# Optimizing Structural Properties Of Neural Networks With Genetic Algorithms

T. Staudt, E. Schultheis*
*University of Göttingen*
(Dated: today)

In this article we examine how structural properties of neural networks, like the size or local synapse densities, affect their learning success for various tasks. To do so, we look at a rate based model for neural networks and apply the FORCE rule for the learning process. A sophisticated matching algorithm allows us to quantify the learning success of a network so that the fitness of structural characteristics, expressed by integer or floating-point parameters, can be evaluated. We then use evolutionary optimization methods on these parameters to show that (1) ring topologies are generally more successful than strictly random topologies, (2) *FAT LIST OF INCREDIBLE RESULTS THAT CHANGE THE UNIVERSE*

Keywords: genetic algorithms, evolutionary algorithms, FORCE learning, rate networks

## INTRODUCTION

Looking at the field of artificial neuronal networks, one finds a great variety of different models describing both the dynamic of networks as well as their plasticities, i.e. their learning behavior. But, given some dynamics and plasticity rules, which features of a network determine whether it is a good candidate for doing a specific task?

Using this question as a guideline, we decided to analyze which high level properties of a neuronal network are important for its learning success under a variant of the FORCE algorithm [1, Architecture A], and how to choose their values for an optimal result.

Instead of fine tuning every single internal weight, we were thus concerned with the general rules of the network's properties. And we wanted our networks to be random to a certain degree, which seems biologically plausible; but we also wanted to allow for structural specialization: How densely are the synapses distributed? How strong shall the feedback signal be, and how strong should the synapses fire on average?

To address these issues in the context of genetic algorithms, we introduce *network genotypes*, which contain parametric information about the networks (*phenotypes*) to be created: Different phenotypes of the same genotype may have strong differences in detail, but they share structural qualities of interest. Improving these genotypes for a specific range of tasks via genetic optimization is the main concern of this article.

We first lay out the main aspects of our model in a rather high-level overview, followed by more details on the actual implementation. Finally, the obtained results are presented and discussed.

## CONCEPTS AND MODEL

*Network Model* We chose the same rate based network architecture that is used in the original FORCE publication [1] (therein denoted as architecture A) and also in [2], where our notation mainly stems from. So we look at a network $\mathcal{N}$ with $N$ internal neurons and states $x_i$ for $1 \leq i \leq N$ that loosely represent the neurons' membrane potential. The firing rate $r_i$ of the $i$-th internal neuron is given by $r_i = \tanh x_i$. Furthermore, the network contains a readout neuron with the state

$$z = \sum_{j=1}^{N} \omega_j^{\text{read}} \, r_j \tag{1}$$

where $\omega^{\text{read}}$ denotes the readout weight vector. Taking into account both internal dynamics as well as an external feedback pathway, the dynamics $t \mapsto x_i(t)$ of the single neurons are governed by [3]

$$\dot{x}_i(t) = -x_i(t) + \sum_{i=1}^{N} \omega_{ij}^{\text{rec}} \, r_j(t) + \sum_{i=1}^{R} \omega_{ij}^{\text{fb}} \, z_j(t) \ , \tag{2}$$

where we introduce the internal recurrent synapse weights $\omega^{\text{rec}}$ and the feedback synapse weights $\omega^{\text{fb}}$. We solved this system of differential equations by Euler integration with a time steps $\mathrm{d}t = 0.01$, which was also used in [1].

*Learning* Learning took place by stepwisely adapting the weights $\omega^{\text{read}}$ in such a way that the readout dynamic $t \mapsto z(t)$ should eventually match a periodic target pattern $t \mapsto z'(t)$, henceforth called a *task*, as accurately as possible. To update $\omega^{\text{read}}$ we chose variant A of the FORCE algorithm [1], which rapidly modifies the feedback loop to keep the errors $|z(t) - z'(t)|$ small from the beginning on.

While the authors of [1] update the weights $\lambda^{\text{read}}$ in fixed intervals, we used an adaptive mechanism to determine how often learning should occur.

*Network Genotypes* In order to find out which structural elements of networks are important for learning sucess, we introduce *network genotypes*. Technically, a genotype $G$ with parameters $\Theta$ may be understood as a stochastic network generator that returns a network $\mathcal{N} = G(\lambda)$ for every seed value $\lambda$.

