Reverse Engineering the Lotka-Volterra Equations

Comparison of local and global optimisation algorithms

December 17, 2020

Isabelle Brakenhoff
isabelle.brakenhoff@student.uva.nl
11283912
Universiteit van Amsterdam

Jorrim Prins
jorrimprins.prins@student.uva.nl
11038934
Universiteit van Amsterdam

***Abstract*—This research attempts to reverse engineer a predator-prey model with multiple objective functions and various techniques for optimisation. We specify the Root Mean Squared Error and Mean Absolute Error as our objectives and use these for optimisation and performance measuring. Differences between local an global optimisation algorithms with an iterative structure, Hill Climbing and Simulated Annealing respectively, are studied and their ability to estimate the parameters for the Lotka-Volterra equations is compared. Optimisation of these algorithms proofs to be a laborious task, especially for Simulated Annealing and definitive conclusions on algorithm performance for various availability of data can not be provided. Nevertheless, promising results are observed for an alternative global optimisation scheme in the Genetic Algorithm, even after little tuning effort.**

I. Introduction

The works of Lotka and Volterra (Lotka, 1920; Goel et al., 1971) have caused a turning point in the history of mathematical ecology to describe various dynamic aspects of interactive populations. The simplicity of implementation and plethora of applications that come with the equations introduced by Lotka and Volterra are unmatched. They are frequently used to model a range of biological systems and are particularly interesting for analysis of predator-prey models (Takeuchi, 1996). This paper will try to fit a dataset of predator and prey populations to the Lotka-Volterra (LV) equations. We reverse engineer the equations, finding optimal parameters by local and global optimisation and the differences between solutions are subsequently investigated. In addition, we provide insight into the effectiveness of the optimisation methods under decreased availability of data and we explore which segments of the dataset are most important for accurate estimation.

The simplest form of the LV equations is an Ordinary Differential Equation (ODE) representation of a closed system with two species, where the population of prey is denoted with x and the population of predators with y. The LV equations assume a constant growth rate of the prey population α in absence of predators. When the predator population is greater than zero, its growth will have a decreasing effect on the prey population. This relation is again assumed to be linear, now with constant rate β. Intuitively, the reverse of these relations is true for the differential equation of the predator population. An increasing prey population increases the predator population linearly with rate δ, but the value for y itself decreases the population of predators again with constant rate γ. All of these effects are assumed to be larger than zero, so the four parameters in our parameter set $(\alpha, \beta, \gamma, \delta) > 0$. This results in the following system of ODE's:

$$\begin{aligned} \frac{dx}{dt} &= x(\alpha - \beta y) \\ \frac{dy}{dt} &= y(\delta x - \gamma) \end{aligned} \tag{1}$$

We assume Equation 1 to be an accurate representation of a predator-prey model and want to optimise the accuracy by optimising the 4 parameters it has. Accuracy in this research is represented by the error between the experimental data and the predicted data from the estimated LV equations. The error can be represented in various ways, we choose to implement two widely used error measures in the Root Mean Squared Error (RMSE) and the Mean Average Error (MAE). These measures are, as we will highlight later, used during the optimisation process and eventually as a performance measure. The performance of both measures has been frequently compared in literature and conclusions vary greatly (Chai and Draxler, 2014; Willmott and Matsuura, 2005). An important technical difference between the two measures is their handling of outliers. As will become visible below, the RMSE will give a higher weight to large errors or outliers than the MAE.[1]

The objective function under RMSE looks as follows:

$$\min \sqrt{\frac{1}{n}\sum_{t=0}^{n}(\hat{z}_t - z_t)^2} \tag{2}$$

where n is specified as the number of total available data points, $\hat{z}_t$ represents either the estimated prey population size $\hat{x}_t$ or the estimated predator population size $\hat{y}_t$ and z_t is equally defined for the actual population sizes. Note that the

[1] This is caused by the square of the error before summation in the RMSE definition.

final RMSE/MAE is the average of the RMSE/MAE's for x and y. In similar notation, we define the objective function for MAE as:

$$\min \frac{1}{n} \sum_{t=0}^{n} |\hat{z}_t - z_t| \tag{3}$$

We are interested in finding an accurate LV representation for our dataset using the error functions above, we specify two iterative algorithms to perform this task. Initial estimates of the parameters are obtained by the Hill Climbing algorithm (HC), which is known to be prune to finding local optima. Expecting our objective landscape to contain many local optima, we alternatively specify a Simulated Annealing algorithm (SA) that should hypothetically outperform HC. Section II highlights these algorithms, followed by an overview of our parameter tuning and additional experiments. The experimental we discuss in this research is presented in Section III and the results of our experiments can be found in Section IV. We conclude our findings in Section V, providing an overview of our research and make implications for future work.

II. Method

The first iterative algorithm we implement is HC and is intuitively defined. SA can be thereafter defined similarly, with an adapted and more complex structure. Both algorithms will update our parameter set $(\alpha, \beta, \gamma, \delta)$ towards an optimal value, evaluating its performance by the measures in Equation 2 and 3. The estimations for $\hat{x}$ and $\hat{y}$ are obtained through integration of the ODE's in Equation 1 by SciPy. In the updating process, several parameter settings have to be specified, which we will discuss below.

