— Supplementary Information —
ABC random forests for Bayesian parameter inference

Louis Raynal[*1], Jean-Michel Marin[†1,2], Pierre Pudlo[3], Mathieu Ribatet[1], Christian P. Robert[4,5], and Arnaud Estoup[2,6]

[1]IMAG, Univ Montpellier, CNRS, Montpellier, France,
[2]IBC, Univ Montpellier, CNRS, Montpellier, France,
[3]Institut de Mathématiques de Marseille, Aix-Marseille Université, France,
[4]Université Paris Dauphine, PSL Research University, Paris, France,
[5]Department of Statistics, University of Warwick, Coventry, UK,
[6]CBGP, INRA, CIRAD, IRD, Montpellier SupAgro, Univ Montpellier, Montpellier, France.

[*]louis.raynal@umontpellier.fr
[†]jean-michel.marin@umontpellier.fr

Table of contents

1 Normal toy example

We consider the hierarchical Normal mean model

$$\begin{aligned} y_i \mid \theta_1, \theta_2 &\sim \mathcal{N}(\theta_1, \theta_2), \\ \theta_1 \mid \theta_2 &\sim \mathcal{N}(0, \theta_2), \\ \theta_2 &\sim I\mathcal{G}(4, 3), \end{aligned}$$

where $I\mathcal{G}(\kappa, \lambda)$ denotes an inverse Gamma distribution with shape parameter κ and scale parameter λ. Let $y = (y_1, \ldots, y_n)$ be a n-sample from the above model. Given these conjugate distributions, the marginal posterior distributions of the parameters θ_1 and θ_2 are closed-forms:

$$\begin{aligned} \theta_1 \mid y &\sim \mathcal{T}\left(n+8, (n\bar{y})/(n+1), (s^2+6)/((n+1)(n+8))\right), \\ \theta_2 \mid y &\sim I\mathcal{G}\left(n/2+4, s^2/2+3\right), \end{aligned}$$

where $\bar{y}$ is the sample mean and $s^2 = \sum_{i=1}^{n}(y_i - \bar{y})^2$ the sum of squared deviations. $\mathcal{T}(\nu, a, b)$ denotes the general t distribution with ν degrees of freedom [7].

From the above expressions and for a given sample y, it is straightforward to derive the exact values of $\mathbb{E}(\theta_1 \mid y)$, $\mathrm{Var}(\theta_1 \mid y)$, $\mathbb{E}(\theta_2 \mid y)$, $\mathrm{Var}(\theta_2 \mid y)$ and posterior quantiles for the two parameters. This provides us with a benchmark on which to assess the performances of ABC-RF. For the present simulation study, we opted for a *reference table* made of $N = 10^4$ replicates of a sample of size $n = 10$ and $k = 61$ summary statistics. Those statistics included the sample mean, the sample variance, the sample median absolute deviation (MAD), all possible sums and products with these three elements resulting in eight new summary statistics and 50 additional independent (pure) noise variables that were generated from a uniform $\mathcal{U}_{[0,1]}$ distribution. The performances of our method were evaluated on an independent test table of size $N_{\text{pred}} = 100$, produced in the same way as the *reference table*. Current ABC methods (rejection, adjusted local linear, ridge and neural network) all depend on the choice of a tolerance level p_ε corresponding to the proportion of selected simulated parameters with lowest distances between simulated and observed summary statistics. On this example we consider a tolerance level of $p_\varepsilon = 0.01$ for ABC with rejection, and $p_\varepsilon = 0.1$ for the ABC methods with adjustment. We also compare estimation results obtained from the adaptive ABC-PMC algorithm described in [10] (Algorithm 5). We implement two designs of this scheme with both 2 000 simulated particles per iteration, 1 000 accepted particles, schemes iterate until we get approximately 10^4 (a-PMC-1) and 10^5 (a-PMC-2) simulated particles. Finally, we carry out additional comparisons with two sequential ABC methods: ABC-PMC based on the algorithm of [1] and ABC-SMC based on the algorithm of [4]. Two different implementations of ABC-PMC (named PMC-1 and PMC-2) are considered. PMC-1 and PMC-2 include 1 000 and 100 simulated particles per iteration, 100

and 10 accepted particles and 10 and 100 iterations, respectively, resulting in 10^4 simulated particles. Two different implementations of ABC-SMC (named SMC-1 and SMC-2) are considered. Both SMC-1 and SMC-2 include 1 000 simulated particles per iteration and a stopping rule based on two pre-computed quantiles of the distances between the observed summary statistics and simulated ones. For SMC-1, we use a quantile of 10% and for SMC-2 a quantile of 1%. At the end, we simulate approximately 40 000 particles for SMC-1 and 360 000 for SMC-2. Furthermore, we also compare with an ABC-SMC scheme (named py-SMC-1 and py-SMC-2) using adaptive population sizes described in [5], using the `pyABC` python module [6]. An initial population size equal to 1 000 is used and two different values of a target density variation. An additional version of ABC-PMC (named py-PMC-1 and py-PMC-2) is implemented using this module to mimic the PMC-1 and PMC-2 designs. In some comparisons, we considere two different situations by including or not a large number of noise variables (50 or 500 noise variables drawn into uniform distributions on [0, 1]) as explanatory variables. The R codes are available at

https://github.com/jmm34/abc-rf-param.

Figure S1 compares the exact values $\psi_1(y) = \mathbb{E}(\theta_1 \mid y)$, $\psi_2(y) = \mathbb{E}(\theta_2 \mid y)$, $\psi_3(y) = \mathrm{Var}(\theta_1 \mid y)$ and $\psi_4(y) = \mathrm{Var}(\theta_2 \mid y)$ with the estimates obtained from the ABC-RF approach. It shows that the proposed estimators have good overall performances for both $\psi_1(y)$ and $\psi_2(y)$, although one can see that $\psi_2(y)$ tends to be slightly overestimated. Our estimators perform less satisfactorily for both $\psi_3(y)$ and $\psi_4(y)$ but remain acceptable. Figure S2 shows furthermore that the quantile estimation are good for θ_1 if less accurate for θ_2.

We then run an experiment to evaluate the precision of the marginal posterior approximation provided by ABC-RF for the parameter θ_1, using two different test datasets and 40 independent *reference tables*. As exhibited in Figure S3, results are mixed. For one dataset, the fit is quite satisfactory, with the RF approximation showing only slightly fatter tails than the true posterior density distribution function (Figure S3; upper panel). For the other dataset, we obtain stronger divergence both in location and precision of the posterior density distribution function (Figure S3; lower panel).

Using the same *reference table*, we now compare our ABC-RF results with a set of five earlier ABC methods, namely, ABC methods based on straightforward rejection, adjusted local linear, ridge regression, adjusted neural networks and adaptive ABC-PMC. Table S1 shows that the ABC-RF approach leads to results better than all other ABC methods for all quantities of interest. Expectations and quantiles are noticeably more accurately predicted. Figure S4 compares differences between estimated and true values of the posterior variances $\psi_3(y)$, $\psi_4(y)$. It shows the global underestimation associated with rejection and ABC methods with adjustment, when compared to ABC-RF, the latter only slightly overestimating the posterior variance. The adaptive ABC-SMC method performs very decently for $\psi_4(y)$, however highly overestimates $\psi_3(y)$. Finally, by looking at the width of the boxplots of Figure S4, we deduce that our ABC-RF estimations exhibit a lower estimation variability.