The genotypes' parameters $\theta \in \Theta$ are either integer or floating point values and can be classified as affecting (1) the topology of $\omega^{\text{rec}}$ or (2) the weight values of $\omega^{\text{rec}}$ and $\omega^{\text{fb}}$. Optimized parameters of type (1) are the occupation probability $p$ for sparse random topologies (Erdös-Renyi topology), the neighborhood range $k$ for ring topologies, and a ratio parameter $q$ used to interpolate between Erdös-Renyi and ring topologies. Optimized parameters of type

(2) are the feedback strength `feedback` $\propto \omega^{\text{fb}}$ and the gain value `gain` $\propto \omega^{\text{rec}}$.

*Fitness* The next step was to define the *fitness* of a given genotype when confronted with a range of different tasks, e.g. waves of different frequencies. To express this formally, we introduce *challenges*: For a given seed value $\lambda$, the challenge $C$ yields a task $z' = C(\lambda)$. If we now apply a success function $S(\mathcal{N}, z') \in \{0, 1\}$ that decides whether the network $\mathcal{N}$ has successfully reproduced the desired output $z'$, the fitness $F_C(G)$ of a genotype $G$ for $C$ may be defined as the expectation value of $\lambda \mapsto S(G(\lambda), C(\lambda))$. For fixed challenges, the main problem of optimizing the genotype thus is to find parameters $\Theta$ that maximize $F_C$.

## METHODS AND IMPLEMENTATION

*High and Low Frequency Challenges* Our concrete goal for this article was to examine which choice of genotype parameters are optimal for learning sinusoidal waves with relatively high or low frequencies. Based on preliminary simulation results we chose $\nu \in []$ as frequency range for the challenge $C_{\text{high}}$, producing high frequency waves, and $\nu \in []$ for $C_{\text{low}}$, producing low frequency waves. Optimizing the fitness functions $F_{C_{\text{high}}}$ or $F_{C_{\text{low}}}$ thus results in genotypes with the desired frequency properties.

*Success* To actually compute the fitness, we need a function $S$ that decides whether a network succeeded to solve a task or not. To this end, a measure $f(z', z)$ for the concordance of the target function $t \to z'(t)$ and the network readout $t \to z(t)$ is sought. Though one could use a simple difference based metric, this has the drawback that plain phase mismatches could produce high differences for properly matching shapes.

A measure that is time-shift independent and maximized for functions of identical shape, on the contrary, is the maximum of the cross correlation

$$f_{\text{cor}}(z, z') = \max_t \frac{\langle z, z'(\cdot - t)\rangle}{\sqrt{\langle z, z\rangle \langle z', z'\rangle}} \ . \tag{3}$$

Algorithmically, the signals $z$ and $z'$ are split into overlapping chunks which are correlated separately. This saves computational power and prevents minute differences in frequency to accumulate into a huge phase shift, so the phase only has to remain relatively constant over the timescale of a single chunk (which is about 10 seconds of simulated time).

Using this measure of quality, the success function for a network $\mathcal{N}$ with readout $z$ for the task $z'$ was (by empirical means) taken to be

$$S(\mathcal{N}, z') = \begin{cases} 1, & \text{if } f_{\text{cor}}(z, z') \geq 0.95, \\ 0, & \text{if } f_{\text{cor}}(z, z') < 0.95 \ . \end{cases} \tag{4}$$

*Genetic Optimization* Due to the stochastic nature of a genotype $G$'s fitness $F_C(G)$, a very high number of samples would be required to numerically obtain sufficiently smooth approximations of $F_C(G)$ for typical optimization methods like gradient descend or simulated annealing. But as the computational cost for even one learning process is considerably high, these methods were not feasible.

Therefore, we optimized $F_C$ using a genetic algorithm, which is inherently capable of coping with the stochastic fitness function. In each iteration of the algorithm, a generation $\mathcal{G}_t = \{G_i \mid 1 \leq i \leq M\}$ of $M$ individuals (Genotypes) is converted into a new one, $\mathcal{G}_{t+1}$, according to the following scheme:

**Evaluation:** The fitness $F_i = F_C(G_i)$ of each individual is measured using at least 50 samples (and adaptively more if the $(F_i)_{1 \leq i \leq N}$ are spaced closely, to at most 100).

**Selection:** We choose 50% of the most successful members of $G_t$ as survivors. With a small chance, a weaker generator can survive, in order to keep the gene pool more diverse.

**Reproduction:** We then replace the $(1-p)|G_t|$ elements that did not make it into the next generation by either *mutating* or *recombining* the survivors and get $\mathcal{G}_{t+1}$. This means we either change a single parameter randomly or take two individuals and randomly choose for each parameter from which parent to take the value.

