). The boostrap prediction intervals techniques we used for linear regressions is too computationally expensive to replicate with GAMs. Instead we produce prediction intervals by assuming normal standard errors (generated by the regression itself) and by resampling residuals directly: we generate 10,000 draws of $z(S(\theta)) + \epsilon$ where z is normally distributed with standard deviation equal to regression's standard error at $S(\theta)$ and ϵ is a randomly drawn residual of the original regression.
The 95% prediction interval for θ^* is then defined by 2.5 and 97.5 percentile of the generated $z(S(\theta^*)) + \epsilon$ set.

A completely non-parametric regression advocated in Raynal et al. (2018) is the random forest(Breiman 2001). We implement this in two ways here. First, as a quantile random forest (Meinshausen 2006), using in `quantregForest` R package (Meinshausen 2017); prediction intervals for any simulation parameter θ^* are the predicted 2.5 and 97.5 quantile at $S(\theta^*)$. Second, as a regression random forest using the `ranger` and `caret` packages in R (Wright and Ziegler 2015; Kuhn 2008). For this method we generate prediction intervals as in GAM regressions. We generate 10,000 draws of $z(S(\theta)) + \epsilon$ where z is normally distributed with standard deviation equal to the infinitesimal jackknife standard error(Wager, Hastie, and Efron 2014) at $S(\theta)$ and ϵ is a resampled residual; we then take the 2.5 and 97.5 percentile of the $z(S(\theta^*)) + \epsilon$ set as our prediction interval.

3 Results

3.1 Predictivity

Table 3 summarises the predictivity of each algorithm across all identifiable estimation problems (here defined as those where at least one algorithm achieves predictivity of 0.05 or above). Estimation by GAM achieves the highest average predictivity and the lowest regret (average distance between its predictivity and the highest predictivity in each simulation). Even so, GAM has the best predictivity only for 30 out of a total of 93 identifiable parameters.

Non-linear methods like Random Forest and Neural Network ABC have high average predictivity and together they account for the top performance of another 35 parameters. Local-linear ABC is on average a very effective algorithm but its average predictivity is penalized by very large errors in a few experiments (in particular the "Hierarchical Normal Mean").

Table 2: Table showing for each algorithm how many parameters were best estimated by that algorithm; regret, defined as the average % loss between the predictivity of the algorithm and the best predictivity in each estimation; the average predictivity overall. Local-linear ABC average is distorted by its poor performance in the 'Hierarchical Normal Mean' simulation. Only estimations for which at least one method achieved predictivity above 0.05 were considered.

Algorithm	# of times highest predictivity	Regret	Average predictivity
Rejection ABC	2	-0.433	0.348
Semi-automatic ABC 4D	2	-0.207	0.476
Semi-automatic ABC 1D	1	-0.297	0.424
Local-linear ABC	15	< -10	< -10
Neural Network ABC	12	-0.120	0.556
Linear Regression	8	-0.248	0.479
GAM	30	-0.072	0.577
Quantile Random Forest	3	-0.187	0.547
Regression Random Forest	20	-0.117	0.571

Table 3: Table showing for each algorithm how many parameters were best estimated by that algorithm; regret, defined as the average % loss between the predictivity of the algorithm and the best predictivity in each estimation; the average predictivity overall. Local-linear ABC average is distorted by its poor performance in the 'Hierarchical Normal Mean' simulation. Only estimations for which at least one method achieved predictivity above 0.05 were considered.

Algorithm	# of times highest predictivity	Regret	Average predictivity
Rejection ABC	2	-0.433	0.348
Semi-automatic ABC 4D	2	-0.207	0.476
Semi-automatic ABC 1D	1	-0.297	0.424
Local-linear ABC	15	< -10	< -10
Neural Network ABC	12	-0.120	0.556
Linear Regression	8	-0.248	0.479
GAM	30	-0.072	0.577
Quantile Random Forest	3	-0.187	0.547
Regression Random Forest	20	-0.117	0.571

Table 4 lists the number of identification failures: parameters for which at least another algorithm achieved predictivity of .1 but the algorithm in the table achieved predictivity below .05. Most parameters are either identified by all the algorithms or none of them. Local-linear regression struggled with the "natural mean hierarchy" simulation. Linear regression failed to estimate the b parameter from the Lotka-Volterra models, the σ parameter from the normal distribution and the A parameter from the ecological traits model. Random forests failed to identify μ and δ from the RBC macroeconomic model.