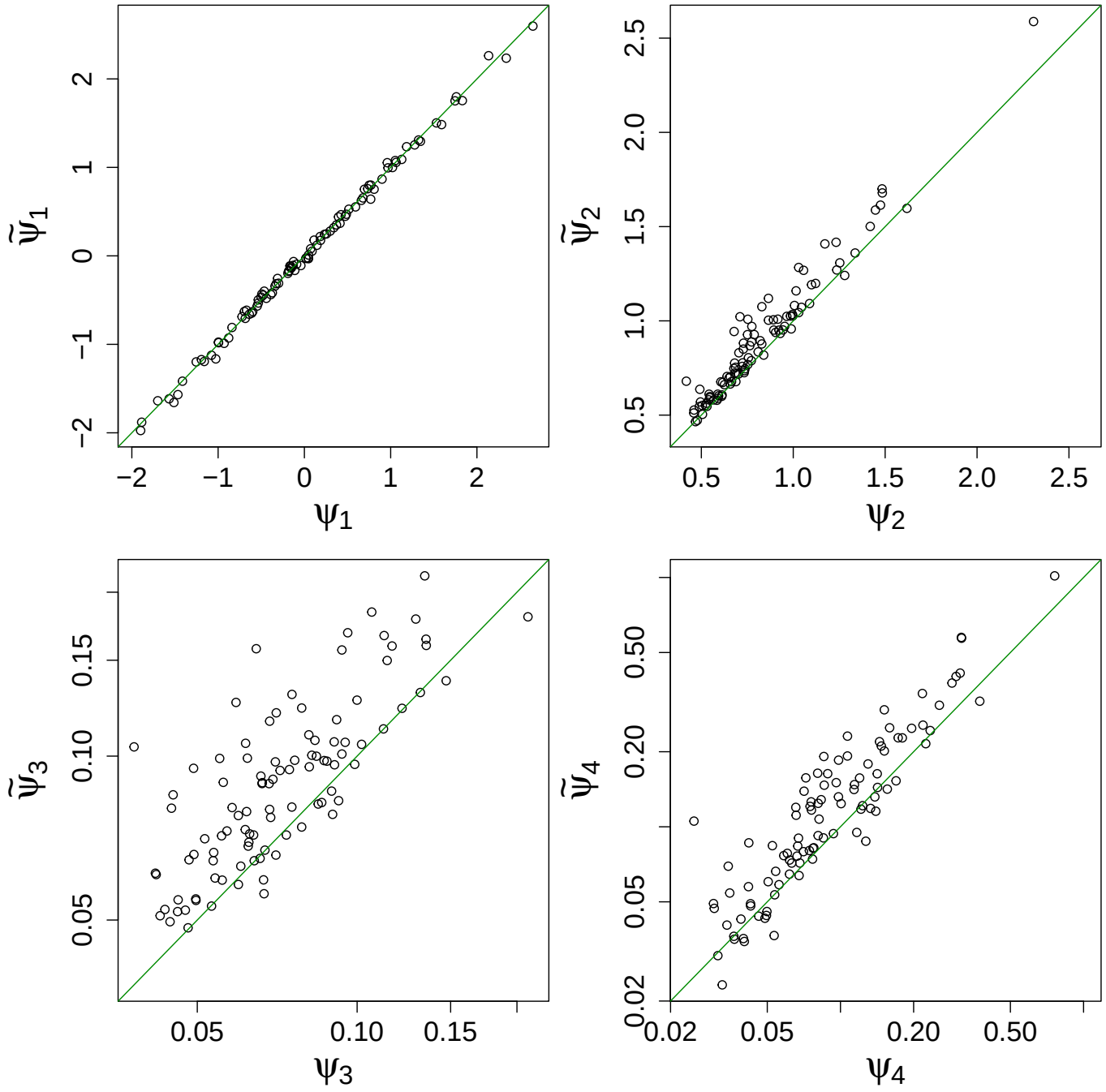

Figure S1: Scatterplot of the theoretical values $\psi_1(y) = \mathbb{E}(\theta_1 \mid y)$, $\psi_2(y) = \mathbb{E}(\theta_2 \mid y)$, $\psi_3(y) = \mathrm{Var}(\theta_1 \mid y)$ and $\psi_4(y) = \mathrm{Var}(\theta_2 \mid y)$ for the Normal model with their corresponding estimates $\tilde{\psi}_1$, $\tilde{\psi}_2$, $\tilde{\psi}_3$, $\tilde{\psi}_4$ obtained using ABC-RF. Variances are represented on a log scale.

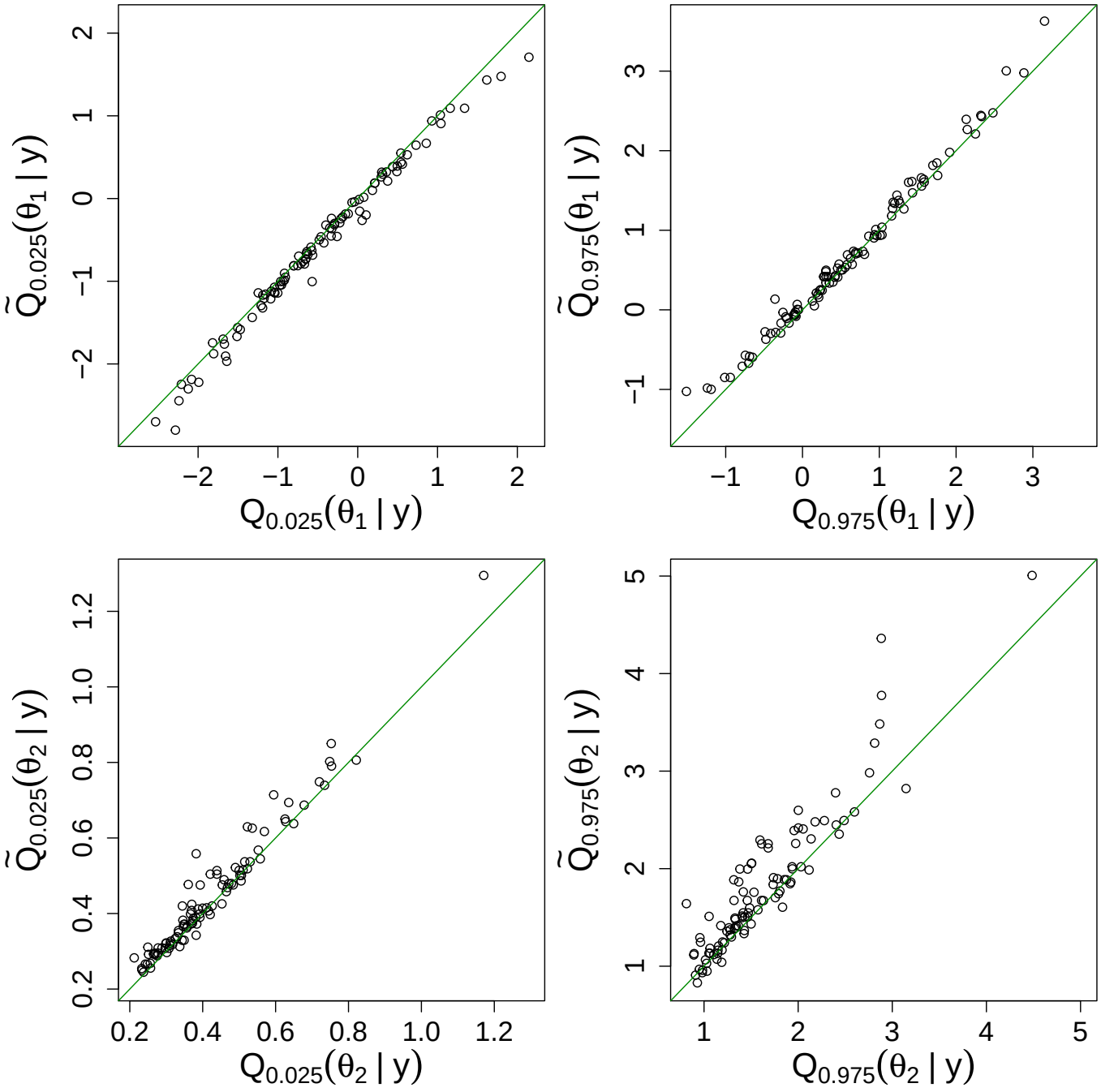

Figure S2: Scatterplot of the theoretical values of 2.5% and 97.5% posterior quantiles for θ_1 and θ_2, for the Normal model with their corresponding estimates obtained using ABC-RF.

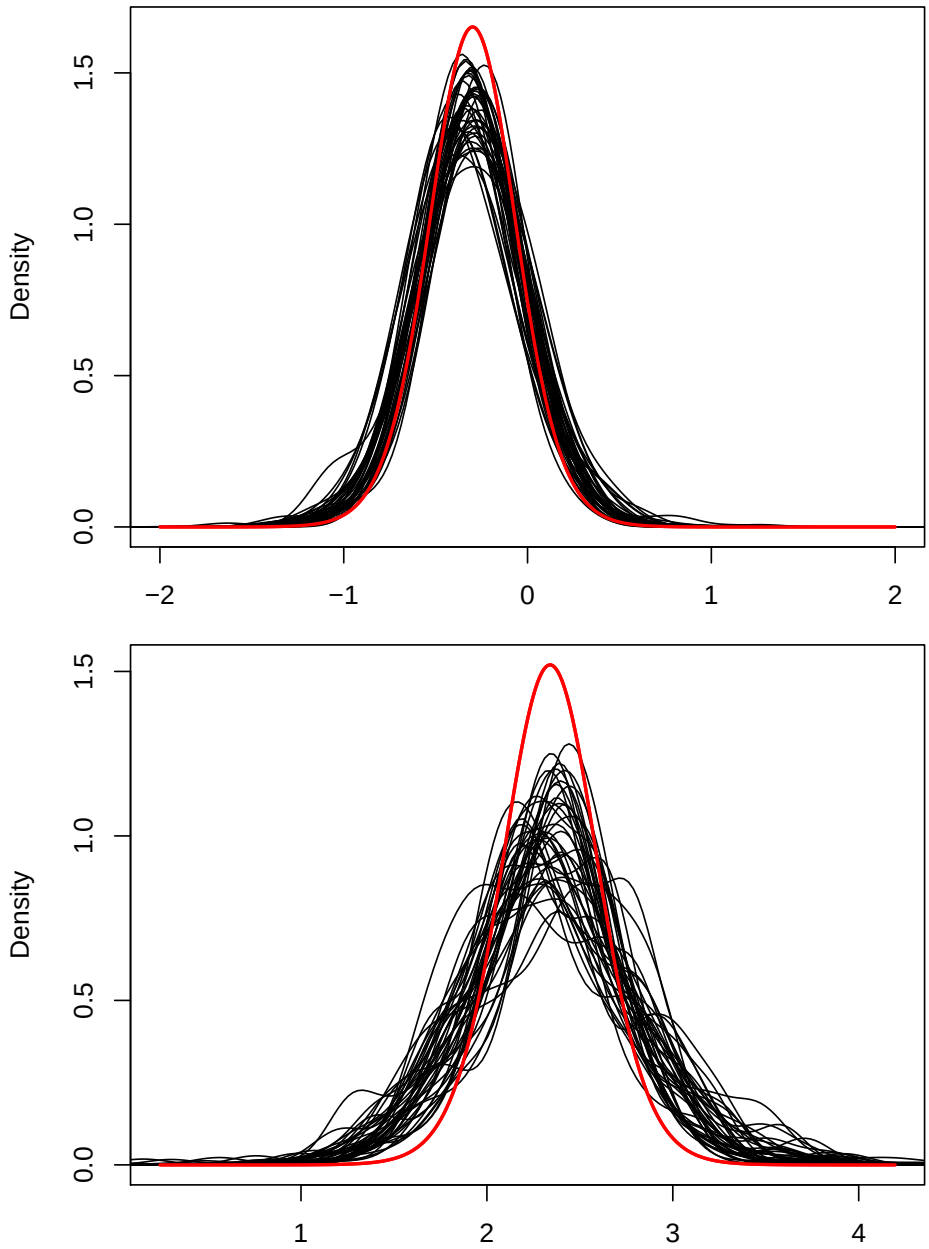

Figure S3: Comparisons of the true posterior density distribution function of θ_1 in the Normal model with a sample of 40 ABC-RF approximations of the posterior density (using RF weights), based on 40 independent *reference tables* and for two different test datasets (upper and lower panels). True posterior densities are represented by red lines.