## RESULTS

### Optimizing Genotypes For Frequency Ranges

Since we wanted to examine how the ideal genotypes vary for different types of problems, we examined how the choice of a frequency range affects the optimized genotype parameters. To this end we looked at sparse random networks (Erdös-Renyi topology with connection probability $p$) and conducted optimizations of the parameters `gain`, `feedback` and $p$ for both high ($C_{\text{high}}$) and low ($C_{\text{low}}$) frequency challenges (see FIG. 3).

The most striking result is the strong growth of the feedback for $C_{\text{high}}$ to values higher than 14, while the value for $C_{\text{low}}$ converges quickly to approximately 1 (FIG. 3 (b)). This behavior may indicate that the feedback pathway must overshadow the (maybe too slow) internal mechanics for high frequencies, while it is the other way round for low frequencies. Indeed, FIG. 3 (a) shows that genotypes with a lower gain value (on average) are more successful for high frequencies, which seems to fit in: excessive recurrent dynamics with a long time scale (compared to the period to be learned) may disturb the learning.

Another interesting aspect is the branching process of $p$ that takes place for $C_{\text{low}}$ in FIG. 3 (c). Albeit the branch seems to be dying out towards the end, it suggests, that very high $p$ values don't ruin the networks capacity to learn low frequencies. This behavior was observed neither for high frequencies nor for optimizations over the full frequency regime.

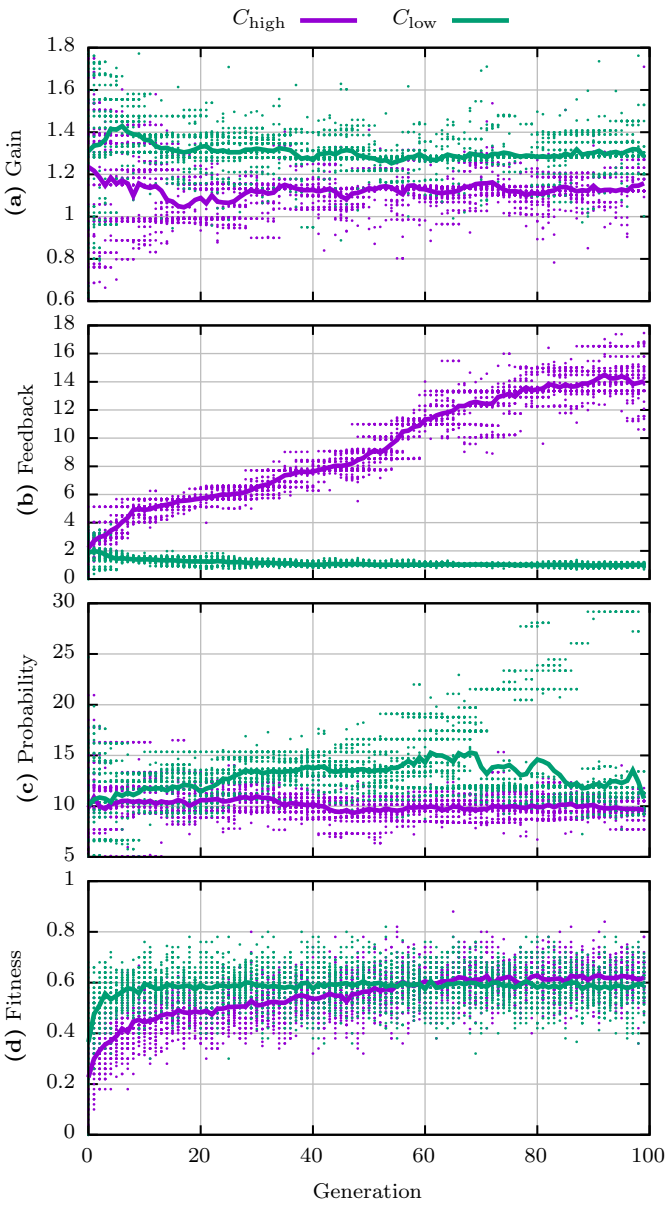


FIG. 1. Genetic optimization process of genotypes with Erdös-Renyi topology ($N = 100$ fixed) for the challenges $C_{\text{high}}$ and $C_{\text{low}}$. In each figure, one can see the 50 genotypes of every Generation (single dots) together with the respective mean values (thick lines). Figures (a) to (c) depict the evolution of the three parameters (`gain`, `feedback`, and $p$) being optimized, while figure (d) shows the development of the fitness values.