A. Optimisation Algorithms

HC starts with a solution within the search space, finds a neighbour of this solution via a specified generator, evaluates this neighbour and replaces the old solution with the neighbour if it performs better. This single iteration is repeated a number of times (N), a complete overview of the algorithm can be found in Algorithm 1. As performing better is defined as scoring lower on the error measure (RMSE or MAE), the solution will converge to a minimum of this measure. Nevertheless, HC is a local optimisation algorithm and accepts a neighbouring solution if and only if its value for the objective improves. The solution of the algorithm will quickly converge to an optimum, but will likely converge to a local one (Al-Betar, 2017). The chance that this happens increases with the dimensionality of the problem. The two following situations can happen: Neighbours are chosen close to the current solution, making the algorithm converge to the nearest minimum (which is not necessarily global). Alternatively, further neighbours are picked and the algorithm will likely overshoot the global minimum to converge to another local minimum. The reasoning behind this can be visualised relatively easy in one dimension, see Figure 1 by Yang (2018) for example. HC would get stuck in any valley on this graph, not necessarily the global minimum.

Algorithm 1: Hill Climbing

Define search space S and neighbour generator G.
Set initial solution P in S.
Evaluate $f(P)$, with f specified error function.
For i in N:
  $P_{new} = G(P)$
  If $f(P_{new}) < f(P)$:
    $P = P_{new}$

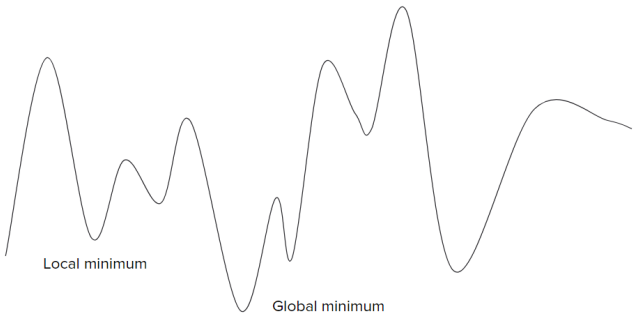

Fig. 1. Example of a one-dimensional objective function with multiple local optima by Yang (2018). Hill Climbing would highly likely converge to one local optima, as it would not be able to overcome a peak. Simulated Annealing would on the other hand theoretically be able to converge to the global minimum.

We try to visualise the convergence to different local optima by optimising the Hill Climbing algorithm with different starting values. Convergence of HC is shown with starting parameter sets of $(0, 0, 0, 0)$, $(1, 1, 1, 1)$ and $(2, 2, 2, 2)$ to visualise the importance of the starting point. As optimal starting values for problems in higher dimensions are near impossible to find, we compare HC and SA with 1 set of starting values for LV parameters and do not try to optimise this starting point any further. The starting set of LV parameters is simply specified to be equal to $(0, 0, 0, 0)$.

The tendency to local optima in HC can be avoided by specifying a more complex iterative scheme, SA follows roughly the same steps but can achieve improved results by using Markov Chain Monte Carlo samples. Where HC updates the current solution solely when it has increased performance, SA will make acceptance of a worse solution possible with a certain possibility. This could help overcome certain hills to eventually reach the global minimum (see Figure 1). The iterative scheme of HC is roughly followed, but the loop is split into $\frac{N}{L}$ iterations of a Markov Chain of length L. Within the inner loop, exploration happens again through stochastically finding neighbours and the final solution of the loops is the new update. An improved solution in the inner loop is always accepted. The possibility of accepting a worse solution is adaptive, will decrease over time and is modelled with a ratio of Boltzmann densities (Sharp and Matschinsky, 2015) as $e^{\frac{-(f(P_{new})-f(P))}{T}}$ (Al-Betar, 2017). A complete overview of the algorithm can be found in Algorithm 2 and the specification of temperature parameter T will follow.

Algorithm 2: Simulated Annealing

Define search space S and neighbour generator G.
Set initial solution P in S, temperature range $(T_0, T_{\frac{N}{L}})$ and cooling schedule $T_i \rightarrow T_{i+1}$.
Evaluate $f(P)$, with f specified error function.
For i in $\frac{N}{L}$:
  For j in L:
    $P_{new} = G(P)$
    If $f(P_{new}) < f(P)$:
      $P = P_{new}$
    Elif $U(0,1) < e^{\frac{-(f(P_{new})-f(P))}{T_i}}$:
      $P = P_{new}$
  $T_i = T_{i+1}$

B. Parameter Tuning and Experiments

As described above, both optimisation methods need to specify a method to generate a neighbour of the current optimal solution to evaluate. This is a highly problem specific process and it can be done in various ways. We explore two methods to effectively perturb our solution, the set of the 4 current LV parameters, to find a neighbour. Solutions in our research are perturbed by adding random noise, which has to come from a symmetric distribution to provide unbiased results. Noise is generated through a uniform distribution ranging from -1 to 1 and is multiplied with a step size s, which we discuss below. The random noise is added to one parameter at a time, or to all parameters simultaneously, and we compare differences in performance between these methods to find neighbours. Furthermore, the problem specifies a lower boundary for the parameters at 0, we find an upper boundary of approximately 10 after some experiments. Every perturbation of parameters that results in a value outside this range will be clipped to the boundary.

