Modelling a Zombie Apocalypse with ABC

Dan Milner, Harry Tata, Hannah Sansford

June 8, 2022

1 Introduction

Complex dynamic systems commensurate with ecology and epidemiology challenge conventional methods of statistical inference as they often have an intractable likelihood. On the other hand, simulating from the model given a parameter value may be relatively easy. Approximate Bayesian Computation (ABC) is a simulation-based inference method that enables the user to perform posterior inference by comparing simulated and observed data according to some distance metric.

Zombies, particularly when instigating an apocalypse, exhibit just such chaotic behaviour. Their aim is to infect susceptible individuals via biting their target, resulting in the bitten individual turning into a zombie after some latent infection period. Zombie apocalypses have been modelled on a number of previous occasions with varying levels of complexity; however, most models result in an intractable likelihood.

In this report we apply ABC to a zombie apocalypse model to estimate the posterior distribution of the parameters that results in the 'observed' apocalypse. Luckily, there is no real apocalypse data that could be used; hence, we decide on a model in advance and, selecting the 'true' parameters for the model, we simulate our 'observed' data. We utilise both sequential Monte Carlo (SMC) techniques and C++ programming in order to improve the computational efficiency of ABC.

2 Model

We consider four basic classes:

- Susceptible (S)
- Infected (I)
- Zombie (Z)
- Removed (R)

Susceptible individuals can become deceased through 'natural' causes (parameter δ). Removed individuals are those that have died either through a zombie attack or from natural causes. If a

human has died in a zombie attack they can resurrect and become a zombie (parameter ζ). A susceptible person can become a zombie through transmission, i.e. an encounter with a zombie (parameter β). Zombies can, therefore, only come from two sources; either they are resurrected from the newly deceased or they are a susceptible that has become infected. Prior to becoming a zombie, there is a period of time for which a susceptible individual is infected before becoming a zombie (parameter ρ). Infected individuals can still die of natural causes before becoming a zombie, otherwise they become a zombie. The birth rate of the human population is assumed to be constant (Π). Zombies can be defeated by removing the head or destroying the brain (parameter α). These transitions between classes are summarised in Figure 1.

The 4-Class model is thus described mathematically by the following system of ODEs:

$$S' = \Pi - \beta SZ - \delta S$$

$$I' = \beta SZ - \rho I - \delta I$$

$$Z' = \beta SZ + \zeta R - \alpha SZ$$

$$R' = \delta S + \alpha SZ - \zeta R$$

Hence, there are six parameters that determine the model dynamics:

- δ: background (non-zombie-related) death rate,
- ζ: zombie resurrection rate,
- β: zombie transmission rate,
- α: zombie destruction rate,
- Π: birth rate,
- ρ: latent infection rate.

We pick the following parameter values to simulate our 'true' zombie apocalypse data: $\delta = 0.0001$, $\zeta = 0.0001$, $\beta = 0.0095$, $\alpha = 0.0005$, $\Pi = 0.0001$, $\rho = 0.05$. To solve the system of ODEs, we use Euler's method with a step size of 0.05. Figure 2 shows the dynamics using our chosen parameters. We treat this realisation of the model outbreak as our 'observed' data, for which we will attempt to estimate the posterior of the parameters of our model using ABC, which we will explain in detail in the following section.

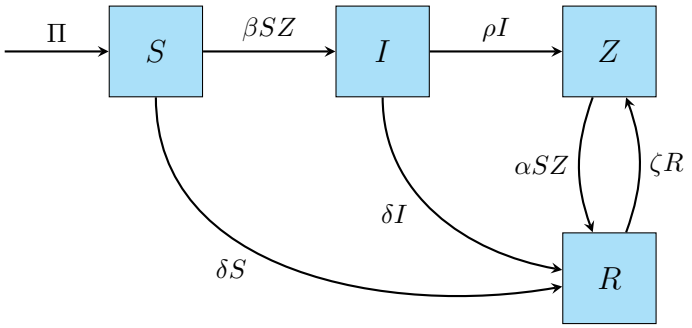

Figure 1: The SIZR model.