	RF	Reject	ALL	ARR	ANN	py-SMC-1
$\psi_1(y) = \mathbb{E}(\theta_1 \mid y)$	**0.18**	0.32	0.34	0.31	0.46	0.23
$\psi_2(y) = \mathbb{E}(\theta_2 \mid y)$	**0.10**	0.14	0.19	0.23	0.21	0.11
$\psi_3(y) = \mathrm{Var}(\theta_1 \mid y)$	**0.30**	2.44	0.74	0.70	0.48	0.96
$\psi_4(y) = \mathrm{Var}(\theta_2 \mid y)$	**0.38**	0.59	0.69	0.71	0.97	0.49
$Q_{0.025}(\theta_1 \mid y)$	**0.31**	1.42	0.60	0.71	0.55	0.53
$Q_{0.025}(\theta_2 \mid y)$	**0.07**	0.14	0.42	0.65	1.01	0.24
$Q_{0.975}(\theta_1 \mid y)$	**0.21**	3.34	0.50	0.66	0.51	0.55
$Q_{0.975}(\theta_2 \mid y)$	**0.13**	0.71	0.20	0.20	0.37	0.16

Table S1: Comparison of normalized mean absolute errors (NMAE) of estimated quantities of interest obtained with ABC-RF and other ABC methodologies. RF, Reject, ALL, ARR, ANN and py-SMC-1 stand for random forest (ABC-RF), rejection, adjusted local linear, adjusted ridge regression, adjusted neural network methods and ABC-SMC with adaptive population size from [5], respectively. The smallest NMAE values are in by bold characters.

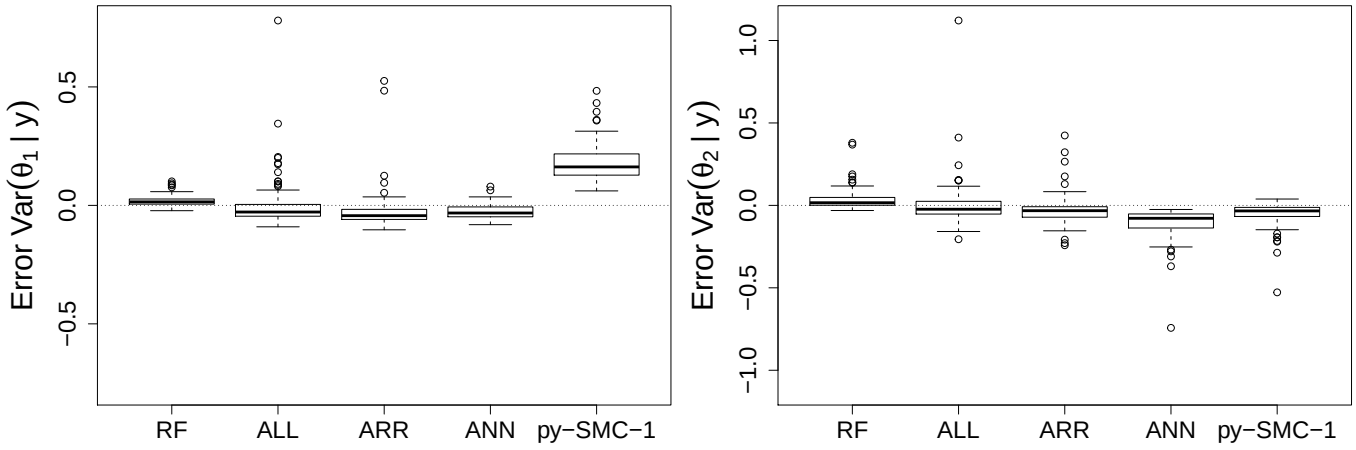

Figure S4: Boxplot comparison of the differences between our predictions for $\mathrm{Var}(\theta_1 \mid y)$ and $\mathrm{Var}(\theta_2 \mid y)$ and the corresponding true values, using ABC-RF and other ABC methods. RF, ALL, ARR and ANN notations as in the legend of Table S1). py-SMC-1 refers to the adaptive ABC-SMC algorithm of [5] with the same tuning parameters as in Table S1. The closer to the $y = 0$ axis, the better the predictions. Boxplots above this axis imply overestimation of the predictions and below underestimation.

	RF	a-PMC-1	a-PMC-2	PMC-1	PMC-2	SMC-1	SMC-2	py-PMC-1	py-PMC-2	py-SMC-1	py-SMC-2
$\mathbb{E}(\theta_1 \mid y)$	**0.18**	0.54	0.34	0.48	0.67	0.45	0.49	0.38	0.32	0.23	0.26
$\mathbb{E}(\theta_2 \mid y)$	**0.10**	0.15	0.13	0.17	0.17	0.16	0.15	0.13	0.12	0.11	0.11
$\mathrm{Var}(\theta_1 \mid y)$	**0.30**	3.85	2.23	3.28	2.65	3.32	1.93	2.30	2.36	0.96	1.41
$\mathrm{Var}(\theta_2 \mid y)$	**0.38**	0.55	0.42	0.73	0.65	0.71	0.42	0.47	0.39	0.49	0.42
$Q_{0.025}(\theta_1 \mid y)$	**0.31**	2.45	1.37	2.18	1.12	2.07	1.26	1.11	1.54	0.53	0.71
$Q_{0.025}(\theta_2 \mid y)$	**0.07**	0.17	0.15	0.16	0.29	0.15	0.17	0.23	0.18	0.24	0.19
$Q_{0.975}(\theta_1 \mid y)$	**0.21**	2.08	1.29	1.63	0.85	1.73	1.08	1.29	1.41	0.55	0.69
$Q_{0.975}(\theta_2 \mid y)$	**0.13**	0.18	0.15	0.19	0.24	0.24	0.19	0.18	0.13	0.16	0.13

Table S2: Comparison of Normalized Mean Absolute Errors (NMAE) obtained with ABC-RF, adaptive ABC-PMC and sequential ABC methods for various tuning parameters based on 100 test datasets. RF stands for the ABC-RF scheme on a reference table of size 10^4. a-PMC-1 and a-PMC-2 stand for two designs of the adaptive ABC-PMC method of [10] (Algorithm 5) with 2 000 simulated particles per iteration, 1 000 accepted particles, schemes iterate until we get approximately 10^4 (a-PMC-1) and 10^5 (a-PMC-2) simulated particles, respectively. PMC-1 and PMC-2 stand for two designs of the sequential ABC-PMC algorithm of [1]: PMC-1 and PMC-2 include 1 000 and 100 simulated particles per iteration, 100 and 10 accepted particles during 10 and 100 iterations, respectively. For PMC-1 and PMC-2, we simulate 10^4 particles. SMC-1 and SMC-2 stand for two designs of the sequential ABC-SMC algorithm of [4]. Both SMC-1 and SMC-2 include 1 000 simulated particles per iteration and a stopping rule based on two pre-computed quantiles of the distances between the observed summary statistics and simulated ones. For SMC-1, we use a quantile of 10% and for SMC-2 a quantile of 1%. We also use the `pyABC` python module [6] to mimic the PMC algorithm as well as a full adaptive SMC method described in [5]. The main drawback of this module is the absence of a stopping rule concerning the maximal number of simulations, thus the total number of simulations is higher than 10 000 (number of simulations used for ABC-RF). py-PMC-1 and py-PMC-2 is a PMC adaptation obtained thanks to this module, using respectively 100 and 1 000 accepted particles per iteration and a stopping rule based on a 1% pre-computed quantile of the distances between observed and simulated data, (22 000 and 133 000 simulations are respectively needed in average). py-SMC-1 and py-SMC-2 designate the `pyABC` all adaptive version of ABC-SMC, including an adaptive population size, with a target density variation parameter respectively equal to 0.15 and 0.1 (see [5] for more details), and with 7 and 5 maximal iterations as stopping rule.