Table 4: In this table we tabulate the number of identification failures for each algorithm which we define as estimation problems where at least one other algorithm had predictivity above .1 but this algorithm had predictivity below 0.05

Algorithm	Identification Failures
Rejection ABC	11
Semi-automatic ABC 4D	1
Semi-automatic ABC 1D	1
Local-linear ABC	8
Neural Network ABC	0
Linear Regression	6
GAM	0
Quantile Random Forest	3
Regression Random Forest	2

Figure 1 compares algorithms pairwise with respect to their predictivity. Even the "best" algorithm, GAM, has lower predictivity for 30-40% of the parameters against to neural network ABC, local-linear regression ABC and random forests. Linear regression of degree one wins about half of the comparisons against any ABC.

It would be wrong however to infer that the choice of estimation algorithm is unimportant and one is better off spending their computational budget on running more simulations. To prove this we focus on all parameters that were estimated both with 1,250 and 5,000 training simulations. We run a linear mixed effect model where predictivity is the dependent variable and the parameter, the algorithm and the training data size are random effects. Table 5 shows that the coefficient associated with training data size is smaller (in absolute terms) than those associated with most algorithms. Finding the best algorithm then is usually better than quadrupling the training data set.

Table 5: Coefficients of running a linear mixed effects model regressing predictivity on algorithm used and size of the training sample (both fixed effects) with each estimation as a random intercept. Algorithm effects are with respect to linear regression.

Term	Estimate	Std. Error	t
Intercept	0.462	0.051	8.986
Local-linear ABC	0.043	0.023	1.874
GAM	0.077	0.022	3.468
Neural Network ABC	0.061	0.022	2.754
Rejection ABC	-0.082	0.022	-3.714
Quantile Random Forest	0.052	0.022	2.365
Regression Random Forest	0.082	0.022	3.707
Semi-automatic ABC 1D	-0.003	0.024	-0.128
Semi-automatic ABC 4D	0.024	0.022	1.090
Small Training Size	-0.020	0.011	-1.879

The appendix contains a table with the predictivity for all parameters generated by each algorithm.

3.2 Coverage

Results on the quality of confidence intervals are easier to interpret, as shown in table 6. Using bootstrap prediction intervals as confidence intervals is superior to both using ABC and quantiles. In terms of coverage

