Optimizing Structural Properties Of Neural Networks With Genetic Algorithms

T. Staudt, E. Schultheis*
University of Göttingen
(Dated: August 7, 2015)

In this article we examine how structural properties of neural networks, like the local synapse density, affect their learning success for various tasks. To do so, we look at a rate based model for neural networks and apply a learning algorithm called FORCE to teach the network to reproduce either low or high frequent sinusoidals. A sophisticated matching algorithm allows us to quantify the learning success of the networks so that the fitness of structural characteristics (expressed via parameters) can be evaluated. We then use evolutionary optimization methods on these parameters, and find that the feedback pathway may be strengthened in order to improve the learning results for high frequencies but at the cost of decreasing the ability to memorize low frequencies. Alternatively, using a ring topology model instead of random connections has a similar effect, yet still retains the ability to learn lower frequencies better.

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 ω^{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 ω^{rec} and the feedback synapse weights ω^{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 ω^{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 ω^{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 ω^{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 success, we introduce *network genotypes*. Technically, a genotype G with parameters Θ may be understood as a stochastic network generator that returns a network $\mathcal{N} = G(\lambda)$ for every seed value λ.

The genotypes' parameters $\theta \in \Theta$ are either integer or floating point values and can be classified as affecting (1) the topology of ω^{rec} or (2) the weight values of ω^{rec} and ω^{fb}. Optimizable parameters of type (1) are the

occupation probability p for sparse random topologies (Erdös-Renyi topology), the neighborhood range k for ring topologies (where either $\lfloor k \rfloor$ or, with probability $k - \lfloor k \rfloor$, $\lfloor k \rfloor + 1$ nearest neighbors are connected), and a ratio parameter q used to interpolate between Erdös-Renyi and ring topology. Optimizable parameters of type (2) are the feedback strength `feedback` $\propto \omega^{\mathrm{fb}}$ and the gain value `gain` $\propto \omega^{\mathrm{rec}}$.

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 λ, 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 Θ that maximize F_C.

METHODS AND IMPLEMENTATION

High and Low Frequency Challenges Our concrete goal for this article was to examine which choices of genotype parameters are optimal for learning sinusoidal waves with relatively high or low frequencies. Based on preliminary simulation results we chose $\nu \in [0.3, 3]$ as frequency range for the challenge C_{high}, producing high frequency tasks, and $\nu \in [0.004, 0.04]$ for C_{low}, producing low frequency tasks. Optimizing the fitness functions $F_{C_{\mathrm{high}}}$ or $F_{C_{\mathrm{low}}}$ will thus yield 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 timeshift independent and maximized for functions of identical shape is the maximum of the cross correlation

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

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

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

Genetic Optimization Due to the stochastic nature of a genotype'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:

1. 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 tightly spaced, to at most 100).

2. 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.

3. Reproduction: We then replace the $|G_t|/2$ elements that did not make it for 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 every parameter from which parent to take the value from.

RESULTS

Optimizing Genotypes For High And Low Frequencies

Since we want to analyze how the ideal genotypes vary for different types of problems, we examine how the choice of either low or high frequencies affects the optimized genotype parameters. We first look at networks with Erdös-Renyi topology, for which we conduct optimizations of the parameters `gain`, `feedback` and p for both high (C_{high}) and low (C_{low}) frequency challenges (see FIG. 1).

The most striking result is the strong growth of the feedback for C_{high} to values higher than 14, while the value for C_{low} converges quickly to approximately 1 (FIG. 1 (b)). This behavior may indicate that the feedback pathway must overshadow the (maybe too slow) internal mechanics for high frequencies, while there is a contrary effect for low frequencies. Indeed, FIG. 1 (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.