	RF	a-PMC-1	a-PMC-2	PMC-1	PMC-2	SMC-1	SMC-2	py-PMC-1	py-PMC-2	py-SMC-1	py-SMC-2
$\mathbb{E}(\theta_1 \mid y)$	**0.14**	0.83	0.68	0.75	1.13	0.78	0.95	0.68	0.58	0.38	0.42
$\mathbb{E}(\theta_2 \mid y)$	**0.10**	0.21	0.19	0.20	0.24	0.20	0.19	0.15	0.14	0.12	0.13
$\mathrm{Var}(\theta_1 \mid y)$	**0.30**	6.87	5.77	5.64	5.74	5.23	3.45	4.26	4.69	2.59	3.16
$\mathrm{Var}(\theta_2 \mid y)$	**0.44**	0.93	0.74	1.03	0.84	0.75	0.56	0.40	0.32	0.39	0.31
$Q_{0.025}(\theta_1 \mid y)$	**0.29**	4.12	3.48	3.73	2.75	3.33	2.30	2.34	2.83	1.64	1.88
$Q_{0.025}(\theta_2 \mid y)$	**0.06**	0.22	0.21	0.20	0.32	0.20	0.21	0.17	0.14	0.15	0.15
$Q_{0.975}(\theta_1 \mid y)$	**0.24**	3.22	2.87	2.61	1.67	2.71	1.81	2.33	2.42	1.48	1.74
$Q_{0.975}(\theta_2 \mid y)$	**0.14**	0.28	0.23	0.25	0.23	0.24	0.21	0.18	0.15	0.17	0.15

Table S3: Same as in Table S2, except that 500 independent noise summary statistics were added as explanatory variables instead of 50. Noise variables were simulated by randomly drawing into uniform distributions on $[0, 1]$. There are thus 11+500 = 511 summary statistics in total (versus 61 summary statistics in Table S2).

We provide further comparisons obtained using ABC-RF, adaptive ABC-PMC and sequential ABC methods for various tuning parameters (Tables S2 and S3). We considered two different situations by including 50 (Table S2) or 500 (Table S3) noise variables (drawn into uniform distributions on [0, 1]) as explanatory variables. We found that the ABC-RF algorithm outperforms all adaptive and sequential methods (and designs) considered, and this adding or not an high amount of noise variables. Note that, in contrast to other methods, ABC-RF was only weakly affected by the presence of a large number of noise variables.

Details on tuning parameters used for ABC sequential methods:

- For all non-adaptive ABC methods, we use an Euclidean distance normalised by the MAD calculated on a precomputed reference table of size 10 000. For adaptive methods, a similar distance is used where the standardisation is performed with iteratively updated MAD values.
- Concerning the transition kernel, a Gaussian distribution is considered. The variance-covariance matrix is taken as the weighted empirical one computed on the accepted parameters of the previous algorithm iteration, multiplied by 2 for the methods we implemented (a-PMC, PMC and SMC) and multiplied by a scaling factor involving the Silverman's rule of thumb (see [5] and [6] for more details) for the remaining ones using `pyABC`.
- For the ABC-SMC algorithm [4], we do not change the tuning parameters described in the original paper (i.e. $\alpha = 0.90$, $N_T = N/2$ where N is the population size.)
- The adaptive ABC-SMC algorithm of [10] requires a value (also denoted α) indicating the proportion of accepted simulations per iteration, here chosen at 0.5.
- To mimic the ABC-PMC strategy we use `pyABC` with an adaptive threshold equal to the median of the previous iteration's distances, with a constant population size (100 or 1 000) and with a minimum threshold value equal to the 1% quantile of a precomputed reference table of size 10 000.
- Finally, for the full adaptive method of [5], we use an adaptive population size depending on a desired target density variation value (E_{cv}) equal to 0.15 or 0.1 in our experimentations and an initial population size equal to 1 000. Note that the default value 0.05 induced a change in the population size from 1 000 to 10 000 in only one iteration, hence that is not relevant in our comparison with ABC-RF due to its high simulation cost. The threshold is taken as the median of the previous iteration's distances (as in [5]). We do not use a minimum threshold value but a maximal number of iteration equal to 7 and 5 respectively. Note that this method requires about 24 000 and 28 000 simulations when 50 noise summary statistics are considered, and about 40 000 and 41 000 when 500 is added.

2 Comparing three methods of variance estimation of parameters

We here compare three methods to estimate posterior variance of a parameter transformation of interest τ using ABC-RF. Two of them have already been explained in the main text (i.e. method 1 and 3 below).

- **Method 1**: One reuses the original random forest (RF) weights $w_t(\eta(y))$ to the out-of-bag square residuals $(\tau^{(t)} - \hat{\tau}^{(t)}_{\text{oob}})^2$, giving the variance estimator

$$\widetilde{\text{Var}}(\tau \mid \eta(y)) = \sum_{t=1}^{N} w_t(\eta(y))(\tau^{(t)} - \hat{\tau}^{(t)}_{\text{oob}})^2.$$

- **Method 2**: A similar estimator can be obtained by building a new RF thanks to the training sample $(\eta(y^{(t)}), \left(\tau^{(t)} - \tau_{\text{oob}})^2\right)_{t=1,\ldots,N}$, resulting in the estimator

$$\text{Var}^{\#}(\tau \mid \eta(y)) = \sum_{t=1}^{N} \widetilde{w}_t(\eta(y))(\tau^{(t)} - \hat{\tau}^{(t)}_{\text{oob}})^2,$$

  where $\widetilde{w}_t(\eta(y))$ is the computed weights of this newly trained RF. This estimator is based on the expression of the posterior variance as a conditional expectation:

$$\text{Var}(\tau \mid \eta(y)) = \mathbb{E}\left([\tau - \mathbb{E}(\tau \mid \eta(y))]^2 \mid \eta(y)\right)$$

  and the fact that such a RF is able to estimate this posterior expectation. This approach is more expensive due to the additional RF requirement.

- **Method 3**: The variance estimator is based on the cumulative distribution function (cdf) approximation,

$$\widehat{\text{Var}}(\tau \mid \eta(y)) = \sum_{t=1}^{N} w_t(\eta(y)) \left(\tau^{(t)} - \sum_{u=1}^{N} w_u(\eta(y))\tau^{(u)}\right)^2.$$

We here compare these three estimators on the Normal toy example detailed above with h the projection on both coordinates of the parameter vector θ (see Section 1 of the supplementary material). We find that the three estimators behave similarly; boxplots are alike and all tend to overestimate posterior variances (Figure S5). Results summarized in Table S4 also supports this similarity in NMAE terms. Because the estimator 1 appears to show slightly lower errors for both parameter θ_1 and θ_2, we decided to use it in the two examples detailed in the main text.

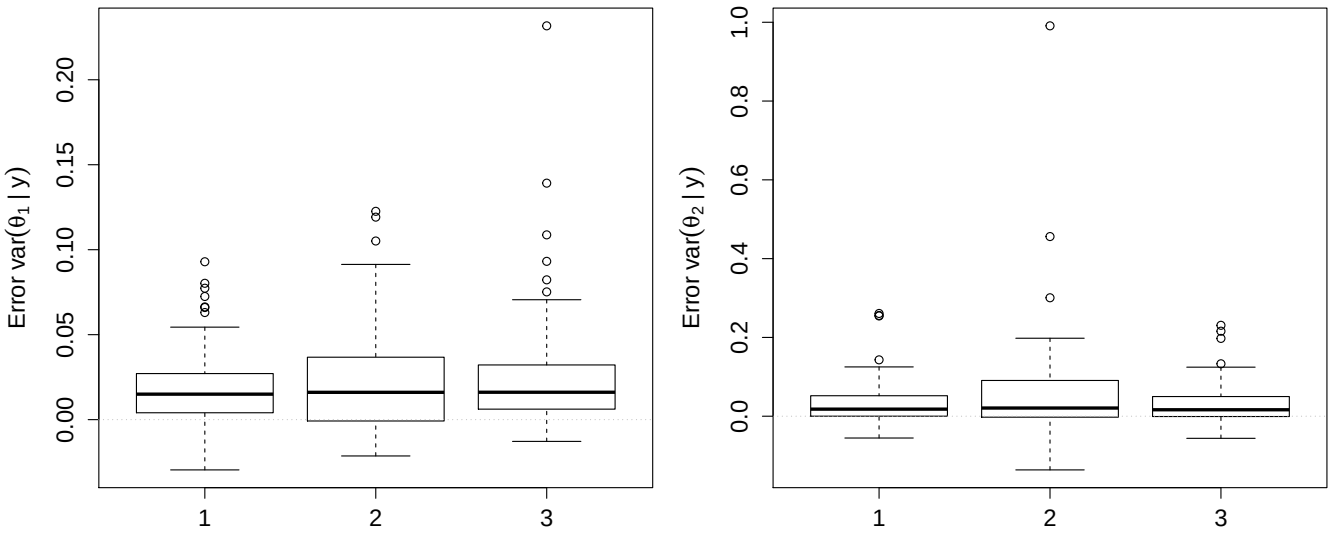

Figure S5: Boxplot comparison of differences between our predictions for $\mathrm{Var}(\theta_i \mid y)$ and the true values with the three methods of variance estimation: by reusing weights (method 1, boxplot 1), by building a new RF on square residuals (method 2, boxplot 2) and by using the estimation of the cumulative distribution function (method 3, boxplot 3).

Method	1	2	3
$\mathrm{Var}(\theta_1 \mid y)$	**0.30**	0.35	0.39
$\mathrm{Var}(\theta_2 \mid y)$	0.38	0.63	**0.37**

Table S4: Comparison of normalized mean absolute errors (NMAE) of estimate variances when using three methods, (see legend of Figure S5 and Section 2 of the supplementary material). The smallest NMAE values are in bold characters.