The step size determines the range of exploration around the current solution. Its optimal value depends on the distribution of the noise term, which we therefore leave unchanged. Furthermore, it depends on the search space and the number of iterations computed (N).[2] We provide experiments with different values of the step size s and try to find an optimal setting. Note that our neighbour-generators have different implications for the settings for s. The first method only adds noise to one parameter and is therefore hypothesised to affect the resulting value for the error function less than the second method. For an equal N in the algorithm, it is therefore less explorative on average. We expect to find a higher optimal s for the method that only changes 1 parameter to compensate for this.

The parameters described above have implications for the definition of both HC and SA models. The SA model however has a number of additional parameters that can impact performance, we explore the length of its Markov Chains L, the distribution of the cooling schedule and the initial and final temperature of the cooling schedule T_0 and T_N.

The length of SA's Markov Chain can be easily portrayed by the inner loop of Algorithm 2. In combination with step size s and number of iterations N it determines the possible distance between the current solution and the proposed new solution. Increasing the length of the chain L leads to possible extra steps of improvement, but will simultaneously lower the number of new explorations that can be performed (which is intuitively equal to N/L). With the limited computational power available, N can not become extremely large and finding an optimally performing value of L is important.

The probability of accepting a solution that performs worse than the current one is regulated by the temperature variable T in the Boltzmann distribution (Sharp and Matschinsky, 2015). Exploration can be performed for high T, as the acceptance probability for a solution is relatively high. Through the iterations, we want to narrow down this process by decreasing T towards zero. This eventually results in a HC-like process where non-improved solutions are never accepted. Experiments are performed to find optimal values of the starting temperature T_0 and the final temperature $T_{\frac{N}{L}}$, so that initial exploration is adequate and the algorithm eventually converges.

The temperature is decreased after every iteration of the outer loop of the SA algorithm and various distributions can be implemented to schedule this decay. Martín and Sierra (2009) compare a variety of distributions, which can be multiplicative, additive, (non-)monotonic or a combination thereof. We decide to implement a linear decay of the temperature and compare this with an exponential and a quadratic decrease. An example of the development of temperatures under these cooling schedules can be observed in Figure 2, with significant differences between the schedules.

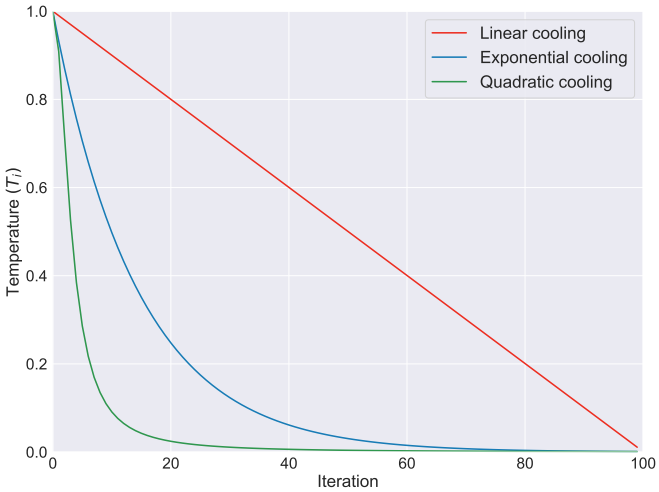

Fig. 2. Development of cooling schedules for Simulated Annealing with temperature ranging from 1 to 10^{-3} and $\frac{N}{L} = 100$.

Having explored the behaviour of the optimisation algorithms under varying parameters, we are interested in performance under a decreased amount of data. Initially, we

[2]More specifically, it also depends on the length of the Markov Chain L when the SA algorithm is used

randomly delete samples from the data to see how accurate the algorithms are under a smaller number of samples. This is done for the prey and predator population separately, as well as simultaneously. Additionally, we examine the importance of the peaks and the bottoms of the data-distribution, deleting high and low values respectively. Comparisons can now be made between the importance of specific datapoints and randomly selected ones for obtaining accurate parameter estimations.

All of the experiments in this research are performed for both RMSE and MAE as objective functions, with a total number of iterations N equal to 5000 and every simulation repeated 25 times for statistical analysis.

C. An out-of-the box alternative: Genetic Algorithms

Where the two iterative algorithms HC and SA have a similar structure and are widely used in local and global optimisation, a variety of alternative options exists. The thermodynamic process of annealing from metallurgy has been the source for using SA in optimisation, and borrowing phenomena from nature to implement elsewhere is common. We attempt to broaden the perspective on the parameter optimisation in this research, by looking at a method that iterates over a generation of solutions rather than a single one. Algorithms like these are often described as Evolutionary Algorithms and we implement a simple Genetic Algorithm (GA) for comparison. We will not perform extensive parameter tuning and other experiments on the GA, but we simply want to indicate an interesting area for extension.