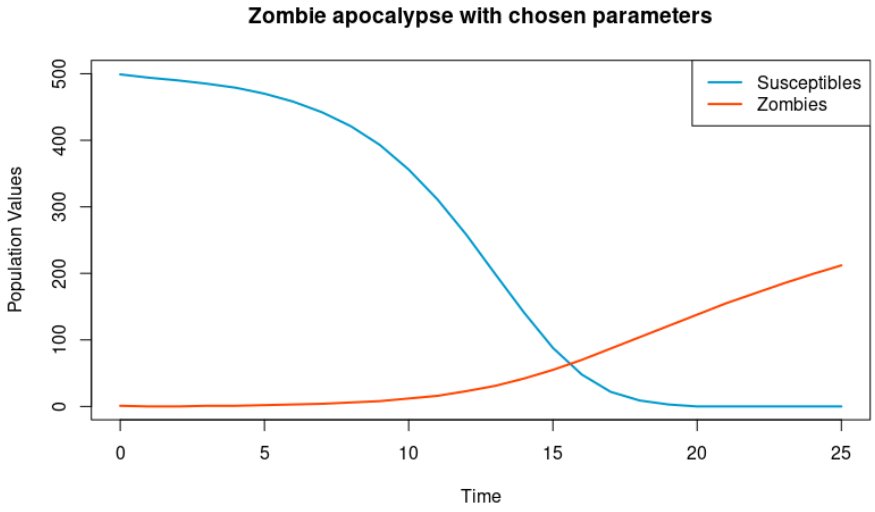

Figure 2: 'True' zombie apocalypse ('observed' data) simulated from model with chosen parameters (see main text).

3 Approximate Bayesian Computation

Likelihood-free inference methods are convenient for complex models, such as epidemic models, where we do not have an explicit expression for the likelihood. In approximate Bayesian computation (ABC), simulation under the implicit model replaces computation of the likelihood. One can use simulations from the model for different parameter values to compare the simulated datasets with the observed data. These simulations are increasingly being used as training datasets for machine learning methods including deep neural networks (Jiang et al., 2017) and random forests (Raynal et al., 2019).

The use of ABC first became popular in the field of population genetics, where simulation from a range of models is possible, but the likelihood is intractable for realistic sized datasets. Pritchard et al. (1999) were the first to use ABC, conducting inference on human demographic history. The method has now been applied in various subject areas including systems biology (Liepe et al., 2014), climate modelling (Holden et al., 2018), astronomy (Hahn et al., 2017) and epidemiology (Minter and Retkute, 2019).

The goal of ABC is to find a posterior distribution for the parameters of the implicit model, explaining the complex and potentially high-dimensional dataset. The method is based on Bayesian statistics: updating our prior beliefs about the model parameters, where $\pi(\theta)$ is the prior, using the information from our simulations. Suppose the dataset consists of n observations $y_{obs} = (y_{obs,1}, ..., y_{obs,n})$. A typical ABC procedure would involve mapping the observations to a lower dimensional set of summary statistics, $s(y)$. The posterior is then proportional to the following elements

$$p_\epsilon(\theta, s|s_{obs}) \propto \pi(\theta) f_n(s|\theta) K_\epsilon(\|s - s_{obs}\|), \tag{1}$$

where $f_n(s|\theta)$ is the implicit density of the model, $K_\epsilon(x)$ is a kernel function with tolerance ϵ, and $\|\cdot\|$ is a distance metric (Beaumont, 2019). The kernel function enables us to include in our

posterior density the parameter values that best approximate the observations; often it is simply $K_\epsilon(x) = \mathbb{I}(x \leq \epsilon)$, where $\mathbb{I}$ is the indicator function. This idea of estimating the likelihood of parameters using simulations that are 'close' to the observed data dates back at least as far as Diggle and Gratton (1984).

As the tolerance value ϵ tends to zero, the simulations we accept are closer to the true data and the approximate posterior becomes closer to its true distribution. Unfortunately, in order to simulate the same number of points at a smaller tolerance level often requires many more simulations, and the simplest algorithms can quickly become computationally inefficient. Luckily, we can harness techniques such as Markov chain Monte Carlo (Marjoram et al., 2003; Wegmann et al., 2009) and sequential Monte Carlo (Beaumont et al., 2009; Drovandi and Pettitt, 2011; Del Moral et al., 2012; Lenormand et al., 2013) to improve the computational efficiency of ABC.