3 Study of covariances of parameters using random forests

3.1 Methodology

We are here interested in another estimate that is frequently produced in a Bayesian analysis, that is the posterior covariance between two univariate transforms of the parameter, $\tau = h(\theta)$ and $\sigma = g(\theta)$ say, $\mathrm{Cov}(\tau, \sigma \mid \eta(y))$. Since we cannot derive this quantity from the approximations to the marginal posteriors of τ and σ, we propose to construct a specific RF for this purpose. With the same notations used in the main text, we denote approximations of posterior expectations for τ and σ, produced by out-of-bag informations, by $\hat{\tau}^{(t)}_{\mathrm{oob}}$ and $\hat{\sigma}^{(t)}_{\mathrm{oob}}$. We use the product of out-of-bag errors for τ and σ in the empirical covariance, and consider $(\tau^{(t)} - \hat{\tau}^{(t)}_{\mathrm{oob}})(\sigma^{(t)} - \hat{\sigma}^{(t)}_{\mathrm{oob}})$ as the response variable. With the previously introduced notations, the corresponding RF estimator is

$$\widetilde{\mathrm{Cov}}(\tau, \sigma \mid \eta(y)) = \frac{1}{B}\sum_{b=1}^{B} \frac{1}{\left|L_b(\eta(y))\right|} \sum_{t:\eta(y^{(t)})\in L_b} n_b^{(t)} (\tau^{(t)} - \hat{\tau}^{(t)}_{\mathrm{oob}})(\sigma^{(t)} - \hat{\sigma}^{(t)}_{\mathrm{oob}}).$$

This posterior covariance approximation requires a total of three regression RFs: one for each parameters and one for the covariance approximation.

3.2 Toy regression example

We now apply our RF methodology to a toy regression example for which its non-zero covariance between parameters is the main quantity of interest, hence we consider the case were g and h are the projections on a given coordinate of the parameter vector θ. For a simulated $n \times 2$ design matrix $X = [x_1, x_2]$, we consider the Zellner's hierarchical model [7, chapter 3]

$$\begin{aligned}(y_1, \ldots, y_n) \mid \beta_1, \beta_2, \sigma^2 &\sim \mathscr{N}_n(X\beta, \sigma^2 Id),\\ \beta_1, \beta_2 \mid \sigma^2 &\sim \mathscr{N}_2(0, n\sigma^2 (X^\top X)^{-1}),\\ \sigma^2 &\sim I\mathscr{G}(4,3),\end{aligned}$$

where $\mathscr{N}_k(\mu, \Sigma)$ denotes the multivariate normal distribution of dimension k with mean vector μ and covariance matrix Σ, and $I\mathscr{G}(\kappa, \lambda)$ an inverse Gamma distribution with shape parameter κ and scale parameter λ. Provided $X^\top X$ is invertible, this conjugate model leads to closed-form marginal posteriors [7]

$$\begin{aligned}\beta_1, \beta_2 \mid y &\sim \mathscr{T}_2\left(\frac{n}{n+1}(X^\top X)^{-1}X^\top y, \frac{3 + y^\top (Id - X(X^\top X)^{-1}X^\top)y/2}{4 + n/2}\frac{n}{n+1}(X^\top X)^{-1}, 8+n\right),\\ \sigma^2 \mid y &\sim I\mathscr{G}\left(4 + \frac{n}{2}, 3 + \frac{1}{2}y^\top (Id - X(X^\top X)^{-1}X^\top)y\right),\end{aligned}$$

where $\mathscr{T}_k(\mu,\Sigma,\nu)$ is the multivariate Student distribution of dimension k, with location parameter μ, scale matrix Σ and degree of freedom ν.

In our simulation experiment, we concentrate on the non zero covariance of the posterior distribution namely $\mathrm{Cov}(\beta_1,\beta_2 \mid y)$. A *reference table* of $N = 10\,000$ replicates of a n-sample with $n = 100$ is generated. We then create $k = 60$ summary statistics: the maximum likelihood estimates of β_1, β_2, the residual sum of squares, the empirical covariance and correlation between y and x_1, covariance and correlation between y and x_2, the sample mean, the sample variance, the sample median, and 50 independent noise variables simulated from a uniform distribution $\mathscr{U}_{[0,1]}$. These noise variables were introduced to be in a sparse context.

Similarly to the Normal example of the main text, we assess the performance of our approach using an independent (Monte Carlo) test dataset of size $N_{\text{pred}} = 100$ and compare estimation accuracy with the ABC-RF approach from the ones with adjusted ridge regression and neural network ABC methodologies. RF are once again built with $B = 500$ trees, $n_{try} = k/3$ and minimum node size equals to 5 and ABC methods rely on the R package `abc` with a tolerance parameter equals to 0.1 for ABC methods with adjustment. ABC with neural network adjustment require the specification of the number of layers composing the neural network. We use again 10 layers, the default number of layers in the R package `abc`. For local linear or ridge regression the corrections are univariate. That is not the case for neural networks which, by construction, perform multivariate correction.

Covariance estimation is a novel feature in this example, Table S5 shows that the ABC-RF approach does better in NMAE terms. As exhibited in Figure S6, ABC-RF overestimates covariances when earlier ABC methods underestimate it. Results are quite encouraging even though we believe the method might still be improved.

	RF	ARR	ANN
$\mathrm{Cov}(\beta_1, \beta_2 \mid y)$	**0.26**	0.85	0.64

Table S5: Comparison of normalized mean absolute errors (NMAE) of estimate posterior covariances between β_1 and β_2 using random forest (RF), adjusted ridge regression (ARR) and adjusted neural network (ANN) ABC methods. The smallest NMAE value is in bold characters.

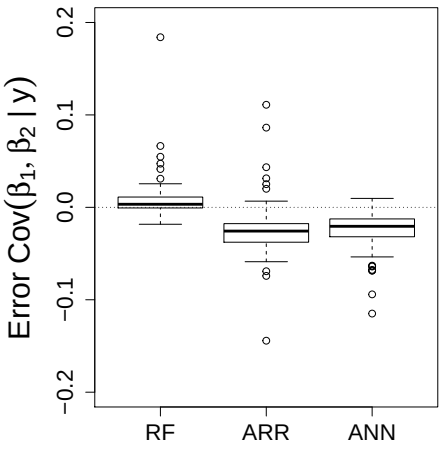

Figure S6: Boxplot comparison of differences between prediction and true values for $\mathrm{Cov}(\beta_1, \beta_2 \mid y)$ using random forest (RF), adjusted ridge regression (ARR) and adjusted neural network (ANN) ABC methods.

4 A basic R code to use the abcrf package version 1.7.1

We provide some basic R lines of code to use the R package abcrf and conduct RF inference about parameters. There are two possibilities to read simulated data: the user wants to use a *reference table* simulated from the software DIYABC v.2.1.0 [3] recorded within a .bin file associated with its .txt header file, or the simulated *reference table* is only contained within a .txt file. Of course if the model is simple enough, the user can simulate the *reference table* himself using its own simulator program. In the following, we assume θ is a vector of p parameters and k summary statistics are considered. The # symbol means the text on its right is a comment and ignored by R. We here focus on a single parameter of interest labelled "poi".

Installing and loading the R package abcrf

```
install.packages("abcrf") # To install the abcrf package (version 1.7.1)
library(abcrf) # To load the package.
```

Reading data: option 1 - using a .bin and .text files obtained using DIYABC

We assume the *reference table* is recorded within the reftable.bin file and its corresponding header in the header.txt file. The function readRefTable is used to recover the data.

```
data <- readRefTable(filename = "reftable.bin", header = "header.txt")
# data is a list containing the scenarios (or models) indices, the matrix
# with the parameters, the summary statistics and other informations.

# We are here interested in the simulated data of the scenario 1.
index1 <- data$scenarios == 1 # To store the model 1 indexes.

# We then create a data frame composed of the parameter of interest poi and
# the summary statistics of the scenario 1.
data.poi <- data.frame(poi = data$params[index1, "poi"],
                       data$stats[index1, ])
```

Reading data: option 2 - using a `.txt` file

We assume that the *reference table* is recorded within `yourTxtFile.txt` file, composed of a first column corresponding to the scenario indices, `p` columns of parameters and `k` columns of summary statistics, the first row is the column labels. The field separator character being a white space.

```
data <- read.table(file = "youTxtFile.txt", header = TRUE, sep = "")
# data is a matrix. The first column is the model indices, the next p are
# the p parameters, the last k are the summary statistics.

index1 <- data[ , 1] == 1 # To store the model 1 indexes.

# We then create a data frame composed of the parameter of interest poi and
# the summary statistics of model 1. p and k have to be defined.
data.poi <- data.frame(poi = data[index1, "poi"],
                       data[index1, (p+2):(p+k+1)])
```

Subsetting your dataset

If required, subsetting your datasets stored in `data.poi` can be easily done with the following line.

```
data.poi <- data.poi[1:10000, ]
# If you are interest in the 10000 first datasets.
```

Training a random forest

The random forest of the ABC-RF method is built thanks to the `regAbcrf` function, its principle arguments being a R formula and the corresponding data frame as training dataset. Additional arguments are available, especially the number of trees (`ntree`, with default values `ntree` = 500), the minimum node size (`min.node.size`, with default value `min.node.size` = 5), and the number of covariates randomly considered at each split (`mtry`). See the `regAbcrf` help for further details.

```
model.poi <- regAbcrf(formula = poi~., data = data.poi, ntree = 500,
                      min.node.size = 5, paral = TRUE)
# The used formula means that we are interested in explaining the parameter
# poi thanks to all the remaining columns of data.poi (i.e. all the
# summary statistics).
# The paral argument determine if parallel computing will be activated
# or not.
```