The main principle of a GA is to randomly create a population of solutions and use Darwinian principles to improve the optimal solution by natural selection. Parents are selected from the population, create offspring that undergo some stochastic mutation and the union of parents and offspring is reduced to the stable population size by selection. This represents one generation, and the process is repeated until the population is homogeneous or delivers a satisfying solution. A complete overview of the algorithm can be found in Algorithm 3.

Algorithm 3: Genetic Algorithm

Randomly initialise population of size N_{pop} from search space S.
Evaluate initial population.
For i in N_{gen}:
  Select N_p parents.
  For j in $\frac{N_p}{2}$:
    For k in N_o:
      $O_k = \alpha P_j + (1-\alpha)P_{j+1} \quad \alpha \in (0,1)$
      Mutate O_k.
  Evaluate O.
  Select N_{pop} survivors from $P \cup O$.

The initial population in our GA consists of 100 solutions, of which the 50 best performing solutions are selected as parents. Performance is again based on the RMSE or MAE of a solution. Each pair of parents creates a randomly chosen number of offspring between 1 and 4, the offspring solution is equal to an arithmetic combination of its parents. With a mutation probability of 0.3, an offspring solution is mutated by a normal distributed perturbation with $N(0, \frac{1}{25})$ to ensure plausible mutations. After mutation, the 100 best solutions survive and become the starting point for the next generation. This process is repeated for 50 generations.

Note that an abundance of possible alternative algorithm or parameter settings is possible, from the parent and survivor selection mechanisms to the offspring creation mechanism, the mutation strategy and of course all the parameters accompany these mechanisms. The purpose of our implementation is merely to give an initial performance indication for an alternative to HC and SA, so we decide to keep the model straightforward.

III. Data

The experimental data that we use provides population sizes for preys and predators. Data is provided for equal intervals between $t = 0$ and $t = 20$ and a scatter plot of the data can be observed in Figure 3. It is provided that the data comes from Lotka-Volterra equations, with some added noise. Peaks in the prey population size are closely followed by peaks in the predator population for every oscillation. The maximum size of the prey population is significantly lower than the maximum size of the predator population, with values of 3.95 and 6.37 respectively.

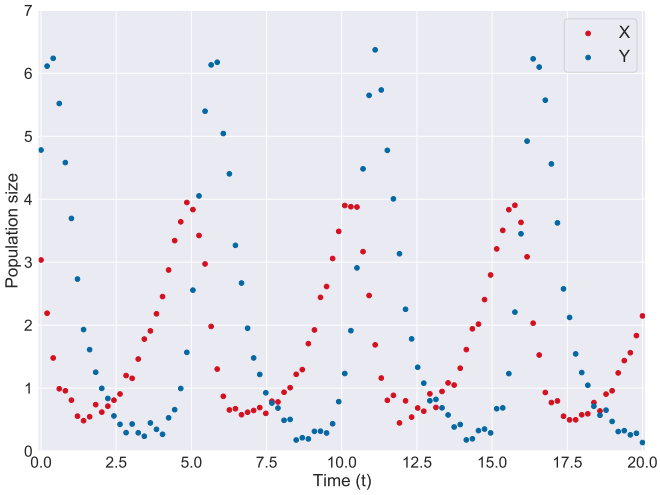

Fig. 3. Complete visualisation of the examined data. Distributions can be clearly observed to come from Lotka-Volterra equations (with added noise).

IV. Results

Exploration of the performance differences between HC and SA algorithms is only necessary because of the local search nature of HC. Changing the starting value for the search is expected to clearly move the convergence of HC to a different minimum. We can clearly observe this effect in Figure 4, where optimal parameter estimations are shown

for varying starting values under RMSE optimisation. From initial little experiments we learn that the optimal parameter values are likely to be found close to 1 or 2, depending on the specific parameter. This can be confirmed by the accurate fit coming from HC estimations with $(1, 1, 1, 1)$ and $(2, 2, 2, 2)$ in Figure 4. On the other hand, the optimisation process seems to get stuck at a weakly performing local optimum for starting parameters $(0, 0, 0, 0)$, as the fit for the LV equations is very inaccurate. We will continue to use the latter starting point for our simulations.

