Published in:
Neurocomputing 36 (1–4), 225–233, 2001

Optimization of Neural Field Models

Christian Igel Wolfram Erlhagen Dirk Jancke

Institut für Neuroinformatik, Ruhr-Universität Bochum, 44780 Bochum, Germany
Tel: +49 234 32 25558
Christian.Igel@neuroinformatik.ruhr-uni-bochum.de

Abstract

There is a growing interest in using dynamic neural fields for modeling biological and technical systems, but constructive ways to set up such models are still missing. We discuss gradient-based, evolutionary and hybrid algorithms for data-driven adaptation of neural field parameters. The proposed methods are evaluated using artificial and neurophysiological data.

Key words: dynamic neural fields; gradient-based optimization; evolutionary optimization; population representation; primary visual cortex of cat

1 Introduction

Neural fields (NFs) have been introduced as mathematical descriptions of cortical neural tissues, where information processing takes place in the form of excitation patterns [22, 1, 12, 7]. They permit a systematic treatment of dynamical processes not only in distributed neural representations, but also in the context of the dynamical system approach to perception and behavior, behavior based robotics, computer vision and sensor fusion, see [17, 8, 19] for further references. However, up to now there exists no constructive way to set up NF models. In order to clarify and substantiate experimental findings, an expert has to adjust the parameters of the NF. Since the fields reveal a highly non-linear dynamic behavior, this is a non-trivial task even for qualitative modeling. The choice of an appropriate set of parameters is an important problem also for technical applications, where NFs have to be designed to perform a given task, e.g., the control of robot behavior [4, 5].

The most common techniques for data-driven parameter adaptation of recurrent neural networks (RNNs) are gradient-based methods [6, 14]. In this study, we adopt these algorithms for adjusting the parameters of NFs. For comparison, we apply an elaborated evolution strategy [9, 10] to NF optimization. We present a hybrid algorithm that combines the local search of gradient-based methods with the more volume-oriented search of evolutionary algorithms.

2 Neural Field Model

We restrict our investigation to a one-dimensional NF formulation based on the model first analyzed by Amari [1]. However, the extension of the optimization algorithms to higher dimensional fields and coupled systems of NFs is straightforward.

Let $u_x(t)$ describe the average membrane potential at time t of a neuron located at site x of the field. Its average activity (mean firing rate) is given by the transfer function $f[u_x(t)]$, which is assumed to be of sigmoidal shape, $f(u) = \alpha/(1 + \exp[\beta(u - u_{\text{shift}})])$. Each neuron x gets time varying input $s_x(t)$ from outside the field and via lateral connections from each of its field neighbors. The average strength, $w(x, x')$, of the connection between any two neurons x and x' is assumed to be a function of their relative distance only, $w(x, x') = w(|x - x'|)$, i.e., the field is homogeneous and the interactions are symmetric. Considering a finite population of n neurons we get the following coupled system of differential equations:

$$\tau\, \dot{u}_x(t) = -u_x(t) + h + \sum_{x'=1}^{n} w(|x - x'|) f[u_{x'}(t)] + s_x(t) \tag{1}$$

for $x = 1, \ldots, n$, where $\tau > 0$ defines the time constant of the dynamics and $h > 0$ the resting level to which the potentials u_x relax in absence of any input. Throughout this study, the interaction kernel is given by the difference of two Gaussians

$$w(x) = \frac{g_{\text{ex}}}{\sqrt{2\pi}\sigma_{\text{ex}}} \exp\left(\frac{-x^2}{2\sigma_{\text{ex}}^2}\right) - \frac{g_{\text{inh}}}{\sqrt{2\pi}\sigma_{\text{inh}}} \exp\left(\frac{-x^2}{2\sigma_{\text{inh}}^2}\right) \quad . \tag{2}$$

and external stimuli are assumed to have Gaussian profiles with variance parameter σ_s and amplitude parameter g_s. Altogether, the described model has $m = 9$ parameters $\boldsymbol{\theta} = (g_{\text{ex}}, \sigma_{\text{ex}}, g_{\text{inh}}, \sigma_{\text{inh}}, \tau, \alpha, \beta, u_{\text{shift}}, h)'$.

The described NF can process information in different modes, e.g., the resulting activation pattern can either decay back to resting level after the external input has disappeared or it can be stable due to lateral interactions. Figure 1 shows an example time course of an NF dynamic, where the field changes from a stable homogeneous resting state to a stable localized solution. Analytical results prove the existence of different types of solutions [1, 13], but do not describe the actual spatiotemporal pattern.