3.1 Rejection algorithm

Instead of pre-defining the tolerance level ϵ, it is sometimes beneficial to simulate a set number of datasets n, of which we retain the proportion p closest to the observed data. This version of the rejection algorithm is preferential in the scenario where one is unsure of a suitable tolerance to use a priori. The algorithm used in this report is outlined below:

Algorithm 1: ABC Rejection

for $i \leftarrow 1$ **to** n **do**
- Sample $\theta_i \sim \pi(\theta)$;
- Simulate data $y_i \sim f(y|\theta_i)$;
- Transform data y_i into summary statistics s_i (optional);
- Calculate distance $\rho(s_i, s_{obs})$

end

Retain proportion p with smallest distance

Hence, in simple terms, the rejection algorithm consists of sampling parameters from a prior distribution and simulating datasets from a model using these parameters. One can then, optionally, transform these datasets into summary statistics, before retaining the proportion p closest to the observations. The accepted sample of parameters then approximates the posterior distribution.

3.2 Application to Zombie model

In our case, the data y_i are simulated from our model described in Section 2 using Euler's method with a step size of 0.05. The summary statistics s_i used in the rejection ABC algorithm are simply daily values of susceptible individuals and zombies. Since we run the dynamics over a period of 25 days, this means we have a total of 50 summary statistics. Our 'observed' data are the model dynamics simulated using the parameters stated in Section 2 and illustrated in Figure 2.

We run the ABC rejection algorithm with uniform priors on each of the parameters (over a reasonable range), $n = 10,000$ simulations, and retain the proportion $p = 0.005$ closest to the observed data. Hence, we retain 50 parameter sets to build our approximate posterior distributions, as illustrated by the histograms in Figure 3. We see that the posteriors for δ, ζ and Π do not vary greatly from the uniform priors, and the posterior means are quite far from the true parameters.

The posteriors for β and ρ are more interesting, and the posterior means seem to be closer to the true parameters; however, the posterior distributions do not concentrate well around the true parameters. Finally, for the parameter α, the posterior mean almost exactly coincides with the true parameter; however, again the distribution does not concentrate well around this value and does not seem too different from the uniform prior.

In order to compare the rejection ABC algorithm to ABC with SMC in the following section, we report the maximal (normalised) distance (tolerance level) ϵ from the accepted summary statistics to the observed data: we get a value of $\epsilon = 26.56$. We write functions to simulate from our model in both R and C++, before using the `ABC_rejection` function from the `EasyABC` R package to run the algorithm. Using the R function, the algorithm took 13.30 seconds to run; whereas, using the C++ function, the algorithm took 7.11 seconds to run - nearly twice as fast.

To illustrate how close our accepted simulations are to the observed data, Figure 4 shows a selection of 25 of the accepted simulations, alongside the observed dynamics. We can see that they vary quite greatly from the true data, indicating that the rejection ABC algorithm requires a very large number of simulations in order to sample from the target parameter space. One can imagine that for a model that is more expensive to simulate from, or when using a very wide prior, this method would be too computationally expensive to be of any use. Figure 5 shows the mean $\pm$ one standard deviation alongside the observed trajectories. We can see that the accepted parameters tend to result in trajectories that underestimate both susceptibles and zombies.