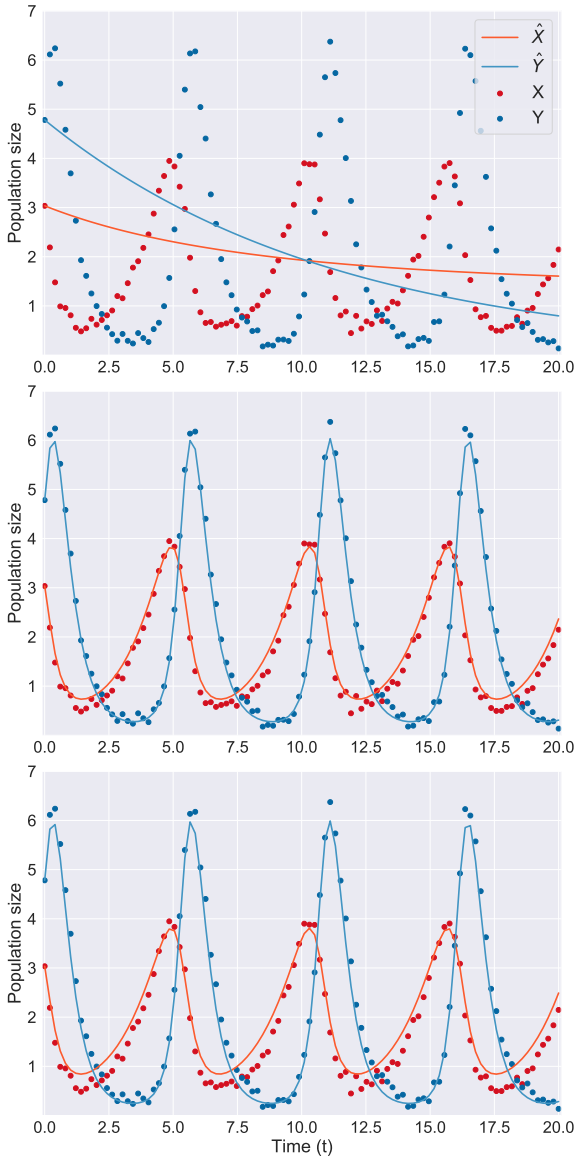

Fig. 4. Data fits of the Lotka-Volterra equation, parameters optimised by Hill Climbing with $N = 5000$, $s = 0.5$, RMSE and starting values $(0, 0, 0, 0)$, $(1, 1, 1, 1)$ and $(2, 2, 2, 2)$ from top to bottom. Optimally performing parameters over 25 replications.

Defining an effective neighbour generating mechanism is our second concern and will automatically induce an investigation of the step size of our solution perturbations. We run experiments comparing a mechanism that perturbs one of the LV parameter at at time to a mechanism that perturbs all parameters simultaneously. For SA, some step size values between 0.1 and 1.0 are explored for the latter mechanism and step sizes of 0.25 is determined to perform optimally under MAE. Under RMSE, all values between 0.25 and 1.0 behave very similarly and clearly outperform Hill Climbing as can be seen in Figure 5. As we hypothesised the former neighbor generator to need a higher step size because of the smaller exploration steps, we examine step sizes from 0.5 to 2 for this mechanism and pick an optimal value of 1. The step size for HC is kept relatively small in both cases and set to 0.5. We decide to use the generator that perturbs every parameter in the set simultaneously, as this converges slightly faster and gives similar results to the other generator.

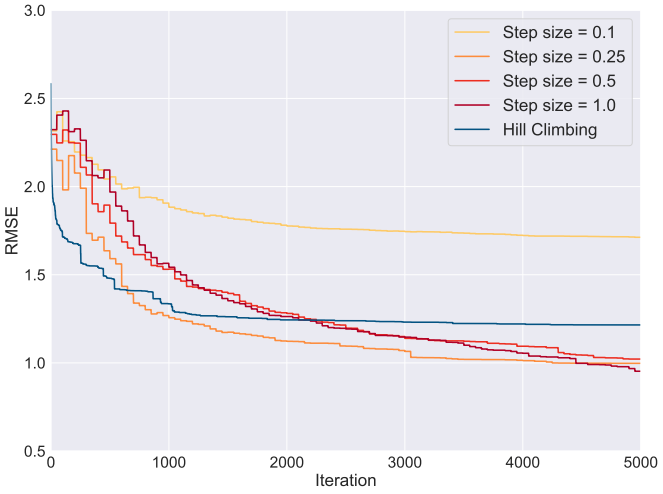

Fig. 5. Convergence of Simulated Annealing algorithms under RMSE compared to the base Hill Climbing model. $N = 5000$, $L = 50$, $T = (1, 10^{-3}$ with quadratic cooling and varying step size for Simulated Annealing. $N = 5000$ and $s = 0.5$ for Hill Climbing. Average convergence over 25 replications.

Subsequently, we attempt to determine an optimal value for the length of the Markov Chains that most effectively finds good solutions. Whereas $L = 50$ delivers decently accurate results, outperforming under several values for s as shown in Figure 5, the solution could be improved by either shortening (more exploration) or extending (more exploitation) the MC lengths. Figure 6 presents convergence of the SA algorithm under multiple settings for L. Increasing L from 50 to 100 does not show any noticeable change in performance. Lowering the value for L towards 40 improves SA performance compared to HC, but further decrements for L to 25 do not show any effect.