3 Gradient-based Optimization

The goal is to adapt the NF parameters such that starting from the relaxed field for all $t \in [0, T]$ and $x = 1, \ldots, n$ either the potential $u_x(t)$ or the average activity $f[u_x(t)]$ follows a desired trajectory $d_x(t)$ for a given stimulus pattern $s_x(t)$. Therefore, we define an error function $E = \int_0^T E(t)\mathrm{d}t$ and perform gradient descent on $E(t)$ with respect to the parameters $\boldsymbol{\theta}$. For each parameter θ_i, $i = 1, \ldots, m$, we have to calculate

$$\frac{\mathrm{d}E}{\mathrm{d}\theta_i} = \int\limits_{t=0}^{T} \sum_{x=1}^{n} e_x(t) \frac{\mathrm{d}u_x(t)}{\mathrm{d}\theta_i} \mathrm{d}t \qquad \text{with} \qquad e_x(t) = \mu_x(t) \frac{\partial E(t)}{\partial u_x(t)} \;, \tag{3}$$

where the masking function $\mu_x(t) \equiv 1$ if data at time t and position x is available and zero otherwise. A common choice for E is the sum-of-squares error function: If target trajectories of the potentials u_x are given, we have $E(t) = \sum_{x=1}^{n} \mu_x(t) \, (u_x(t) - d_x(t))^2$ and $e_x(t) = \mu_x(t) \, 2(u_x(t) - d_x(t))$. For firing rates $f[u_x(t)]$ we get $E(t) = \sum_{x=1}^{n} \mu_x(t) \, (f[u_x(t)] - d_x(t))^2$ with $e_x(t) = \mu_x(t) \, 2(f[u_x(t)] - d_x(t)) f'[u_x(t)]$.

From gradient-based optimization of standard recurrent neural networks (RNN), as considered for example in [6, 14], it is known that there are basically two ways to compute the error gradient, one is to solve the system of differential equations describing $\mathrm{d}u_x(t)/\mathrm{d}\theta_i$ for $x = 1, \ldots, n$ and $i = 1, .., m$ *forward in time*, and the other is to run the corresponding adjoint system *backward in time*.

3.1 Forward Method

The derivatives $\mathrm{d}u_x(t)/\mathrm{d}\theta_i$ can be determined by solving the so called variation system forward in time. The variation system that estimates the effect of parameter changes on the NF dynamics is simply derived by differentiating the NF equation (1) with respect to the parameters. For example, for a parameter θ_i $(i = 1, \ldots, 4)$ of the interaction kernel w we have

$$\begin{aligned} \frac{\mathrm{d}}{\mathrm{d}t} \left(\frac{\mathrm{d}u_x(t)}{\mathrm{d}\theta_i} \right) &= -\frac{1}{\tau} \frac{\mathrm{d}u_x(t)}{\mathrm{d}\theta_i} \\ &\quad + \frac{1}{\tau} \sum_{x'=1}^{n} \left[\frac{\partial w(|x - x'|)}{\partial \theta_i} f[u_{x'}(t)] + w(|x - x'|) f'[u_{x'}(t)] \frac{\mathrm{d}u_{x'}(t)}{\mathrm{d}\theta_i} \right] \;. \end{aligned} \tag{4}$$

The initial conditions are given by $\mathrm{d}u_x(0)/\mathrm{d}\theta_i = 0$ for $x = 1, \ldots, n$ and $i = 1, \ldots, m$.

An equation like (4) has to be solved for each parameter for all positions. Thus, the forward method requires $\mathcal{O}(mn^2)$ computations and $\mathcal{O}(mn)$ memory per simulated time-step. Normally $m \ll n$ holds, so the complexity can be regarded as tolerable compared to fully connected RNNs, where m corresponds to the number of weights and equals n^2.

The forward method can be used as an online algorithm which updates the parameters continuously, using $e_x(t)(\mathrm{d}u_x(t)/\mathrm{d}\theta_i)$ to derive $\mathrm{d}\theta_i/\mathrm{d}t$. This method is called real time recurrent learning [21].