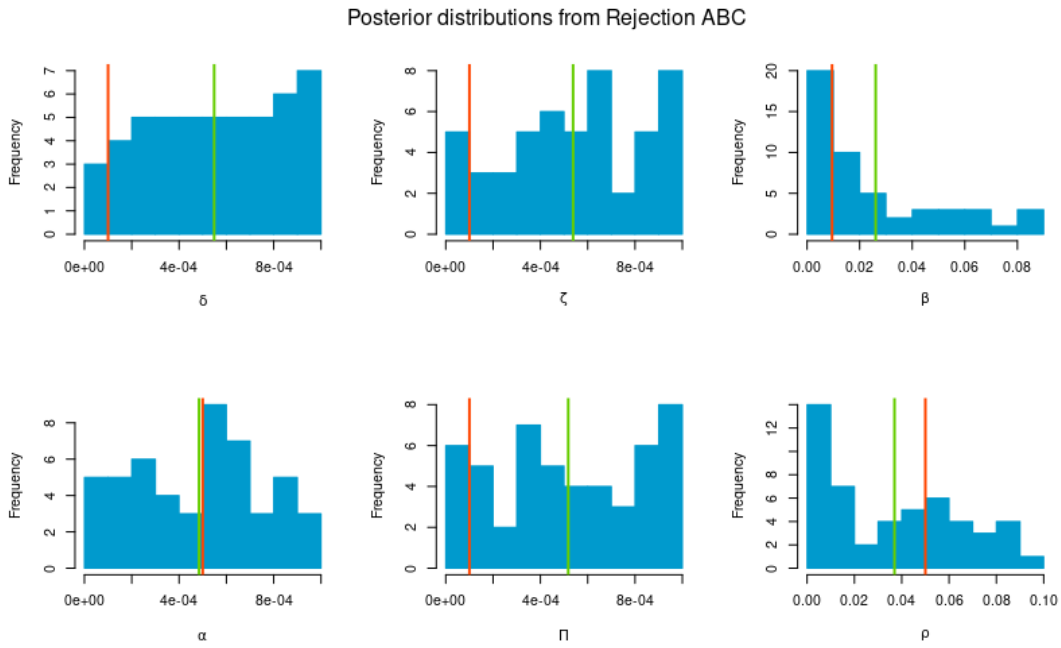

Figure 3: The posterior distributions of the model parameters, attained from the ABC rejection algorithm, with $n = 10,000$, $p = 0.005$. The values of the true parameters are indicated by the red lines, while the posterior means are indicated by the green lines.

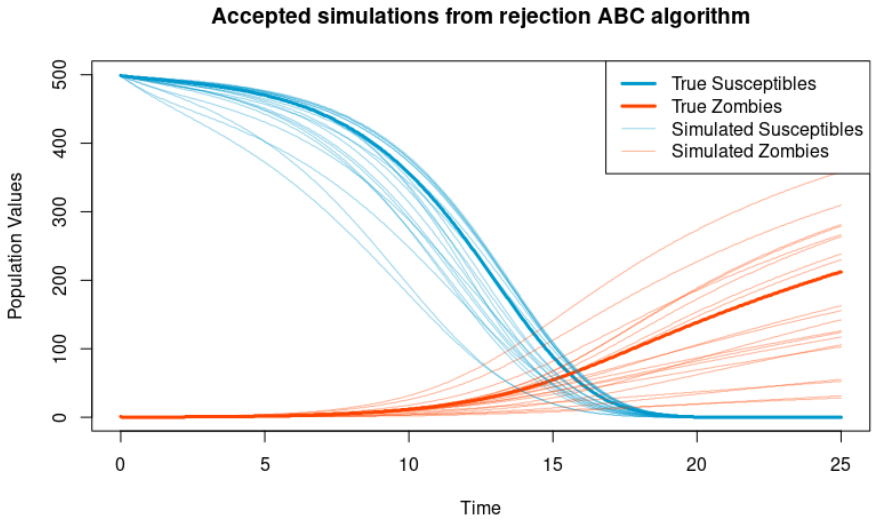

Figure 4: The above figure shows a selection of simulations from the model with accepted parameter values from the ABC rejection algorithm, with $n = 10,000$, $p = 0.005$, alongside the 'observed' dynamics.