Graphical representations to access the performance of the method

The evolution of the out-of-bag mean squared error depending on the number of tree can be easily represented with the `err.regAbcrf` function (e.g. Figure 6 of the main text).

```
errorOOB <- err.regAbcrf(object = model.poi, training = data.poi,
                         paral = TRUE)
```

The contribution of summary statistics in ABC-RF estimation for the parameter of interest can be retrieved with the `plot` function applied to an object resulting from `regAbcrf`.

```
plot(x = model.poi, n.var = 25)
# The contributions of the 25 most important summary statistics are
# represented (e.g. Figure S5).
```

Making predictions

Finally, given a data frame `obs.poi` containing the summary statistics you want the predictions of posterior quantities of interest for a given dataset (usually the observed dataset). When using DIYABC, note that the summary statistics of the observed dataset are recorded in a file name `statobs.txt`. The `predict` method can be used for this purpose. The column names need to be the same than those in the summary statistics of `data.poi`.

```
# Reading the observed dataset with
obs.poi <- read.table("statobs.txt", skip=2)
# If obs.poi is not a dataframe or the column names do not match,
```

```
# you can use the following lines:
obs.poi <- as.data.frame(obs.poi)
colnames(obs.poi) <- colnames(data.poi[ ,-1])

# Prediction is complete by
pred.obsPoi <- predict(object = model.poi, obs = obs.poi,
                       training = data.poi, quantiles = c(0.025,0.975),
                       paral = TRUE)
# The 2.5 and 97.5 order quantiles are computed by specifying
# quantiles = c(0.025,0.975).

# pred.obsPoi is a list containing predictions of interest.

# Posterior mean can be retrieved by
pred.obsPoi$expectation

# Posterior variance by
pred.obsPoi$variance

# Posterior quantiles by
pred.obsPoi$quantiles
```

A graphical representation of the approximate posterior density of `poi` given `obs.poi` can be obtained using the `densityPlot` function.

```
densityPlot(object = model.poi, obs = obs.poi, training = data.poi, paral = TRUE)
```

5 Summary statistics available in the software DIYABC v.2.1.0 for SNP data

For single nucleotide polymorphic (SNP) markers, the program DIYABC v.2.1.0 [3] proposes a set of summary statistics among those used by population geneticists. These summary statistics are mean values, variance values and proportion of null values across loci, which allow a rough description of the allelic spectrum. Such summary statistics characterize a single, a pair or a trio of population samples.

Single population statistics

`HPO_i`: proportion of monomorphic loci for population i
`HM1_i`: mean gene diversity across polymorphic loci [9]
`HV1_i`: variance of gene diversity across polymorphic loci
`HMO_i`: mean gene diversity across all loci [9]

Two population statistics

`FPO_i&j`: proportion of loci with null FST distance between the two samples for populations i and j [12]
`FM1_i&j`: mean across loci of non null FST distances
`FV1_i&j`: variance across loci of non null FST distances
`FMO_i&j`: mean across loci of FST distances [12]
`NPO_i&j`: proportion of 1 loci with null Nei's distance [8]
`NM1_i&j`: mean across loci of non null Nei's distances
`NV1_i&j`: variance across loci of non null Nei's distances
`NMO_i&j`: mean across loci of Nei's distances [8]

Three population statistics

`APO_i_j&k`: proportion of loci with null admixture estimate when pop. i comes from an admixture between j and k
`AM1_i_j&k`: mean across loci of non null admixture estimate
`AV1_i_j&k`: variance across loci of non null admixture estimated
`AMO_i_j&k`: mean across all locus admixture estimates [2]

6 Supplementary figures and tables for the Human population genetics example

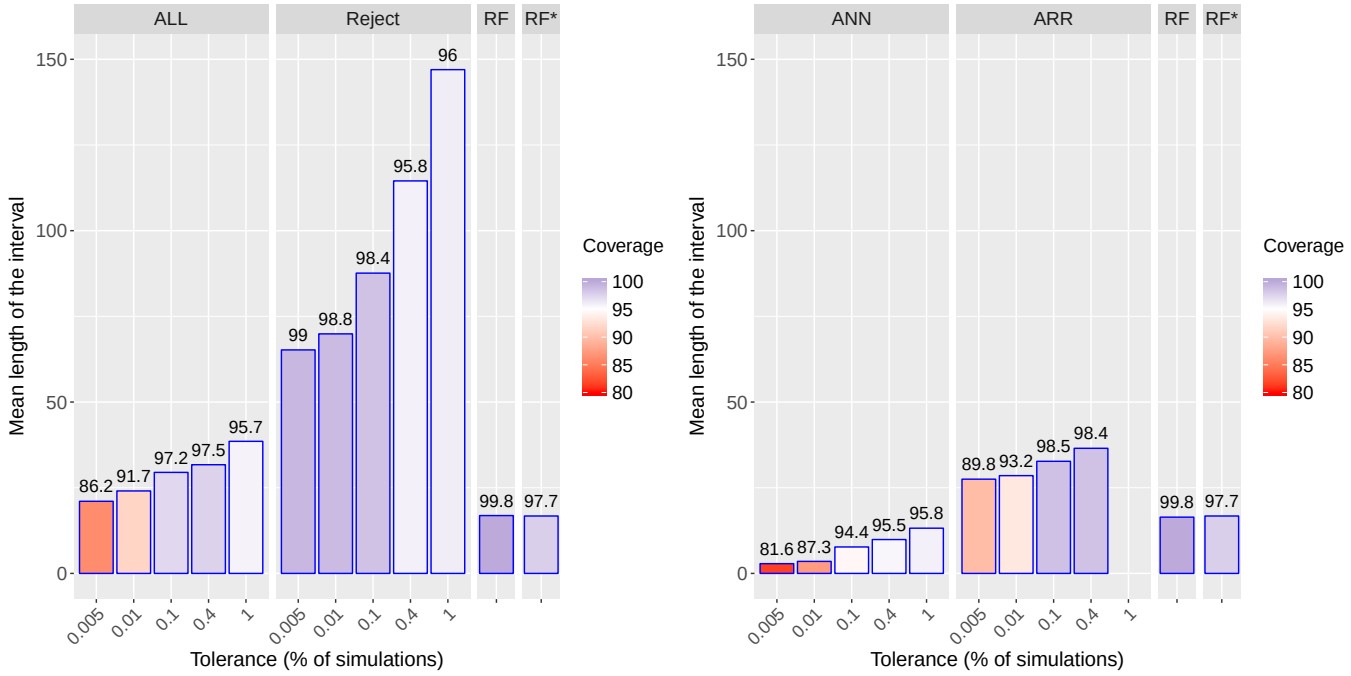

Figure S7: Range and coverage comparison of approximate 95% credible intervals on the ratio N2/Na of the Human population genetics example, obtained with ABC-RF (RF) and with earlier ABC methods : rejection (Reject), adjusted local linear (ALL) or ridge regression (ARR) or neural network (ANN) with various tolerance levels for Reject, ALL, ARR and ANN. Coverages values are specified by bar colors and superimposed values. Heights indicate CI mean lengths. Na is the ancestral African effective population size before the population size change event and N2 the African effective population size after the population size change event (going backward in time). RF* refers to results obtained using ABC-RF when adding 20 additional independent noise variables generated from a uniform $\mathscr{U}_{[0,1]}$ distribution. RF refers to results without noise variables.

Method	Tolerance level	ra NMAE	N2/Na NMAE
RF	NA	0.018	0.053
RF*	NA	0.019	0.053
Reject	0.005	0.151	0.355
Reject	0.01	0.178	0.454
Reject	0.1	0.322	1.223
Reject	0.4	0.574	2.025
Reject	1	0.856	4.108
ALL	0.005	0.028	0.166
ALL	0.01	0.028	0.249
ALL	0.1	0.035	0.139
ALL	0.4	0.044	0.170
ALL	1	0.062	0.209
ARR	0.005	0.027	0.220
ARR	0.01	0.027	0.317
ARR	0.1	0.035	0.140
ARR	0.4	0.044	0.163
ARR	1	–	–
ANN	0.005	**0.007**	**0.037**
ANN	0.01	0.007	0.038
ANN	0.1	0.013	0.064
ANN	0.4	0.016	0.123
ANN	1	0.025	0.095

Table S6: Comparison of normalized mean absolute errors (NMAE) for the estimation of the parameters ra and N2/Na using ABC-RF (RF) and ABC with rejection (Reject), adjusted local linear (ALL) or ridge regression (ARR) or neural network (ANN) with various tolerance levels for Reject, ALL, ARR and ANN. NA stands for not appropriate. The smallest NMAE values are in by bold characters. NA stands for not appropriate. RF* refers to results obtained using ABC-RF when adding 20 additional independent noise variables generated from a uniform $\mathscr{U}_{[0,1]}$ distribution. RF refers to results without noise variables.