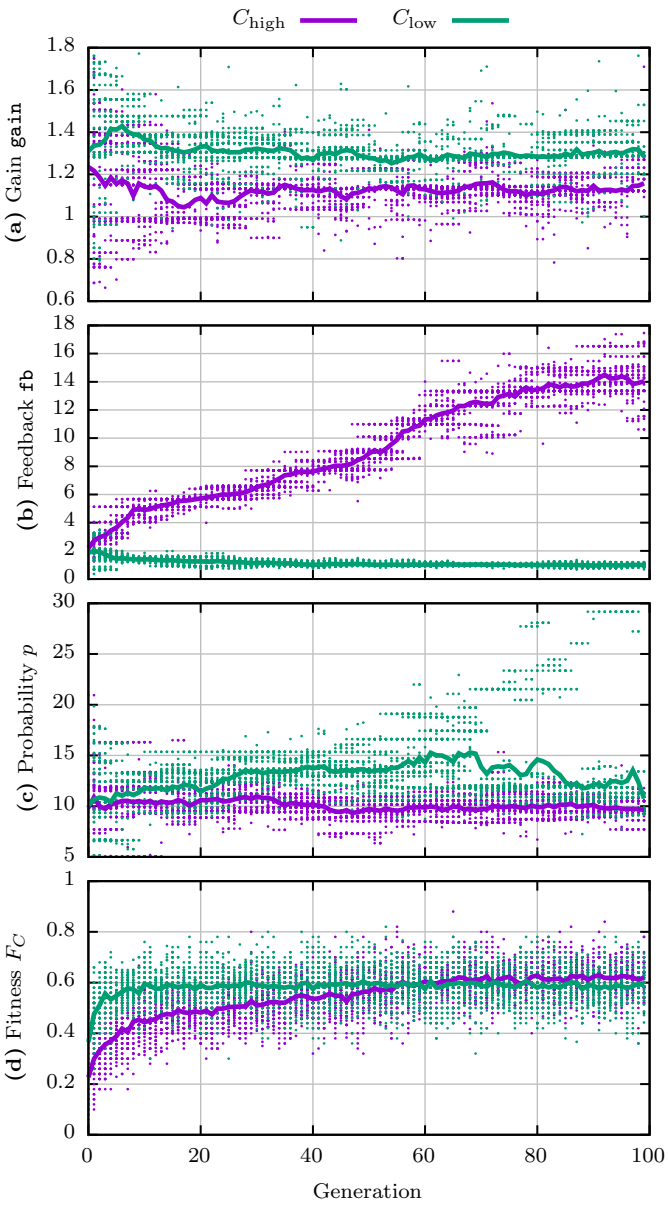

FIG. 1. Genetic optimization process of genotypes with Erdös-Renyi topology ($N = 100$ fixed) for the challenges C_{high} and C_{low}. In each figure, one can see the 50 genotypes of every Generation (single dots) together with the respective mean values (thick lines). Figures (a) to (c) depict the evolution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

Another interesting aspect is the branching process of p that takes place for C_{low} in FIG. 1 (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.

But how well was the fitness actually improved during the optimization? When looking at FIG. 1 (d), one sees that the fitness for C_{high} increased during nearly the whole process, while the algorithm failed to further

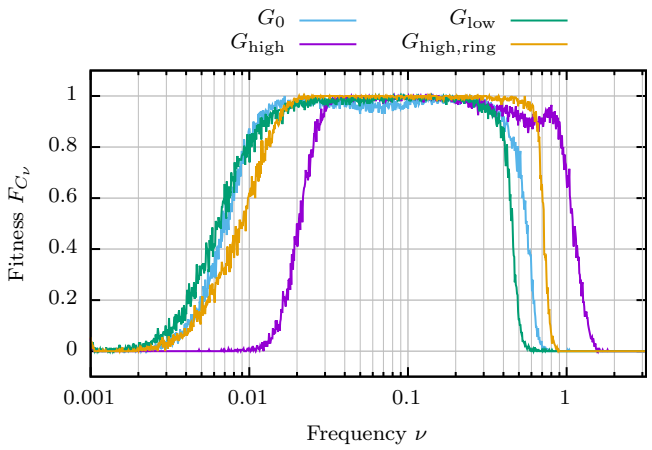

FIG. 2. The fitness values of four differently obtained genotypes when tested against deterministic challenges C_ν that yield simple sinusoidal waves with frequency ν for every seed value. Here G_0 is an unoptimized genotype with parameters taken from [1], whereas the genotypes G_{high} and G_{low} did particularly well during the optimization for C_{high} and G_{low} depicted in FIG. 1. While these three genotypes had strict Erdös-Renyi topologies, $G_{\text{high,ring}}$ is a successful sample of an optimization for C_{high} under an interpolation of Erdös-Renyi and ring topology (see FIG. 3 in the next section).

optimize the genotypes' fitness for C_{low} after about 20 generations. A major reason for this is probably, that the initial population for the C_{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_{high} and G_{low} optimized for C_{high} and C_{low}. Here it becomes clear that G_{high} can indeed succeed at higher frequencies than G_0, but that it also fails at learning lower frequencies – which was expected. However, while the high frequency capacities of G_{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 significantly better than the initial population.

Topology Optimizations

To find out whether the topology may be used to improve the learning for high frequencies without increasing the feedback, we performed an optimization in which the network topologies were an interpolation between Erdös-Renyi and ring topology. For this simulation, the parameters `gain` $= 1.5$ and `feedback` $= 2$ are kept constant during the optimization process, leaving the connection probability p for the Erdös-Renyi topology, the ring neighbourhood range k and the interpolation value q between Erdös-Renyi and ring topology as free parameters.