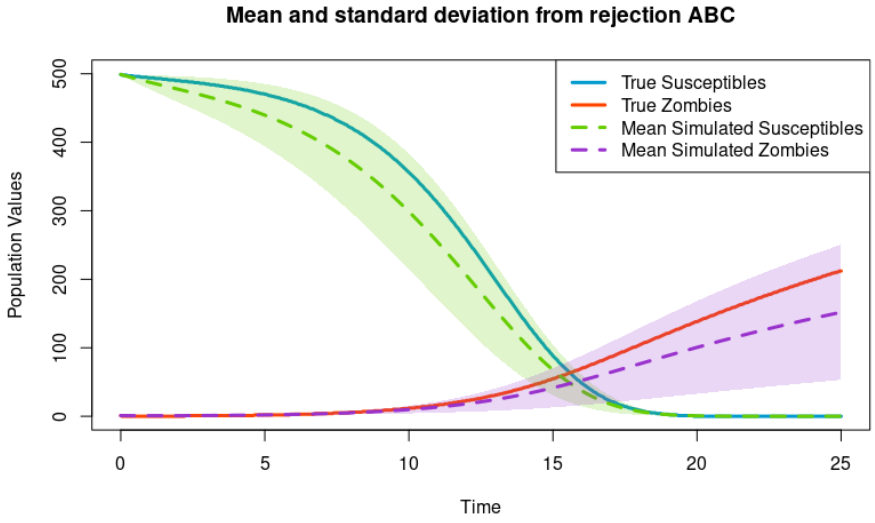

Figure 5: The above figure compares the 'observed' dynamics (solid curves) with the mean of the accepted simulations from the ABC rejection algorithm (dashed curves). The shaded regions indicate $\pm$ 1 standard deviation from the mean.

4 Sequential Monte Carlo

Sequential Monte Carlo (SMC) methods applied to ABC are concerned with reducing the number of model runs in order to achieve a certain quality of posterior approximation. We have now seen that the simple ABC rejection algorithm is very computationally demanding, which often limits applications to simple models; the number of simulations required to sample the entire parameter space grows exponentially with the number of parameters (Beaumont, 2010). SMC methods aim to progressively approximate the posterior using a decreasing sequence of tolerance levels $\{\epsilon_1, ..., \epsilon_T\}$. The idea is to sample from areas of the parameter space with a higher likelihood, rather than systematically from the whole space, in order to gain computational efficiency. The basis of SMC algorithms is to construct a sequence of target distributions, that one propagates a set of particles through in order to move from the initial distribution to the final distribution.

4.1 Population Monte Carlo

The first stage of the PMC ABC method of Beaumont et al. (2009) is identical to that of rejection ABC in Algorithm 1, giving the first stage approximation to the posterior distribution. One then resamples parameter values from a density kernel

$$q(\theta) = \sum_{j=1}^{N} w_j^{(t-1)} K(\theta | \theta_j^{(t-1)}; \tau_t^2),$$

with N being the number of samples in the first, and subsequent, iterations. A popular choice of K is a Gaussian kernel, and Beaumont et al. (2009) shows that a good choice of τ is twice the empirical variance of the simulated parameters in the previous iteration. The bias that is induced by sampling from a proposal distribution, rather than the prior, is fixed by assigning an importance weight to each particle, as outlined in Algorithm 2 below.

A drawback of this method is that it requires the user to pre-define a decreasing sequence of tolerance levels $\{\epsilon_1, ..., \epsilon_T\}$. A poor choice of sequence could offset the possible benefits of the importance sampling procedure. In the population genetic example conducted by Beaumont et al. (2009), ϵ_1 is based on a preliminary simulation as the 0.1 quantile and they perform four iterations with $\epsilon_2 = 0.75\epsilon_1$, $\epsilon_3 = 0.9\epsilon_2$ and $\epsilon_4 = 0.9\epsilon_3$.

4.2 Application to Zombie model

We apply the PMC-ABC algorithm using the `ABC_sequential` function from the `EasyABC` package, setting `method="Beaumont"`. We set $N = 50$ in Algorithm 2 and choose the sequence of tolerance levels to be $\{30, 10, 5, 1, 0.5\}$. The choice of the first tolerance level was informed by the epsilon value obtained from the rejection ABC algorithm (26.23): $\epsilon = 30$ was chosen so that the first stage of the algorithm would not require too many simulations. The subsequent tolerance levels were chosen through intuition and trial and error.