3.2 Backward method

In case of RNNs, calculating the error gradient backward in time saves computations. We briefly show that this does not hold for NFs using a parameter θ_i $(i = 1, \ldots, 4)$ of the interaction kernel w as an example. The basic idea of the backward method or *adjoint method*, which can be regarded as a continuous time version of *Backpropagation Through Time* [20], is to simulate the NF forward in time and then to solve an adjoint system of the variation system backward in time, cf. [6, 11]. Let $q_{x\theta_i}(t)$ be the variable belonging to the system adjoint to (4). Then the error gradient can be calculated as

$$\frac{1}{\tau}\int_0^T \sum_{x'=1}^{n} q_{x'\theta_i}(t) \sum_{x''=1}^{n} \frac{\partial w(|x'-x''|)}{\partial\theta_i} f[u_{x''}(t)]\mathrm{d}t = \int_0^T \sum_{x'=1}^{n} e_{x'}(t)\frac{\mathrm{d}u_{x'}(t)}{\mathrm{d}\theta_i}\mathrm{d}t \quad . \tag{5}$$

Because of the non-local nature of the parameters $\boldsymbol{\theta}$, the sums in the left-hand-side of Eq. (5) do not vanish in contrast to the corresponding formula for RNNs (for a similar reason the Green's function approach [18] does not reduce the complexity either). Hence this method requires $\mathcal{O}(mn^2)$ steps compared to $\mathcal{O}(n+m)$ in case of RNNs [14]. As in general the memory requirements scale with the number of simulated time steps and because the algorithm is not suitable for online adaptation, the forward method should be preferred for optimization of NFs.

4 Evolutionary Optimization

4.1 Evolution Strategy

Evolutionary algorithms (EAs) are a class of direct, easy to parallelize optimization methods inspired by natural evolution and have been applied successfully in the field of neural networks [23]. A variety of EAs for parameter optimization exists [3], in our study a state-of-the-art evolution strategy (ES) is employed, namely the $(\mu/\mu_\mathrm{I}, \lambda)$-CMA-ES [9, 10]. The notation $(\mu/\mu_\mathrm{I}, \lambda)$ indicates that the new parent population consists of the μ best of the λ offspring and the use of intermediate recombination involving all parents. CMA stands for covariance matrix adaptation, i.e., the ability of the algorithm to adjust the complete covariance matrix of the Gaussian mutation distribution by means of *self-adaptation*. The CMA implements important self-adaptation concepts, in particular *derandomization* and *cumulation*. Derandomization basically means that the mutation distribution is changed in a deterministic way so that the probability to reproduce steps in the search space that have led to the actual population is increased. Cumulation denotes the use of information from previous generations in the self-adaptation process by considering the search path the population has taken. The CMA-ES is described in detail in [9, 10].

4.2 A Hybrid Algorithm

The ES as a more volume-oriented search method is better suited for exploring the search space than gradient-based algorithms. On the other hand, gradient-based optimization is likely to be superior for local optimization. Therefore, we use a hybrid method, in which the CMA-ES is only used until the error drops below a certain threshold E_{switch}, then the evolved solution serves as the starting point for the gradient-based optimization.

5 Experimental Evaluation

5.1 Artificial Data

The first task is to reproduce a synthetic potential pattern shown in Fig. 1. It is generated by an NF as described by Eq. (1). The model consists of 100 neurons and is iterated for 100 time steps. The parameters are $g_{\mathrm{ex}} = 40$, $\sigma_{\mathrm{ex}} = 10$, $g_{\mathrm{inh}} = 60$, $\sigma_{\mathrm{inh}} = 40$, $h = 3$, $\alpha = 1.2$, $\beta = -1.5$, $u_{\mathrm{shift}} = 0.2$, and $\tau = 5$. Two Gaussian stimuli ($g_{\mathrm{s}} = 20$, $\sigma_{\mathrm{s}} = 10$) are given for the first 50 time steps at the positions 35 and 65, respectively; the activation fuses to a single peak. We assume that stimulus shape, position and duration are known, i.e., we have nine free parameters. As the data was generated by an NF model, a perfect fit can be achieved.

Two sets of trials were conducted for each optimization algorithm. First, the parameters of the NF were initialized uniformly at random subject to $g_{\mathrm{ex}}, g_{\mathrm{inh}} \in [10, 100]$, $\sigma_{\mathrm{ex}} \in [3, 50]$, $\sigma_{\mathrm{inh}} \in [\sigma_{\mathrm{ex}}, 50]$, $h \in [-5, 0]$, $\alpha \in [1, 2]$, $\beta \in [-10, -0.5]$, $u_{\mathrm{shift}} \in [-0.1, 0.1]$, and $\tau \in [1, 10]$. In the second experimental setup, the initial parameters were chosen uniformly at random in $\pm 10\%$ of the corresponding optimal values. This was done in order to analyze the search performance in the vicinity of the optimal solution. 200 independent trials were performed per experimental setup. The trials were stopped either when the mean squared error fell below 10^{-6} or when the *training progress* as defined by [15] measured after a *training strip* of length 10 dropped below 10^{-3}. Before presentation of the stimulus pattern, the field equation is iterated until the NF has relaxed, but for at most $T_{\mathrm{ini}} = 100$ iterations.