The final SA settings we need to explore are related to the actual annealing process and the values for the temperature over time. As discussed, variation can be applied on start and final temperatures, as well as the decay in between them. Initial experiments show very weak performance for final temperatures deviating from 10^{-3} so this value for $T_{\frac{N}{L}}$ is kept constant. We start by picking $T_0 = 1$ because initial

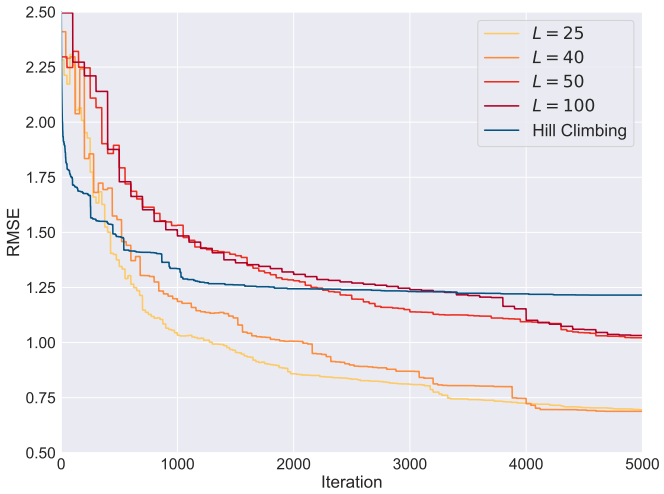

Fig. 6. Convergence of Simulated Annealing algorithms under RMSE compared to the base Hill Climbing model. $N = 5000$, $s = 0.25$, $T = (1, 10^{-3}$ with quadratic cooling and varying Markov Chain lengths for Simulated Annealing. $N = 5000$ and $s = 0.5$ for Hill Climbing. Average convergence over 25 replications.

experiments show good convergence for this value and we use this value to compare the different cooling schedules presented in Figure 2. As expected, linear cooling performs very weakly compared to the exponential and quadratic schemes, and the quadratic cooling schedule shows to perform optimally under RMSE convergence in Figure 7. Increasing or decreasing the starting value for T by a factor 10 seems to have only little effect, and we therefore stay with our initial setting of $T_0 = 1$.

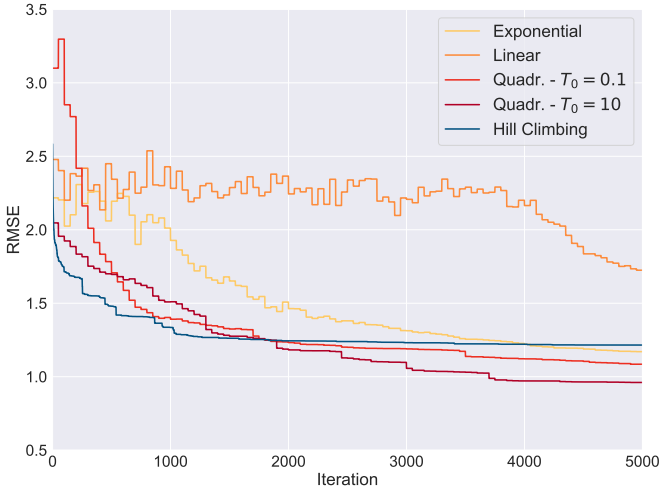

Fig. 7. Convergence of Simulated Annealing algorithms under RMSE compared to the base Hill Climbing model. $N = 5000$, $L = 50$, $s = 0.25$, $T_{\frac{N}{L}}$ with varying T_0 and cooling schedule for Simulated Annealing. $N = 5000$ and $s = 0.5$ for Hill Climbing. Average convergence over 25 replications. Note that the temperature experiments are performed under quadratic cooling, as this shows to perform optimally.

The experiments described above attempt to find optimal parameter settings for SA to estimate the parameters of our LV model, it however only shows the average convergence over 25 replications and does not say anything about the distribution of our results. It also only shows results under RMSE optimisation and evaluation and does not say anything about MAE values. Table I provides an overview of the most important findings from experiments specified above, presenting the mean and standard deviation of the RMSE and MAE for final optimal solutions.

Model	L	s	**RMSE** (σ)	**MAE** (σ)
SA	40	0.25	0.688 (0.734)	0.511 (0.426)
	50	1.0	0.953 (0.443)	0.754 (0.280)
	50	0.25	0.998 (0.815)	0.436 (0.453)
HC	-	0.5	1.215 (0.679)	0.221 (0.221)

TABLE I
AVERAGE RMSE AND MAE AND THEIR STANDARD DEVIATIONS BETWEEN BRACKETS, COMPUTED OVER 25 REPLICATIONS OF THE SIMULATIONS. SA AND HC MODELS ARE PRESENTED WITH VARYING SETTINGS FOR L AND s.

The RMSE scores are sorted in decreasing order, reaching an optimal value for the SA algorithm with $L = 40$ and $s = 0.25$. However, the standard deviations for all the models are very high and no statistically significant differences between the models are observed. Even between the SA and HC models. The SA model with $L = 50$ and an increased s of 1.0 seems to slightly worsen performance on average, but decreases the standard deviation significantly for both performance measures. The hypothesis of equal variance between the top and second-row models can be rejected by an F-test with 99 and 95 percent confidence level for RMSE and MAE respectively. The values of these F-test are 2.75 (compared to $F_{crit,0.99} = 2.66$) and 2.31 (compared to $F_{crit,0.95} = 1.98$).

The most noticeable observation can be made from the column with MAE values. Although no significant statistical evidence can be found because of the high standard deviations, the HC algorithm seems to severely outperform all the SA specifications. We will therefore continue to focus on the RMSE as the optimisation objective, as it suits the problem more adequately.