Using our R function to simulate from the model, the algorithm required a total of 3768 simulations and took 2.19 seconds to run. Using the C++ function, 3272 simulations were performed and the run-time was 0.81 seconds. The final maximum epsilon value of the accepted simulations was 0.484. This is nearly 7 times faster than the rejection ABC algorithm, and the accepted simulations

Algorithm 2: Population Monte Carlo ABC (PMC)

Given a decreasing sequence of tolerance levels $\{\epsilon_1, ..., \epsilon_T\}$;
for $t = 1$ **do**
  for $i = 1$ *to* N **do**
    until $\rho(y, y_{obs}) < \epsilon_1$;
    Simulate $\theta_i^{(1)} \sim \pi(\theta)$ and $y \sim f(y|\theta_i^{(1)})$
  end
  Set $w_i^{(1)} = 1/N$
end
for $t = 2$ *to* T **do**
  for $i = 1$ *to* N **do**
    until $\rho(y, y_{obs}) < \epsilon_t$;
    Sample θ_i^* from $\theta_j^{(t-1)}$ with probabilities $w_j^{(t-1)}$;
    Generate $\theta_i^{(t)} \sim K(\theta|\theta_i^*; \tau_t^2)$ and $y \sim f(y|\theta_i^{(t)})$
  end
  Set $w_i^{(t)} \propto \pi(\theta_i^{(t)}) / \sum_{j=1}^{N} w_j^{(t-1)} K(\theta_i^{(t)}|\theta_j^{(t-1)}; \tau_t^2)$;
  Set τ_{t+1}^2 as twice the weighted emprirical variance of the $\theta_i^{(t)}$'s
end

are considerably closer ($\epsilon = 26.56$ in rejection ABC). Figure 6 illustrates this well, showing 25 of the accepted trajectories; they are much closer to the observed trajectories, and seem to be equally distributed above and below. Figure 7 shows that the mean of the accepted trajectories almost exactly coincides with the observed ones, and the standard deviation is much smaller than in the rejection ABC output.

Finally, the posterior distributions for the model parameters, as shown in Figure 8, are much better than those obtained from rejection ABC. In particular, the posterior means for the parameters β, α and ρ almost exactly coincide with the true parameters. The shape of the posterior distribution for these parameters has also certainly deviated from the uniform priors, looking almost Gaussian, centered at the true parameter values. The posteriors for δ, ζ and Π still look fairly uniform (as with rejection ABC), indicating that these parameters have less of an affect on the trajectories and hence are harder to estimate.

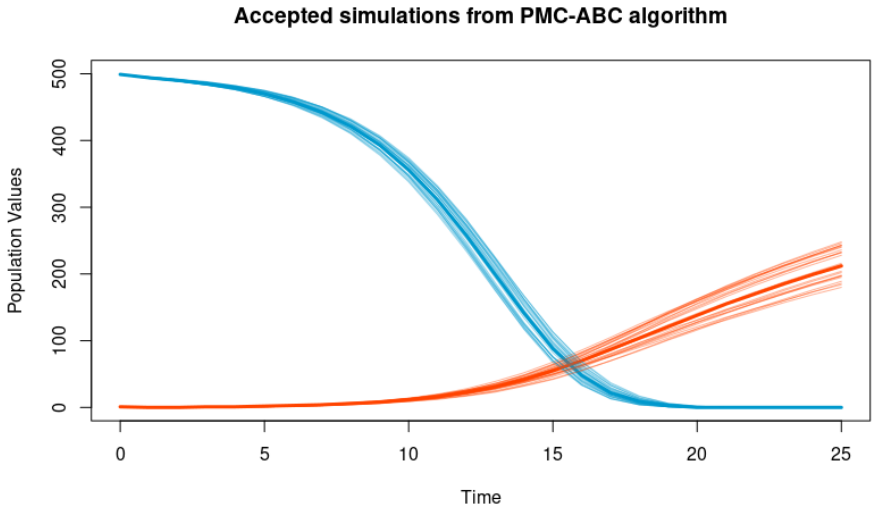

Figure 6: The above figure shows a selection of simulations from the model with accepted parameter values from the PMC-ABC algorithm, with the decreasing sequence of tolerance levels $\{10, 5, 1, 0.5\}$ and $N = 50$, alongside the 'observed' dynamics.