We used the BFGS algorithm [16] for gradient-based optimization. The line search involved did not rely on gradient information. The ES employed was a $(2/2_{\mathrm{I}}, 6)$-CMA-ES. In the hybrid algorithm we set $E_{\mathrm{switch}} = 0.1$.

The results are shown in Fig. 2, which plots the success rate versus the computational effort. A trial is regarded as successful iff the mean squared error fell below 10^{-6}. The computational effort of calculating the error was defined as 1. Based on the consideration in Sec. 3, the effort of deriving the gradient was set to $m + 1$, i.e., determining the gradient corresponds to 10 fitness evaluations.

The success rate of the solely gradient-based strategy is significantly lower (after effort of 20000, χ^2-test, $p < .001$) than of the other methods, whereas the slight difference between the success rates of the evolutionary and the hybrid algorithm is not statistically significant. In case of the random initialization, only four trials of the BFGS and 25 of each other strategy succeeded in finding a model with an error below 10^{-6}.

The BFGS finds the optimal—and also gets trapped in the non-optimal—solutions considerably faster than the ES. The hybrid algorithm is much faster than the pure ES, but still worse than the BFGS. Considering the successful trials from both initialization methods, the average computational effort until the error dropped below 10^{-6} was 1503.45 for the BFGS, 3291.299 for the hybrid algorithm, and 5378.89 for the ES (all differences are significant, Wilcoxon rank sum test, $p < .001$).

5.2 Neurohysiological Data

We use single cell activity recorded in primary visual cortex of cat to demonstrate how our approach optimizes the parameters of an NF to account for real-world data. A neural population representation for the horizontal position of a stimulus in visual space has been reconstructed by means of an optimal linear estimator, see [12, 7] for details. It is based on the spiking activity of 178 neurons in response to a flashed small spot of light.

The population representation consists of a monomodal activity distribution in visual field coordinates, which gradually builds up and decays back to resting level. Since the neuronal distributions are given as a function of the represented variable *horizontal position in visual space*, the one-dimensional field is defined over this dimension. To optimize the parameters of the NF, we have chosen the pattern of maximum activity of the population representation as a target distribution, see Fig. 3. To account also for the temporal evolution of the neural representation the parameters of a more complex field model have to be adapted [12].

The best result out of 10 trials with a mean squared error of 0.002 is shown in Fig. 3.

6 Discussion

Gradient-based techniques can be adopted for NF optimization, where forward methods should be preferred. Our experiments show that the optimization of NFs is a difficult task; the algorithms tend to get trapped in local minima. In particular, bifurcations in the NF dynamics exacerbate the search process. These difficulties can partly be overcome by incorporating additional expert knowledge, e.g., by ensuring that necessary conditions for a desired type of solution are met. Nonetheless, in our modeling task based on neurophysiological data gradient-based adjustment of NF parameters has proven to be successful.

The observed problem of local minima is in accordance with the results reported in [2] for RNNs employed for spatiotemporal neural modeling. In contrast to RNNs, NFs have the advantage to be homogeneous by definition, therefore only single prototypes are needed as training patterns. In addition, the high-level parameterization of NFs allows for a comparatively easy interpretation of the system.

In our experiments, evolutionary optimization is slower than gradient-based optimization (although the gradient cannot be calculated as efficiently as in the case of RNNs), but finds the optimal solution more often. It can be made faster by switching to gradient-based optimization after the basin of attraction of the optimum has been found. However, a more sophisticated coupling of gradient-based optimization and evolution could further improve the performance.

Acknowledgment

This work was supported by the *Ministerium für Schule und Weiterbildung, Wissenschaft und Forschung des Landes Nordrhein-Westfalen* under Grant 516 – 102 001 98.

References

[1] S. Amari. Dynamics of pattern formation in lateral-inhibition type neural fields. *Biological Cybernetics*, 27:127–132, 1977.