ra

Method	Tol. level	Expectation	$Q_{0.025}$	$Q_{0.05}$	$Q_{0.95}$	$Q_{0.975}$
RF	NA	0.221	0.112	0.134	0.279	0.287
RF*	NA	0.225	0.112	0.142	0.282	0.290
Reject	0.005	0.223	0.061	0.069	0.364	0.389
Reject	0.01	0.220	0.060	0.070	0.389	0.418
Reject	0.1	0.276	0.062	0.074	0.511	0.543
Reject	0.4	0.388	0.068	0.086	0.739	0.791
Reject	1	0.502	0.073	0.095	0.906	0.928
ALL	0.005	0.278	0.219	0.229	0.322	0.337
ALL	0.01	0.257	0.232	0.238	0.274	0.278
ALL	0.1	0.207	0.170	0.171	0.233	0.237
ALL	0.4	0.194	0.144	0.152	0.233	0.241
ALL	1	0.196	0.115	0.126	0.278	0.299
ARR	0.005	0.260	0.252	0.254	0.265	0.266
ARR	0.01	0.252	0.239	0.242	0.260	0.262
ARR	0.1	0.211	0.171	0.178	0.239	0.244
ARR	0.4	0.196	0.140	0.149	0.241	0.251
ARR	1	–	–	–	–	–
ANN	0.005	0.227	0.221	0.223	0.232	0.234
ANN	0.01	0.226	0.219	0.221	0.231	0.233
ANN	0.1	0.228	0.217	0.220	0.236	0.239
ANN	0.4	0.232	0.216	0.221	0.242	0.248
ANN	1	0.206	0.183	0.187	0.227	0.233

Table S7: Estimation of the parameter ra and N2/Na for the observed human population genetics dataset using ABC-RF (RF), and ABC with rejection (Reject), adjusted local linear (ALL) or ridge regression (ARR) or neural network (ANN) with various tolerance levels (Tol. level) for Reject, ALL, ARR and ANN. NA stands for not appropriate. RF* refers to results obtained using ABC-RF when adding 20 additional independent noise variables generated from a uniform $\mathscr{U}_{[0,1]}$ distribution. RF refers to results without noise variables.

N2/Na

Method	Tol. level	Expectation	$Q_{0.025}$	$Q_{0.05}$	$Q_{0.95}$	$Q_{0.975}$
RF	NA	4.508	3.831	3.959	5.153	5.424
RF*	NA	4.594	3.821	3.910	5.241	6.552
Reject	0.005	6.282	2.937	3.223	10.086	11.337
Reject	0.01	6.542	2.746	3.116	10.837	11.852
Reject	0.1	8.001	2.131	2.574	15.690	18.531
Reject	0.4	11.605	1.795	2.331	28.011	38.532
Reject	1	23.483	0.672	1.185	84.649	147.657
ALL	0.005	30.041	1.256	1.879	83.369	174.340
ALL	0.01	9.289	3.946	4.586	16.686	20.361
ALL	0.1	8.235	5.736	5.995	11.573	12.719
ALL	0.4	10.752	4.588	4.996	21.656	27.300
ALL	1	7.222	5.684	5.829	9.631	10.475
ARR	0.005	10.528	4.395	5.677	19.224	22.722
ARR	0.01	8.264	5.020	5.485	12.544	13.313
ARR	0.1	8.394	5.643	5.948	12.075	13.313
ARR	0.4	10.802	6.113	6.505	17.487	20.511
ARR	1	–	–	–	–	–
ANN	0.005	5.746	5.512	5.563	5.937	5.982
ANN	0.01	6.148	5.883	5.934	6.353	6.420
ANN	0.1	25.921	23.857	24.250	27.672	28.133
ANN	0.4	8.515	7.652	7.810	9.147	9.436
ANN	1	7.021	5.692	5.856	8.677	9.370

Table S8: Same as table S7 for the parameter N2/Na.

7 Supplementary tables and a figure about practical recommendations regarding the implementation of the ABC-RF algorithm

NMAE

N ($\times 10^3$)	10	25	50	75	100	150	199
ra	0.028	0.023	0.021	0.020	0.019	0.018	0.018
N2/Na	0.080	0.067	0.059	0.057	0.055	0.053	0.053

OOB MSE

N ($\times 10^3$)	10	25	50	75	100	150	199
ra ($\times 10^{-4}$)	1.670	1.176	0.914	0.823	0.745	0.695	0.664
N2/Na ($\times 10^3$)	0.194	0.179	0.143	0.125	0.115	0.111	0.110

Table S9: Comparison of normalized mean absolute errors (NMAE) and out-of-bag mean squared errors (OOB MSE) for the estimation of the parameters ra and N2/Na obtained with ABC-RF, using different *reference table* sizes (N). We use the test table mentioned in subsection 3.2 of the main text. The number of trees in the RF is 500.

N ($\times 10^3$)	10	25	50	75	100	150	199
ra expectation	0.231	0.222	0.224	0.223	0.222	0.223	0.221
ra $Q_{0.025}$	0.097	0.095	0.102	0.104	0.106	0.109	0.112
ra $Q_{0.975}$	0.317	0.309	0.305	0.305	0.289	0.292	0.287
N2/Na expectation	4.538	4.588	4.652	4.530	4.475	4.483	4.508
N2/Na $Q_{0.025}$	3.651	3.679	3.782	3.802	3.751	3.840	3.831
N2/Na $Q_{0.975}$	6.621	6.221	6.621	5.611	5.555	5.315	5.424

Table S10: Estimation of the parameters ra and N2/Na for the observed population genetics dataset with ABC-RF, using different *reference table* sizes (N). The number of trees in the RF is 500.

NMAE

N_{min}	1	2	3	4	5	10	20	50	100	200	500
ra	0.019	0.019	0.019	0.019	0.019	0.019	0.020	0.021	0.023	0.027	0.033
N2/Na	0.054	0.055	0.054	0.055	0.055	0.055	0.055	0.058	0.062	0.068	0.082

OOB MSE

N_{min}	1	2	3	4	5	10	20	50	100	200	500
ra ($\times 10^{-4}$)	0.745	0.739	0.744	0.739	0.745	0.760	0.783	0.925	1.129	1.480	2.280
N2/Na ($\times 10^{3}$)	0.114	0.116	0.115	0.115	0.115	0.116	0.119	0.131	0.153	0.183	0.252

Table S11: Comparison of normalized mean absolute errors (NMAE) and out-of-bag mean squared errors (OOB MSE) for the estimation of the parameters ra and N2/Na obtained with ABC-RF, using different minimum node sizes (N_{min}). We use the reference table of size $N = 100\,000$ and the test table mentioned in subsection 3.2 of the main text. The number of trees in the RF is 500.

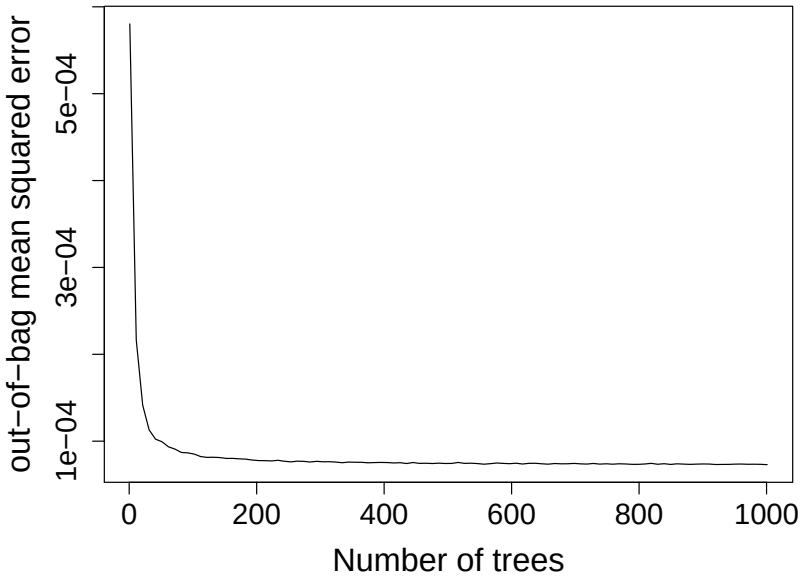

Figure S8: Relations between the number of trees in the forest and the ABC-RF out-of-bag mean squared errors, for a reference table of size $N = 100\,000$ in the human population genetics example.

8 Contribution of summary statistics in ABC-RF estimation of the parameters ra and N2/Na of the Human population genetics example

In the same spirit than in Pudlo *et al.* [11], a by-product of our ABC-RF-based approach is to automatically determine the (most) relevant statistics for the estimation of each parameter by computing a criterion of variable importance (here a variable is a summary statistic). For a summary statistic j, this measure is equal to the total amount of decrease of the residual sum of squares (RSS) due to splits over this given summary statistic, divided by the number of trees.

Indeed, at a given parent node where j is used, with n datasets, and for a given parameter of interest τ, the decrease of the RSS due to the split event is defined by the formula

$$\sum_{i=1}^{n}(\tau_i-\bar{\tau})^2-\left(\sum_{i\in \text{ left node}}(\tau_i-\bar{\tau}_L)^2 \quad +\sum_{i\in \text{ right node}}(\tau_i-\bar{\tau}_R)^2\right),$$

where $\bar{\tau}$, $\bar{\tau}_L$ and $\bar{\tau}_R$ are respectively the average of parameter values of the datasets in the parent node, left daughter and right daughter nodes. Computing this decrease among all nodes where a summary statistics j is used, among all the trees of the forest and dividing it by the number of trees is an importance measurement of the covariate j.