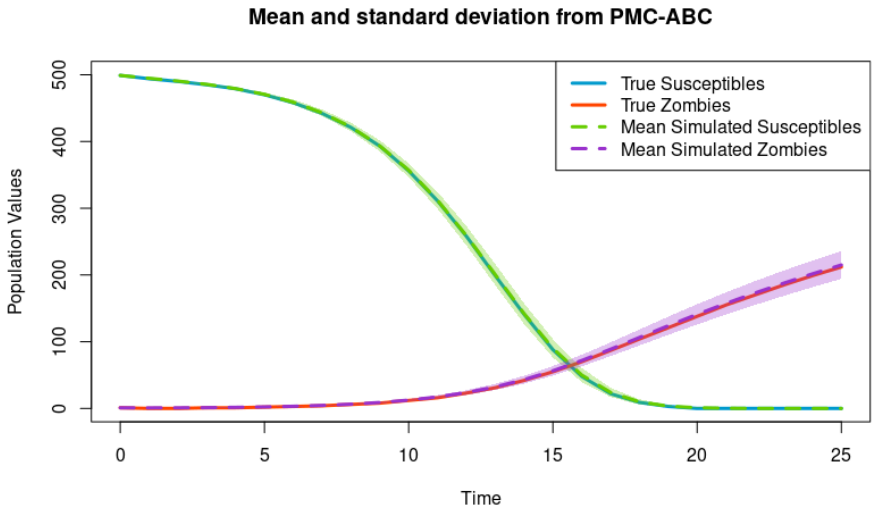

Figure 7: The above figure compares the 'observed' dynamics (solid curves) with the mean of the accepted simulations from the PMC-ABC algorithm (dashed curves). The shaded regions indicate $\pm$ 1 standard deviation from the mean.

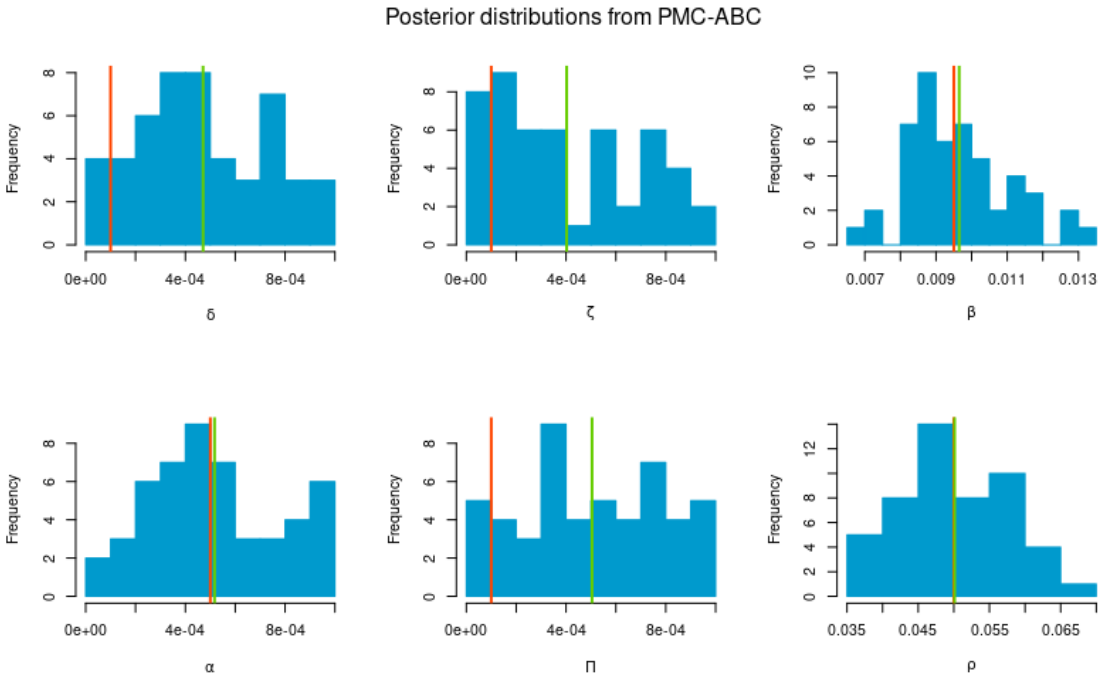

Figure 8: The posterior distributions of the model parameters, attained from the PMC-ABC algorithm, with the decreasing sequence of tolerance levels $\{10, 5, 1, 0.5\}$ and $N = 50$. The values of the true parameters are indicated by the red lines, while the posterior means are indicated by the green lines.