For the second part of our research, we want to find the effect of decreasing the amount of data used for estimation on final performance. We therefore use a simple scheme and linearly reduce the percentage of data used by 10 percent each time. Datapoints are left out randomly, and this is done by leaving them out for the prey data, the predator data or the complete dataset. These simulations are all performed under the optimal SA model with a Markov Chain length L equal to 40 and a step size s of 0.25, with quadratic cooling and a temperature going from 1 to 10^{-3}. Figure 8 shows average RMSE over 25 replications with the percentage of data used decreasing from 100 to 10. A slight increase in RMSE can be observed for all three curves, but again we can not provide any statistically significant difference between the beginning and end points.

Additionally, we look at a situation where only the peaks or lower parts of the oscillations from the LV equation are used for estimations. The lower parts are specified as data from

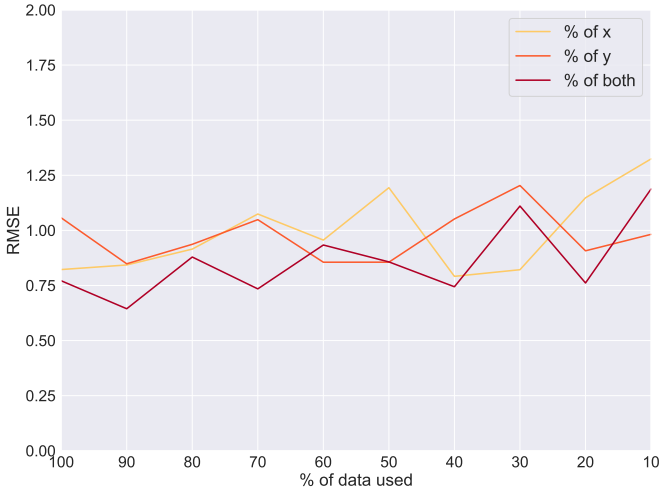

Fig. 8. Evaluation of the average RMSE over 25 replications for the optimal SA model, under a decreasing amount of data for either x, y or both.

every period where x and y are below 2, and the peaks as equally determined for a value higher than 2. This split delivers a sample of approximately 30 percent of the original data for both the peaks and valleys, it can therefore be compared to randomly reducing the data by 30 percent. Again, the optimal SA model is used for simulations and the mean and standard deviation of the RMSE can be observed in Table II. The lowest average RMSE is found under estimation with only the lower parts of the dataset with a value of 0.667. Using only the peaks for estimation shows to change the results severely, increasing the average RMSE. The standard deviations for the estimations are extremely low compared to all the results from before, and statistical evidence can be found for a difference between using only the peaks or only the valleys. Testing for the equality of average RMSE's for using peak and valley data gives a Welch statistic of 25.63, this is way larger than the critical value of the associated T-statistic at the 0.1 percent level of 3.402. Only using data from the lower parts of the dataset can therefore be significantly observed to result in a lower average RMSE than for the peak data.[3]

	RMSE	σ
Only valleys	0.667	0.070
Decreased x & y	0.735	0.719
Only peaks	1.042	0.022
Decreased y	1.049	0.742
Decreased x	1.074	0.753

TABLE II
AVERAGE RMSE AND STANDARD DEVIATIONS, COMPUTED OVER 25 REPLICATIONS OF THE SIMULATIONS. ESTIMATIONS PERFORMED WITH OPTIMAL SA MODEL UNDER APPROXIMATELY 70 PERCENT OF TOTAL DATA.

The basic GA that is described in detail in section II-C is used for simulations under both RMSE and MAE measures.

[3]Note that we do not specifically make this comparison for the other decreased datasets. The standard deviations for these models are approximately equal to the mean and the statistic would therefore never reach a value of $\pm$ 1.7 that is minimally necessary for rejecting hypotheses under any T-distribution.

The complete dataset is used for fitting, so we can create an general comparison to the other optimisation algorithm we have implemented. As the GA contains not one, but a population of solutions, Figure 9 shows convergence of the average error of the population, as well as the error of the best individual solution in the population. This would be the optimal parameter set and is therefore most important. The average error is helpful to determine convergence of the algorithm, it is converged when the average error is approximately equal to the optimal error.

We observe reversed behavior to the SA algorithm, as the MAE objective shows to provide better estimates through the GA algorithm than the RMSE function. The algorithm has converged after 50 generations as the average and best errors are close. The mean error of the algorithm is 0.358 and 0.245 for RMSE and MAE respectively, reaching way lower errors than any of the SA simulations from before. The standard deviations of this algorithm are also improved to be approximately half of the mean (0.156 and 0.109 for RMSE and MAE respectively), even after the little tuning that is applied to the algorithm in this research.