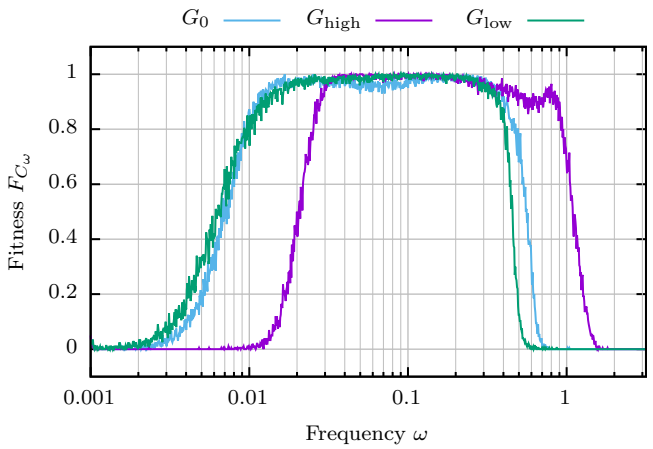


FIG. 2. The covered frequency range for an unoptimized genotype $G_0$ as well as genotypes $G_{\text{high}}$ and $G_{\text{low}}$ that are optimized for $C_{\text{high}}$ respectively $C_{\text{low}}$. Each of the three genotypes was tested against constant challenges $C_\omega$ that yielded simple sinusoidal waves with frequency $\omega$ for every seed value. The topology of all genotypes was a mere Erdös-Renyi topology. The parameters for the unoptimized genotype $G_0$ were taken to be default values from [1] (which also uses Erdös-Renyi topologies), while we chose genotypes $G_{\text{high}}$ and $G_{\text{low}}$ that did particularly well when explicitly optimizing for high and low frequencies (see FIG. 3 for more details).

But how well was the fitness actually improved during the optimization? When looking at FIG. 3 (d), one sees that the fitness for $C_{\text{high}}$ increased during nearly the whole process, while the algorithm failed to further optimize the genotypes' fitness for $C_{\text{low}}$ after about 20 generations. A major reason for this is probably, that the initial population for the $C_{\text{low}}$ optimization already contained genotypes with parameters similar to the (at least locally) optimal ones.

One can also look at the success of the optimization algorithm in a more specific way: FIG. 2 shows how well an unoptimized genotype $G_0$ handles fixed-frequency challenges when compared to the genotypes $G_{\text{high}}$ and $G_{\text{low}}$ optimized for $C_{\text{high}}$ and $C_{\text{low}}$. Here it becomes clear that $G_{\text{high}}$ can indeed succeed at higher frequencies than $G_0$, but that it also fails at learning lower frequencies – which was expected. On the contrary, while the high frequency capacities of $G_{\text{low}}$ have decreased when compared to $G_0$, the low frequency capacities have only marginally improved. This reinforces the impression that the optimization was unable to produce genotypes that can cope with low frequencies.

### Topology Optimizations

To find out whether the topology can be used to improve the learning, we performed an optimization in which the networks were an interpolation between sparse random connections (Erdös-Renyi) and a ring topology where the $k$ nearest neighbors are connected (for non integer $k$, the fractional part determines the probability to use $[k]$ or $[k] + 1$). Since we are interested in the influence of the topology, the parameters gain and feedback are kept constant during the optimization process, leaving the connection probability for ER topology $p$, the ring connection length $k$ and the interpolation value $r$ between ER and ring as free parameters.

The results of the optimization are depicted in FIG. **??**. Unfortunately, the improvement in fitness after the first few generations in extremely low. The development of the ration parameter, however, unambiguously shows that the