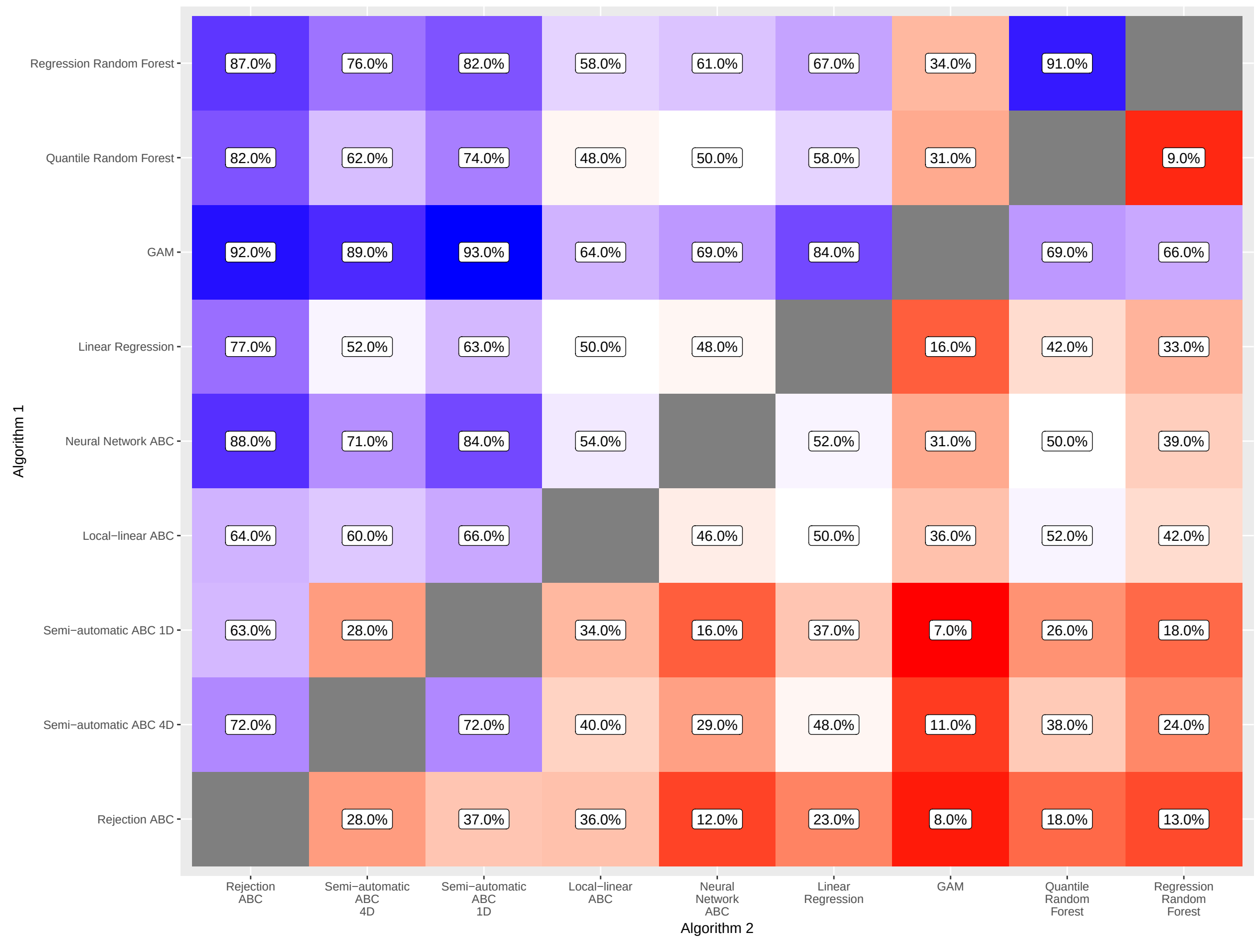

Figure 1: Percentage of times algorithm 1 has higher predictivity than algorithm 2 for all estimations where at least one algorithm achieves .1 or more predictivity. A blue cell means that algorithm 1 performs generally better, a red cell means that algorithm 2 does.

error, the absolute difference between 95% and the proportion of parameters that out of sample are within the generated intervals, GAM is the most accurate 42% of the time and have the lowest median error while linear regressions have the lowest mean coverage error. Regression adjusted ABC and quantile random forests make the largest coverage errors.

Table 6: Table showing for each algorithm median and average coverage error: the absolute difference between 95% and the proportion of parameters actually falling within the algorithm's stated 95% confidence interval (out of sample). The lower the error the more precise the algorithm. For each algorithm we also list the number of parameters for which the stated coverage was the most accurate out of sample compared to the other algorithms

Algorithm	# of times most accurate	Median Coverage Error	Mean Coverage Error
Rejection ABC	9	0.0124	0.0232
Semi-automatic ABC 4D	6	0.0188	0.0188
Semi-automatic ABC 1D	1	0.0152	0.0172
Local-linear ABC	2	0.0244	0.0613
Neural Network ABC	8	0.0364	0.1317
Linear Regression	17	0.0064	0.0071
GAM	46	0.0036	0.0137
Quantile Random Forest	6	0.0324	0.0400
Regression Random Forest	12	0.0110	0.0171

4 Discussion

This paper provides two key results. First, if we are concerned primarily with the quality of point estimates, there is no substitute for trying multiple algorithms and rank them by cross-validation. GAM regressions do provide a good starting point. Second, bootstrap prediction intervals are the best method to generate confidence intervals around parameter estimates.

The key advantage of reference table algorithms is that the same reference table can be used to train and test all algorithms at once. This is what makes cross-validation feasible since running simulations is usually the most expensive part of parameter estimation (accounting for 80% of the computational time in the simple agent-based model of Grazzini and Richiardi 2015). Compare this with search based methods such as simulated minimum distance: to cross-validate we would need to search the parameter space from scratch for each row of the testing set. This is unfeasible for all but the simplest models.

We know of no agent-based model that used cross-validation to choose how to estimate its parameters (with the exception of the comparison between ABC MCMC and simulated minimum distance in Grazzini, Richiardi, and Tsionas 2017). The common approach seems to be to pick one estimation method and apply it. We have proven here that this is suboptimal: no estimation method seems to be a priori better than the others.