Figure S9 shows the contributions of the 30 most important summary statistics (among the 112 statistics proposed by DIYABC) for the ABC-RF estimation of the parameters ra and N2/Na of the Human population genetics example (see Section 3.1 of the main text). The most informative summary statistics are clearly different depending on the parameter of interest. For the admixture rate between two sources populations (ra), all ten most informative statistics correspond to statistics characterizing a pair or a trio of populations (e.g. AV or FMO statistics; see Section 5 of the supplementary materials). Moreover, all those "best" statistics include the populations ASW, GBP and YRI which correspond to the target and the two source populations respectively. On the contrary, for the effective population size ratio N2/Na, seven of the ten most most informative statistics correspond to statistics characterizing within population genetic variation (e.g. HV or HMO; see Section 5 of the supplementary materials). In this case, all those "best" statistics include the African population, which makes sense since N2 is the effective population size in the studied African population and Na in the population ancestral to all studied populations. It is worth stressing that, although the most informative summary statistics make sense in relation to the studied parameters it was difficult if not impossible to a priori and objectively select those statistics. This is not an issue when using the ABC-RF approach as the method automatically extracts the maximum of information from the entire set of proposed statistics.

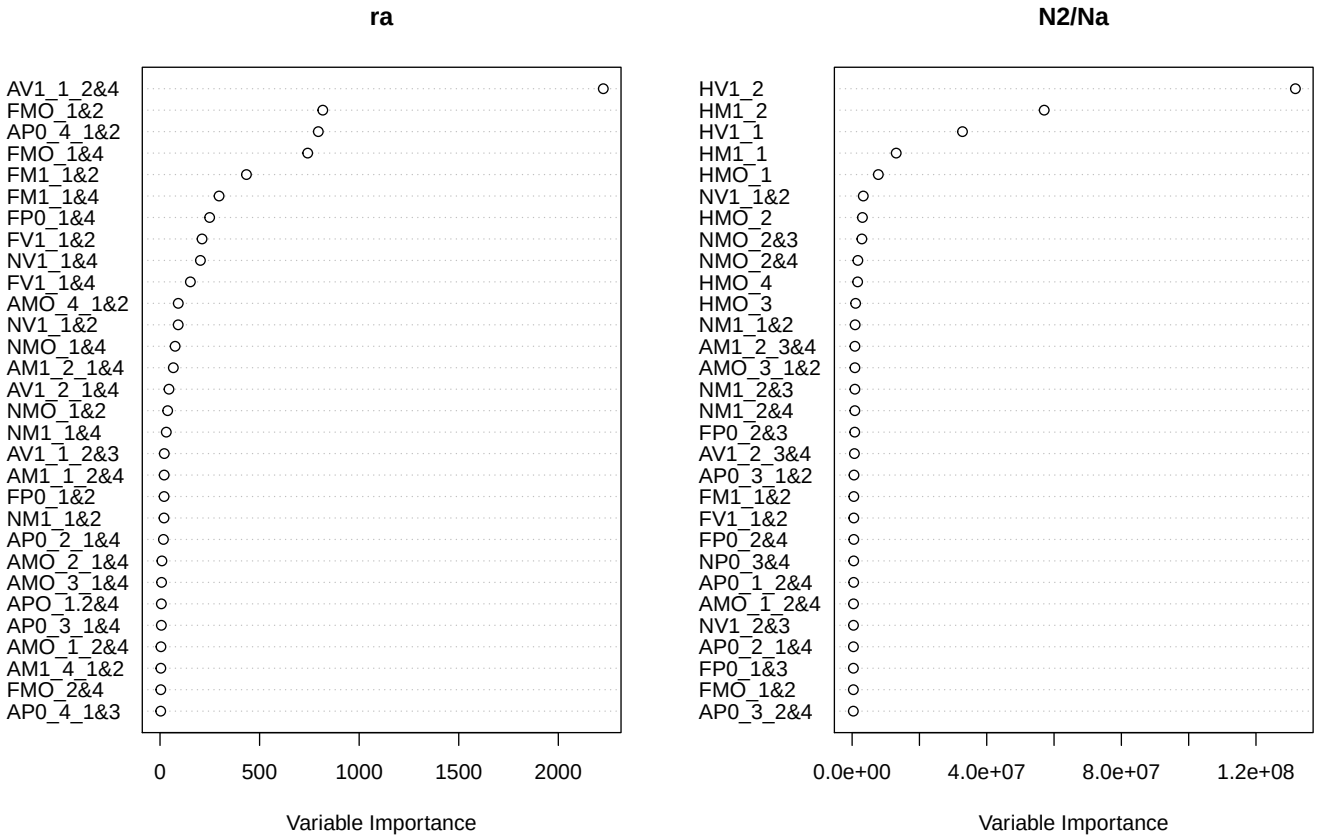

Figure S9: Contributions of the 30 most important summary statistics for the ABC-RF estimation of the parameters ra and N2/Na of the Human population genetics example. The contribution of each statistics is evaluated as the total amount of decrease of the residual sum of squares, divided by the number of trees, for each of the 112 used summary statistics provided for SNP markers by DIYABC. The higher the variable importance the more informative the statistics. The population index(s) is indicated at the end of each statistics. 1 = pop ASW (Americans of African ancestry), 2 = pop YRI (Yoruba, Africa), 3 = pop CHB (Han, Asia) and 4 = pop GBP (British, Europe). For instance FMO_1&4 = mean across loci of Fst distance between the populations 1 and 4. See also Section 5 of the supplementary materials and the Figure 4 of the main text. Note the difference of scale for the importance criterion for parameters ra and N2/Na. This difference can be explained by the difference of scale in parameter values. Indeed, it directly influences the RSS. The parameter ra being bounded in $[0, 1]$ contrary to N2/Na, a higher decrease can be expected for the ratio N2/Na than ra.

9 Computation times required by the statistical treatments of the studied methods processed following the generation of the reference table

We here present a comparison of the computation time requirement for the different methods studied in this paper, when predicting estimations of the admixture rate ra in the human population genetics example. ABC methods with rejection or adjusted with local linear regression provide the best results in terms of CPU time even when the tolerance level is equal to 1. The ABC-RF strategy requires moderately higher computing time. The calculation of the RF weights is the most expensive computation part (i.e. 3/4 of computation time). ABC methods using ridge regression or neural network correction become very time consuming when the tolerance level is high.

Method	Tol. level	CPU time (in minutes)
RF	NA	16.64
Reject	0.005	7.54
Reject	0.01	7.54
Reject	0.1	7.78
Reject	0.4	7.98
Reject	1	9.14
ALL	0.005	7.66
ALL	0.01	7.70
ALL	0.1	8.81
ALL	0.4	9.21
ALL	1	11.32
ARR	0.005	6.71
ARR	0.01	6.97
ARR	0.1	40.57
ARR	0.4	560.39
ARR	1	–
ANN	0.005	22.31
ANN	0.01	33.60
ANN	0.1	216.61
ANN	0.4	1160.67
ANN	1	4028.63

Table S12: Comparison of the computation time (in minutes) required - after the generation of the reference table - for the estimation of the parameter of interest ra on a dataset test table, using ABC-RF (RF), ABC with rejection (Reject), adjusted local linear (ALL), ridge regression (ARR) and neural network (ANN), with various tolerance levels for Reject, ALL, ARR and ANN. The test table included 1 000 pseudo-observed datasets and the reference table included 199 000 simulated datasets summarized with 112 statistics. Results were computed on a cluster with 28 CPU cores of 2.4 GHz. NA stands for not appropriate.

References

[1] Beaumont, M. A., Cornuet, J.-M., Marin, J.-M., and Robert, C. P. (2009). Adaptive approximate Bayesian computation. *Biometrika*, **96**(4), 983–990.

[2] Choisy, M., Franck, P., and Cornuet, J.-M. (2004). Estimating admixture proportions with microsatellites: comparison of methods based on simulated data. *Mol Ecol*, **13**, 955–968.

[3] Cornuet, J.-M., Pudlo, P., Veyssier, J., Dehne-Garcia, A., Gautier, M., Leblois, R., Marin, J.-M., and Estoup, A. (2014). DIYABC v2.0: a software to make approximate Bayesian computation inferences about population history using single nucleotide polymorphism, DNA sequence and microsatellite data . *Bioinformatics*, **30**(8), 1187–1189.

[4] Del Moral, P., Doucet, A., and Jasra, A. (2012). An adaptive sequential Monte Carlo method for approximate Bayesian computation. *Statistics and Computing*, **22**, 1009–1020.

[5] Klinger, E. and Hasenauer, J. (2017). A scheme for adaptive selection of population sizes in approximate Bayesian computation - sequential Monte Carlo. *Computational Methods in Systems Biology: 15th International Conference*, pages 128–144.

[6] Klinger, E., Rickert, D., and Hasenauer, J. (2018). pyABC: distributed, likelihood-free inference. *Bioinformatics*.

[7] Marin, J.-M. and Robert, C. P. (2014). *Bayesian Essentials with R*. Springer.

[8] Nei, M. (1972). Genetic distance between populations. **106**(949), 283–292.

[9] Nei, M. (1987). *Molecular Evolutionary Genetics*. Columbia University Press, New York, USA, first edition.

[10] Prangle, D. (2017). Adapting the ABC distance function. *Bayesian Analysis*, **12**(1), 289–309.

[11] Pudlo, P., Marin, J.-M., Estoup, A., Jean-Marie, C., Gauthier, M., and Robert, C. P. (2016). Reliable ABC model choice via random forests. *Bioinformatics*, **32**(6), 859–866.

[12] Weir, B. and Cockerham, C. (1984). Estimating F-statistics for the analysis of population structure. *Evolution*, **38**(6), 1358–1370.