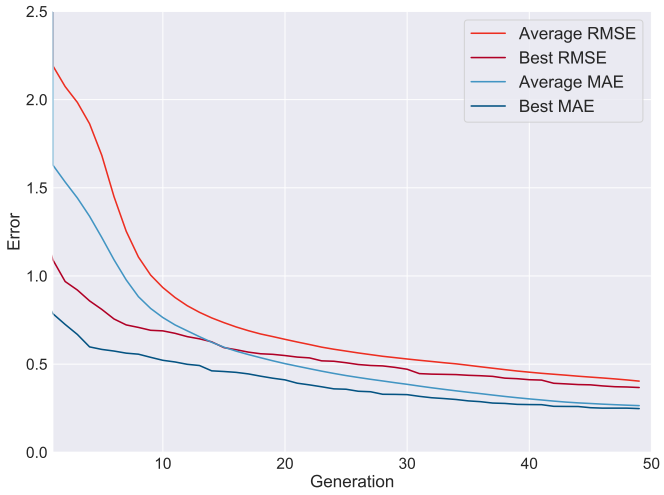

Fig. 9. Converge of a basic specification of a Genetic Algorithm. Providing the error of the best solution and average error for both the RMSE and MAE.

V. CONCLUSION

This research attempts to reverse engineer the parameters of a LV model, fitting the model to an experimental dataset of prey and predator populations. Adequately reverse engineering the LV equations was expected to be an objective with an abundance of local optima, and local optimisation of the LV parameters was expected to perform weakly. This research compares local and global optimisation under objective functions specified by the RMSE and the MAE. The local optimisation algorithm in this research is defined as HC in Algorithm 1 and its more global alternative SA in Algorithm 2.

Experiments are provided to investigate the influence of the parameters in an SA optimisation, ranging from the step size

and the length of the Markov Chain to the cooling schedule and its temperature range. Although difference in performance are observed in these experiments, disproportionately large standard deviations withhold us from making statistically significant conclusions. Under MAE optimisation, we even find the HC algorithm outperforming our SA specifications.

We subsequently examine the effect of reducing the amount of data used for estimation, only using the RMSE as an objective. First, a random (decreasingly sized) sample is taken from the prey data, predator data or the full dataset for estimation, but this approach is inconclusive. We thereafter specify peaks and valleys in the dataset and estimate using only the peaks or only the valleys. We find statistical evidence for reaching a lower average RMSE when optimisation is performed only on the valleys.

As a lot of the experiments in this research leave us inconclusive, we look further than the iterative algorithms in HC en SA and specify a third global optimisation scheme with a GA. This GA shows to severely outperform the earlier specified methods, even after spending little time tuning the parameters of the model. Future research on further specification of these models might be an interesting alternative approach to SA, also because the standard deviations resulting from our GA model are less extreme.

Another approach to improving performance of the SA would be changing the starting values of the optimisation process. The starting parameters could for example be set with a smart start, using another optimisation technique to bring SA (or even HC) close to the actual optimum. This would ideally shorten the necessary number of iterations. Secondly, the starting parameters could be set randomly every replication, but this would likely increase the variance of estimations even further. Even though thorough examination of the SA parameters has been performed in this research, the settings do not proof to be optimally specified yet. This would be an important improvement for the future.

Concludingly, we provide an overall overview of the difference between local Hill Climbing and the more global Simulated Annealing. Under the computational power available, parameters are set optimally. Ideally, the models could be tested under more GridSearch-like circumstances and with a higher number of iterations. Furthermore, other fields of global optimisation seems promising and should certainly be investigated.

REFERENCES

Al-Betar, M. A. (2017). β-hill climbing: an exploratory local search. *Neural Computing and Applications*, 28(1):153–168.

Chai, T. and Draxler, R. R. (2014). Root mean square error (rmse) or mean absolute error (mae)?–arguments against avoiding rmse in the literature. *Geoscientific model development*, 7(3):1247–1250.

Goel, N. S., Maitra, S. C., and Montroll, E. W. (1971). On the volterra and other nonlinear models of interacting populations. *Reviews of modern physics*, 43(2):231.

Lotka, A. J. (1920). Analytical note on certain rhythmic relations in organic systems. *Proceedings of the National Academy of Sciences*, 6(7):410–415.

Martín, J. F. D. and Sierra, J. M. R. (2009). A comparison of cooling schedules for simulated annealing. In *Encyclopedia of Artificial Intelligence*, pages 344–352. IGI Global.

Sharp, K. and Matschinsky, F. (2015). Translation of ludwig boltzmann's paper "on the relationship between the second fundamental theorem of the mechanical theory of heat and probability calculations regarding the conditions for thermal equilibrium" sitzungberichte der kaiserlichen akademie der wissenschaften. mathematisch-naturwissen classe. abt. ii, lxxvi 1877, pp 373-435 (wien. ber. 1877, 76:373-435). reprinted in wiss. abhandlungen, vol. ii, reprint 42, p. 164-223, barth, leipzig, 1909. *Entropy*, 17:1971–2009.

Takeuchi, Y. (1996). *Global dynamical properties of Lotka-Volterra systems*. World Scientific.

Willmott, C. J. and Matsuura, K. (2005). Advantages of the mean absolute error (mae) over the root mean square error (rmse) in assessing average model performance. *Climate research*, 30(1):79–82.

Yang, J. (2018). Non-linearity and local optimum mindset.