Papers proposing a new estimation algorithms tend to showcase their approach against one or two examples. It would help the literature to have a larger, standardized set of experiments to gauge any newcomer. We hope this paper and its code repository to be a first step. However it may be impossible to find an estimation algorithm that is always best and we should prioritize methods for which cross-validation can be done without having to run more simulations.

The no free lunch theorem(Wolpert and Macready 1995) argues that when averaging over the universe of all search problems all optimization algorithms (including random search) perform equally. A supervised learning version of the same (Wolpert 2011) suggests that "on average" all learning algorithms and heuristics are equivalent. These are deeply theoretical results whose practical applications are limited: nobody has ever suggested abandoning cross-validation because of it, for example. However, some weak form of it seems to hold empirically when parametrizing: for each estimation algorithm there is a simulation parameter for which it does best.

Appendix

Full Predictivity Table

Experiment	Parameter	Rejection ABC	Semi-automatic ABC 4D	Semi-automatic ABC 1D
alpha-stable 5,000	alpha	0.425	0.591	0.608
alpha-stable 5,000	mu	0.633	0.727	0.727
alpha-stable 5,000	sigma	0.039	0.715	0.718
alpha-stable 1,250	alpha	0.414	0.559	0.602
alpha-stable 1,250	mu	0.656	0.718	0.705
alpha-stable 1,250	sigma	0.019	0.706	0.714
Birds ABM - 2 SS	scout.prob	0.004	0.051	0.004
Birds ABM - 2 SS	survival.prob	0.851	0.776	0.842
Birds ABM - 105 SS	scout.prob	0.194	0.472	0.405
Birds ABM - 105 SS	survival.prob	0.731	0.791	0.757
Broken Line 5,000	b	0.810	0.905	NA
Broken Line 1,250	b	0.810	0.899	NA
Coalescence	rho	0.094	0.161	0.146
Coalescence	theta	0.382	0.428	0.422
Earthworm	B_0	-0.021	-0.019	-0.019
Earthworm	E	0.216	0.254	0.261
Earthworm	E_c	0.023	0.048	0.038
Earthworm	E_s	-0.002	0.000	0.000
Earthworm	IGm	0.196	0.327	0.334
Earthworm	M_b	0.053	0.306	0.284
Earthworm	M_c	0.051	0.112	0.101
Earthworm	M_m	0.304	0.341	0.267
Earthworm	M_p	0.117	0.268	0.118
Earthworm	r_B	0.236	0.330	0.326
Earthworm	r_m	0.026	0.118	0.100
g-k distribution 5,000	A	0.322	0.634	0.574
g-k distribution 5,000	B	0.206	0.567	0.543
g-k distribution 5,000	g	0.045	0.401	0.404
g-k distribution 5,000	k	0.625	0.524	0.481
g-k distribution 1,250	A	0.334	0.589	0.563
g-k distribution 1,250	B	0.195	0.531	0.530
g-k distribution 1,250	g	0.052	0.356	0.398
g-k distribution 1,250	k	0.623	0.445	0.438
Hierarchical Normal Mean 1,250	theta1	0.409	0.509	0.510
Hierarchical Normal Mean 1,250	theta2	0.380	0.421	0.436
Hierarchical Normal Mean 5,000	theta1	0.429	0.430	0.522
Hierarchical Normal Mean 5,000	theta2	0.383	0.317	0.414
Lotke-Volterra Noisy	a	0.576	0.785	0.769
Lotke-Volterra Noisy	b	0.258	0.436	0.263
Lotke-Volterra Non-Noisy	a	0.529	0.792	0.774
Lotke-Volterra Non-Noisy	b	0.231	0.464	0.336
Locally Identifiable 5,000	theta1	0.199	0.207	0.202
Locally Identifiable 5,000	theta2	0.203	0.150	0.199
Locally Identifiable 1,250	theta1	0.184	0.191	0.186
Locally Identifiable 1,250	theta2	0.209	0.176	0.200
Moving Average 5,000	ma1	0.384	0.427	0.398
Moving Average 5,000	ma2	0.385	0.635	0.494
Moving Average 1,250	ma1	0.358	0.375	0.347
Moving Average 1,250	ma2	0.373	0.613	0.458
Median and MAD 5,000 - 2 SS	mu	0.754	0.747	0.753
Median and MAD 5,000 - 2 SS	sigma	0.769	0.769	0.768
Median and MAD 1,250 - 2 SS	mu	0.743	0.735	0.741
Median and MAD 1,250 - 2 SS	sigma	0.768	0.760	0.759
Median and MAD 5,000 - 4 SS	mu	0.648	0.744	NA

Acknowledgments

This research is funded in part by the Oxford Martin School, the David and Lucile Packard Foundation, the Gordon and Betty Moore Foundation, the Walton Family Foundation, and Ocean Conservancy.