5 Conclusion

For a relatively basic zombie apocalypse model (essentially an epidemic model), with an intractable likelihood function, we have seen the limitations of using the simplest form of ABC algorithm. For more complex models, that are expensive to simulate from, involve stochasticity and have a larger parameter space, the effects of these limitations will be amplified greatly.

We have explored methods to mitigate these issues: sequential Monte Carlo (SMC) techniques and C++ programming have both proved fruitful. We implemented a simple SMC method based on population Monte Carlo, which capitalises on exploring the parameter space that resulted in simulations close to the observed data on previous iterations. Simulating from our model using a C++ function also resulted in significant reductions in algorithm run-times.

Further improvements could be made through exploiting the 'embarrassingly parallel' nature of ABC. However, we struggled to parallelise our code within the report's time constraints.

Code for all the experiments conducted in this report can be found at `https://github.com/hsansford1/zombie_project` .

References

Beaumont, M. A. (2010). Approximate bayesian computation in evolution and ecology. *Annual Review of Ecology, Evolution, and Systematics*, 41(1):379–406.

Beaumont, M. A. (2019). Approximate bayesian computation. *Annual Review of Statistics and Its Application*, 6(1):379–403.

Beaumont, M. A., Cornuet, J.-M., Marin, J.-M., and Robert, C. P. (2009). Adaptive approximate bayesian computation. page arXiv:0805.2256.

Del Moral, P., Doucet, A., and Jasra, A. (2012). An adaptive sequential monte carlo method for approximate bayesian computation. *STATISTICS AND COMPUTING*, 22(5):1009–1020.

Diggle, P. J. and Gratton, R. J. (1984). Monte carlo methods of inference for implicit statistical models. *Journal of the Royal Statistical Society. Series B (Methodological)*, 46(2):193–227.

Drovandi, C. C. and Pettitt, A. N. (2011). Estimation of parameters for macroparasite population evolution using approximate bayesian computation. *Biometrics*, 67(1):225–33.

Hahn, C., Vakili, M., Walsh, K., Hearin, A. P., Hogg, D. W., and Campbell, D. (2017). Approximate bayesian computation in large-scale structure: constraining the galaxy–halo connection. *Monthly Notices of the Royal Astronomical Society*, 469(3):2791–2805.

Holden, P. B., Edwards, N. R., Hensman, J., and Wilkinson, R. D. (2018). Abc for climate: dealing with expensive simulators. *Handbook of approximate Bayesian computation*, pages 569–95.

Jiang, B., Wu, T.-y., Zheng, C., and Wong, W. H. (2017). Learning summary statistic for approximate bayesian computation via deep neural network. *Statistica Sinica*, pages 1595–1618.

Lenormand, M., Jabot, F., and Deffuant, G. (2013). Adaptive approximate bayesian computation for complex models. *Computational Statistics*, 28(6):2777–2796.

Liepe, J., Kirk, P., Filippi, S., Toni, T., Barnes, C. P., and Stumpf, M. P. (2014). A framework for parameter estimation and model selection from experimental data in systems biology using approximate bayesian computation. *Nature protocols*, 9(2):439–456.

Marjoram, P., Molitor, J., Plagnol, V., and Tavaré, S. (2003). Markov chain monte carlo without likelihoods. *Proceedings of the National Academy of Sciences*, 100(26):15324–15328.

Minter, A. and Retkute, R. (2019). Approximate bayesian computation for infectious disease modelling. *Epidemics*, 29:100368.

Pritchard, J. K., Seielstad, M. T., Perez-Lezaun, A., and Feldman, M. W. (1999). Population growth of human y chromosomes: a study of y chromosome microsatellites. *Mol Biol Evol*, 16(12):1791–8.

Raynal, L., Marin, J.-M., Pudlo, P., Ribatet, M., Robert, C. P., and Estoup, A. (2019). Abc random forests for bayesian parameter inference. *Bioinformatics*, 35(10):1720–1728.

Wegmann, D., Leuenberger, C., and Excoffier, L. (2009). Efficient approximate bayesian computation coupled with markov chain monte carlo without likelihood. *Genetics*, 182(4):1207–18.