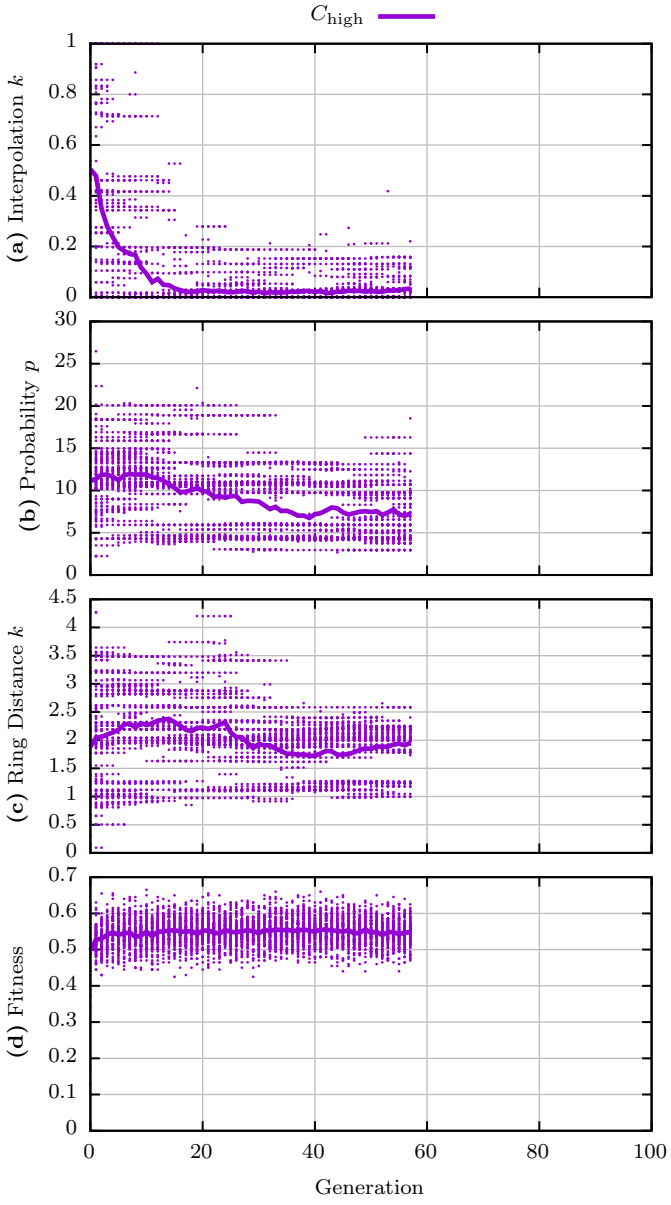


FIG. 3. Genetic optimization process of genotypes with Erdös-Renyi topology ($N = 100$ fixed) for the challenges $C_{\text{high}}$ and $C_{\text{low}}$. In each figure, one can see the 50 genotypes of every Generation (single dots) together with the respective mean values (thick lines). Figures (a) to (c) depict the evolution of the three parameters (`gain`, `feedback`, and $p$) being optimized, while figure (d) shows the development of the fitness values.

selection process prefers generators that consist mostly of ring topology with ring parameter $k \approx 1$. The ER connection probability shows a very broad distribution, which is simply due to the low dependence of the generated networks on $p$ when the usage of ER topology $r$ approaches zero.

## DISCUSSION

Our investigation showed that the frequency of the wave that is to be reconstructed strongly determines whether FORCE learning is able to learn that curve. Still, a good choice of network parameters is still able to improve the success rate for certain frequency ranges. Some parameters ($p$, $feedback$) can vary in a relatively large interval without influencing the success very much, whereas others are confined to a relatively small interval (gain).

The code developed for this paper provides a framework which can be used to perform a variety of similar investigations, e.g. the inclusion of feed forward topologies, or the extension to task other than simple sinusoidals. To that end, it would be wise to improve that selection algorithm by using the same tasks for all generators in each generation, instead of evaluating them all independently. This would make the selection process more robust because it removes the dependence on the task distribution.

One problem with the approach presented here is that, in order to use the results of the genetic algorithm, we determined a single set of parameters by taking the mean of the values of the most successful generators, any information about an interdependence between the parameters is lost. We checked that there are no strong correlations between any pair of parameters by doing 3D plots that showed the two parameters for each generator in development with the generations and could not detect any obvious correlation, but have not investigated this problem thoroughly.

The biggest problem with the current implementation, however, is the large amount of "magic" numbers, i.e. hyper-parameters of the optimization, that are used in the code. Their values were determined by an informal search of testing a few optimizations. However, this should mostly affect the performance of the optimization algorithm and not the results themselves.

* thomas.staudt@stud.uni-goettingen.de, erik.schultheis@uni-goettingen.de

[1] D. Sussillo and L. F. Abbott, Neuron **63**, 544 (2009).
[2] G. M. Hoerzer, R. Legenstein, and W. Maass, Cerebral Cortex **24**, 677 (2014).
[3] Note that we did not include external input signals in this formula, as is done in [1] and [2]. We in fact enabled inputs in our code base, but did not use them for the main results.