Bibliography

Anderson. 1935. "The Irises of the Gaspe Peninsula." *Bulletin of the American Iris Society* 59.

Beaumont, Mark A, Wenyang Zhang, and David J Balding. 2002. "Approximate Bayesian computation in population genetics." *Genetics* 162 (4): 2025–35. https://doi.org/Genetics%20December%201,%202002%20vol.%20162%20no.%204%202025-2035.

Blum, M G B, M A Nunes, D Prangle, and S A Sisson. 2013. "A comparative review of dimension reduction methods in approximate Bayesian computation." *Statistical Science* 28 (2): 189–208. https://doi.org/10.1214/12-STS406.

Blum, Michael G B, and Olivier Francois. 2010. "Non-linear regression models for Approximate Bayesian Computation." *Statistics and Computing* 20 (1): 63–73. https://doi.org/10.1007/s11222-009-9116-0.

Breiman, Leo. 2001. "Random Forests." *Machine Learning* 45 (1): 5–32. https://doi.org/10.1023/A:1010933404324.

Canova, Fabio, and Luca Sala. 2009. "Back to square one: Identification issues in DSGE models." *Journal of Monetary Economics* 56 (4): 431–49. https://doi.org/10.1016/j.jmoneco.2009.03.014.

Carrella, Ernesto, Richard M. Bailey, and Jens Koed Madsen. 2018. "Indirect inference through prediction," July. http://arxiv.org/abs/1807.01579.

Creel, Michael. 2017. "Neural nets for indirect inference." *Econometrics and Statistics* 2 (April): 36–49. https://doi.org/10.1016/j.ecosta.2016.11.008.

Csilléry, Katalin, Olivier François, and Michael G. B. Blum. 2012. "Abc: An R package for approximate Bayesian computation (ABC)." *Methods in Ecology and Evolution* 3 (3): 475–79. https://doi.org/10.1111/j.2041-210X.2011.00179.x.

Davison, A. C. (Anthony Christopher), and D. V. Hinkley. 1997. *Bootstrap methods and their application*. Cambridge, United Kingdom: Cambridge University Press.

Fasiolo, M, and S Wood. 2014. "An introduction to synlik (2014)." *R Package Version 0.1. 0*.

Fernández-Villaverde, Jesús, Juan F Rubio Ramírez, and Frank Schorfheide. 2016. "Solution and Estimation Methods for DSGE Models." In *Handbook of Macroeconomics*, 2:527——724. https://doi.org/10.1016/bs.hesmac.2016.03.006.

Francis, Andrew R, Scott A Sisson, Honglin Jiang, Fabio Luciani, and Mark M Tanaka. 2009. "The epidemiological fitness cost of drug resistance in Mycobacterium tuberculosis." *Proceedings of the National Academy of Sciences* 106 (34): 14711–5. https://doi.org/10.1073/pnas.0902437106.

Grazzini, Jakob, and Matteo Richiardi. 2015. "Estimation of ergodic agent-based models by simulated minimum distance." *Journal of Economic Dynamics and Control* 51 (February): 148–65. https://doi.org/10.1016/j.jedc.2014.10.006.

Grazzini, Jakob, Matteo G. Richiardi, and Mike Tsionas. 2017. "Bayesian estimation of agent-based models." *Journal of Economic Dynamics and Control* 77 (April): 26–47. https://doi.org/10.1016/j.jedc.2017.01.014.

Hartig, F., J.M. Calabrese, B. Reineking, T. Wiegand, and A. Huth. 2011. "Statistical inference for stochastic simulation models - theory and application." *Ecology Letters* 14 (8): 816–27. https://doi.org/10.1111/j.1461-0248.2011.01640.x.

Hastie, Trevor, and Robert Tibshirani. 1986. "Generalized Additive Models." *Statistical Science* 1 (3): 297–318. http://web.stanford.edu/%7B~%7Dhastie/Papers/gam.pdf.