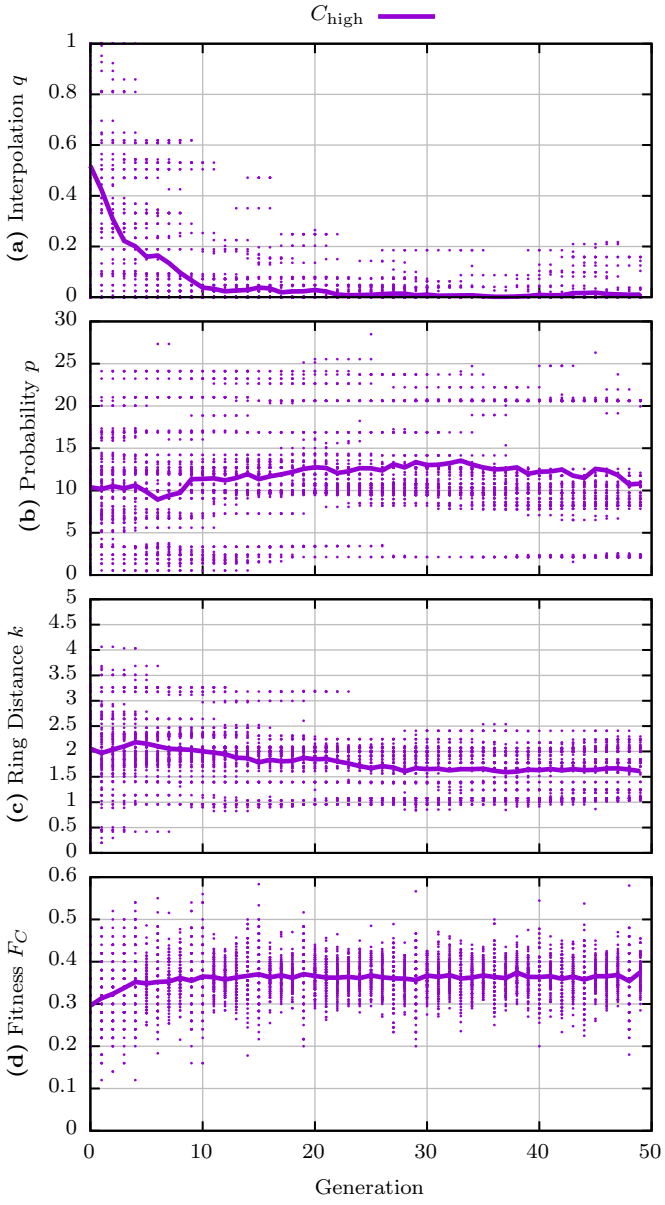

FIG. 3. Genetic optimization of genotypes with an fraction q of Erdös-Renyi topology and $1-q$ of ring topology for the challenge C_{high}. This figure is to be understood like FIG. 1.

The results of the optimization are depicted in FIG. 3. Unfortunately, the improvement in fitness after the first few generations is extremely low. The development of the interpolation parameter q, however, unambiguously shows that the selection process prefers generators with a high fraction of ring topology and with $k \approx 1.5$. The Erdös-Renyi connection probability p shows a very broad distribution, which is due to the weak dependence of the generated networks on p when q approaches zero.

The fitness curve of $G_{\text{high,ring}}$ in figure 2 confirms, that though G_{high} is still better for high frequencies due to the strong feedback, the ring topology provides an improvement compared to the Erdös-Renyi topology when optimizing for C_{high}.

DISCUSSION

While we only presented results concerning simple sinusoidals of varying frequencies here, the code developed during our project provides a framework which can be used to perform a variety of similar and extended investigations; for example the inclusion of feed forward topologies and arbitrary (higher dimensional) tasks with optional external input pathways (e.g. as a timer signal to make the network synchronize better).

One peculiarity to be recognized throughout our project work is that the optimization algorithm does not excel at providing strictly converging, single most optimal parameter values, but that we are rather given more or less broad "survival ranges" for each parameter as result. And though the final genotypes varied quite a bit in their fitness, we did not observe strong correlations between the different parameters in these ranges when doing sporadic tests.

Indeed, looking at all of our simulations in their entirety, the genotype parameters showed less influence on the learning results than we initially expected, and during our inquiry it often seamed that the statistics of the challenges might overshadow the effect of the parameters – at least when looking at more complex tasks than monofrequent waves. A wise improvement would therefore be to make the challenges more deterministic, e.g. to use the same tasks for all genotypes in each generation instead of drawing them independently from some (wide range) distribution.

Another issue to be mentioned with the current implementation is the large amount of empirical ("magic") numbers, i.e. hyper-parameters of the optimization used in the code. Examples hereof are the strength of the mutation process or the mechanics of the selection process (actual number of survivors, survival chances for weaker genotypes, etc.). Their values were determined by an informal search of testing a few optimizations but were not investigated thoroughly. This should, however, mostly affect the performance of the optimization algorithm and not the results themselves.

Nevertheless we were able to produce successful optimization results and could indeed show that adaption to high frequencies can not only be accomplished by an implausible effect, namely the high feedback values, but also by changing the topological structure of the network. In this regard it would certainly be interesting to further inquire how genotypes that allow for a more fine grained control over the topology behave under the optimization.

* 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 e.g. [2].