Hastie, Trevor, Robert Tibshirani, and Jerome Friedman. 2009. *The Elements of Statistical Learning*. 1st ed. Vol. 1. Berlin: Springer series in statistics Springer, Berlin. https://doi.org/10.1007/b94608.

Jabot, Franck. 2010. "A stochastic dispersal-limited trait-based model of community dynamics." *Journal of Theoretical Biology* 262 (4): 650–61. https://doi.org/10.1016/j.jtbi.2009.11.004.

Jabot, Franck, Thierry Faure, and Nicolas Dumoulin. 2013. "EasyABC: Performing efficient approximate Bayesian computation sampling schemes using R." Edited by Robert B. O'Hara. *Methods in Ecology and Evolution* 4 (7): 684–87. https://doi.org/10.1111/2041-210X.12050.

Karabatsos, George, and Fabrizio Leisen. 2017. "An Approximate Likelihood Perspective on ABC Methods." *Statistics Surveys* 12 (0): 66–104. https://doi.org/10.1214/18-SS120.

Klima, Grzegorz, Karol Podemski, and Kaja Retkiewicz-Wijtiwiak. 2018. "gEcon: General Equilibrium Economic Modelling Language and Solution Framework."

Kuhn, Max. 2008. "Building Predictive Models in R Using the caret Package." *Journal of Statistical Software* 28 (5): 1—–26. https://doi.org/10.18637/jss.v028.i05.

Meinshausen, Nicolai. 2006. "Quantile Regression Forests." Vol. 7. http://www.jmlr.org/papers/volume7/meinshausen06a/meinshausen06a.pdf.

———. 2017. "quantregForest: Quantile Regression Forests." https://cran.r-project.org/package=quantregForest.

Nunes, Matthew A, and Dennis Prangle. 2015. "abctools: An R Package for Tuning Approximate Bayesian Computation Analyses." *The R Journal* 7 (2): 189—–205. https://journal.r-project.org/archive/2015/RJ-2015-030/RJ-2015-030.pdf.

Prangle, Dennis. 2017. "gk: An R Package for the g-and-k and generalised g-and-h Distributions," June. http://arxiv.org/abs/1706.06889.

Prangle, Dennis, Paul Fearnhead, Murray P. Cox, Patrick J. Biggs, and Nigel P. French. 2014. "Semi-automatic selection of summary statistics for ABC model choice." *Statistical Applications in Genetics and Molecular Biology* 13 (1): 67–82. https://doi.org/10.1515/sagmb-2013-0012.

Pritchard, Jonathan K., Mark T. Seielstad, Anna Perez-Lezaun, and Marcus W. Feldman. 1999. "Population growth of human Y chromosomes: A study of y chromosome microsatellites." *Molecular Biology and Evolution* 16 (12): 1791–8. https://doi.org/10.1093/oxfordjournals.molbev.a026091.

Railsback, Steven F, and Volker Grimm. 2012. *Agent-based and individual-based modeling: a practical introduction*. Princeton University Press. https://books.google.co.uk/books?hl=en%7B/&%7Dlr=%7B/&%7Did=tSI2DkMtoWQC%7B/&%7Doi=fnd%7B/&%7Dpg=PP1%7B/&%7Ddq=Agent-Based+and+Individual-Based+Modeling:+A+Practical+Introduction%7B/&%7Dots=dR3Y0A9NQL%7B/&%7Dsig=aHHw34ZN0K4pnP1zzJ20XBOJ26s%7B/&%7Dredir%7B/_%7Desc=y%7B/#%7Dv=onepage%7B/&%7Dq=Agent-Based%20and%20Individual-Based%20Modelin.

Raynal, Louis, Jean-Michel Marin, Pierre Pudlo, Mathieu Ribatet, Christian P Robert, and Arnaud Estoup. 2018. "ABC random forests for Bayesian parameter inference." https://doi.org/10.24072/pci.evolbiol.100036.

Rubio, F. J., and Adam M. Johansen. 2013. "A simple approach to maximum intractable likelihood estimation." *Electronic Journal of Statistics* 7 (1): 1632–54. https://doi.org/10.1214/13-EJS819.

Salle, Isabelle, and Murat Yıldızoğlu. 2014. "Efficient sampling and meta-modeling for computational economic models." *Computational Economics* 44 (4): 507–36. https://doi.org/10.1007/s10614-013-9406-7.

Snoek, Jasper, Hugo Larochelle, and Ryan P Adams. 2012. "Practical bayesian optimization of machine learning algorithms." In *Advances in Neural Information Processing Systems*, 1–12. http://arxiv.org/abs/1206.2944.

Stow, Craig A., Jason Jolliff, Dennis J. McGillicuddy, Scott C. Doney, J. Icarus Allen, Marjorie A.M. Friedrichs, Kenneth A. Rose, and Philip Wallhead. 2009. "Skill assessment for coupled biological/physical models of marine systems." *Journal of Marine Systems* 76 (1-2): 4–15. https://doi.org/10.1016/j.jmarsys.2008.03.011.

Thiele, Jan C., Winfried Kurth, and Volker Grimm. 2014. "Facilitating parameter estimation and sensitivity analysis of agent-based models: A cookbook using NetLogo and R." *Journal of Artificial Societies and Social Simulation* 17 (3): 11. https://doi.org/10.18564/jasss.2503.

Toni, Tina, David Welch, Natalja Strelkowa, Andreas Ipsen, and Michael P.H Stumpf. 2009. "Approximate Bayesian computation scheme for parameter inference and model selection in dynamical systems." *Journal of the Royal Society Interface* 6 (31): 187–202. https://doi.org/10.1098/rsif.2008.0172.

Vaart, Elske van der, Mark A. Beaumont, Alice S A Johnston, and Richard M. Sibly. 2015. "Calibration and evaluation of individual-based models using Approximate Bayesian Computation." *Ecological Modelling* 312: 182–90. https://doi.org/10.1016/j.ecolmodel.2015.05.020.

Wager, Stefan, Trevor Hastie, and Bradley Efron. 2014. "Confidence Intervals for Random Forests: The Jackknife and the Infinitesimal Jackknife." Vol. 15. http://jmlr.org/papers/volume15/wager14a/wager14a.pdf.

Wasserman, Larry. 2006. *All of nonparametric statistics.* Springer. https://books.google.co.uk/books?hl=en%7B/&%7Dlr=%7B/&%7Did=MRFlzQfRg7UC%7B/&%7Doi=fnd%7B/&%7Dpg=PA2%7B/&%7Ddq=All+of+Nonparametric+Statistics%7B/&%7Dots=SPQStc52Hx%7B/&%7Dsig=zUCTsE2QUPlGrvkyV2QZxKnQxGU%7B/&%7Dredir%7B/_%7Desc=y%7B/#%7Dv=onepage%7B/&%7Dq=All%20of%20Nonparametric%20Statistics%7B/&%7Df=false.

Wilkinson, Richard. 2013. "Approximate Bayesian Computation (ABC) NIPS Tutorial." http://media.nips.cc/Conferences/2013/Video/Tutorial2B.pdf.

Wolpert, David H. 2011. "The Supervised Learning No-Free-Lunch Theorems." In *Soft Computing and Industry*, 25–42. https://doi.org/10.1007/978-1-4471-0123-9_3.

Wolpert, David H., and William G. Macready. 1995. "No free lunch theorems for search." Santa Fe Institute. https://doi.org/10.1145/1389095.1389254.

Wood, Simon N. 2010. "Statistical inference for noisy nonlinear ecological dynamic systems." *Nature* 466 (7310): 1102–4. https://doi.org/10.1038/nature09319.

Wood, Simon N. 2004. "Stable and Efficient Multiple Smoothing Parameter Estimation for Generalized Additive Models." *Journal of the American Statistical Association* 99 (467): 673–86. http://www.jstor.org/stable/27590439%7B/%%7D5Cnhttp://www.jstor.org.proxy.lib.duke.edu/stable/pdfplus/27590439.pdf?acceptTC=true.

———. 2017. *Generalized Additive Models: An Introduction with R.* 2nd ed. New York, New York, USA: Chapman; Hall/CRC.

Wood, Simon N., and Nicole H. Augustin. 2002. "GAMs with integrated model selection using penalized regression splines and applications to environmental modelling." *Ecological Modelling* 157 (2-3): 157–77. https://doi.org/10.1016/S0304-3800(02)00193-X.

Wright, Marvin N., and Andreas Ziegler. 2015. "ranger: A Fast Implementation of Random Forests for High Dimensional Data in C++ and R." *Journal of Statistical Software* 77 (1). https://doi.org/10.18637/jss.v077.i01.