[2] K. Arai, S. Das, E. L. Keller, and E. Aiyoshi. A distributed model of the saccade system: simulations of temporally perturbed saccades using position and velocity feedback. *Neural Networks*, 12:1359–1375, 1999.

[3] T. Bäck and H.-P. Schwefel. An overview of evolutionary algorithms for parameter optimization. *Evolutionary Computation*, 1(1):1–23, 1993.

[4] T. Bergener, C. Bruckhoff, P. Dahm, H. Janßen, F. Joublin, R. Menzner, A. Steinhage, and W. von Seelen. Complex behaviour by means of dynamical systems for an anthropomorphic robot. *Neural Networks*, 12:1087–1099, 1999.

[5] E. Bicho, P. Mallet, and G. Schöner. Target representation on an autonomous vehicle with low-level sensors. *International Journal of Robotics Research*, 19:424–447, 2000.

[6] K. Doya. Recurrent networks: Supervised learning. In M. A. Arbib, editor, *The Handbook of Brain Theory and Neural Networks*, pages 796–800. MIT Press, 1995.

[7] W. Erlhagen, A. Bastian, D. Jancke, A. Riehle, and G. Schöner. The distribution of neural population activation as a tool to study interaction and integra-

tion in cortical representations. *Journal of Neuroscience Methods*, 94:53–66, 1999.

[8] M. A. Giese. *Dynamic Neural Field Theory for Motion Perception*. Kluwer Academic Publishers, 1999.

[9] N. Hansen and A. Ostermeier. Convergence properties of evolution strategies with the derandomized covariance matrix adaptation: The $(\mu/\mu, \lambda)$-CMA-ES. In *EUFIT'97, 5th Europ. Congr. on Intelligent Techniques and Soft Computing*, pages 650–654. Verlag Mainz, Wissenschaftsverlag, 1997.

[10] N. Hansen and A. Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation, special issue on self-adaptation*, 2000. to appear.

[11] P. Hartman. *Ordinary Differential Equations*. John Wiley & Sons, Inc., Boston, 1964.

[12] D. Jancke, W. Erlhagen, H. R. Dinse, A. C. Akhavan, M. Giese, A. Steinhage, and G. Schöner. Parametric population representation of retinal location: Neural interaction dynamics in cat primary visual cortex. *The Journal of Neuroscience*, 19(20):9016–9028, 1999.

[13] K. Kishimoto and S. Amari. Existence and stability of local excitations. *Journal of Mathematical Biology*, 7:303–318, 1979.

[14] B. A. Pearlmutter. Gradient calculations for dynamic recurrent neural networks: A survey. *IEEE Transactions on Neural Networks*, 6(5):1212–1228, 1995.

[15] L. Prechelt. Early stopping – but when? In G. B. Orr and K.-R. Müller, editors, *Neural Networks: Tricks of the Trade*, volume 1524 of *LNCS*, chapter 2, pages 57–69. Springer-Verlag, 1999.

[16] W. Press, S. Teukolsky, W. Vetterling, and B. Flannery. *Numerical Recipes in C*. Cambridge University Press, 2. edition, 1994.

[17] G. Schöner, K. Kopecz, and W. Erlhagen. The dynamic neural field theory of motor programming: Arm and eye movements. In P. Morasso and V. Sanguineti, editors, *Self-Organization, Computational Maps and Motor Control*, pages 271–310. Elsvier Science, 1997.

[18] G.-Z. Sun, H.-H. Chen, and Y.-C. Lee. Green's function method for fast online learning algorithm of recurrent neural networks. In J. E. Moody, S. J. Hanson, and R. P. Lippmann, editors, *Neural Information Processing Systems*, volume 4, pages 333–340. Morgan Kaufmann Publishers, 1992.

[19] J. G. Taylor. Neural 'bubble' dynamics in two dimensions: foundations. *Biological Cybernetics*, 80:393–409, 1999.

[20] P. J. Werbos. Backpropagation through time: What it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560, 1990.

[21] R. J. Williams and D. Zipser. A learning algorithm for continually running fully recurrent neural networks. *Neural Computation*, 1(2):270–280, 1989.

[22] H. R. Wilson and J. D. Cowan. A mathematical theory of the functional dynamics of cortical and thalamic nervous tissue. *Kybernetik*, 13:55–80, 1973.

[23] X. Yao. Evolving artificial neural networks. *Proc. of the IEEE*, 87(9):1423–1447, 1999.

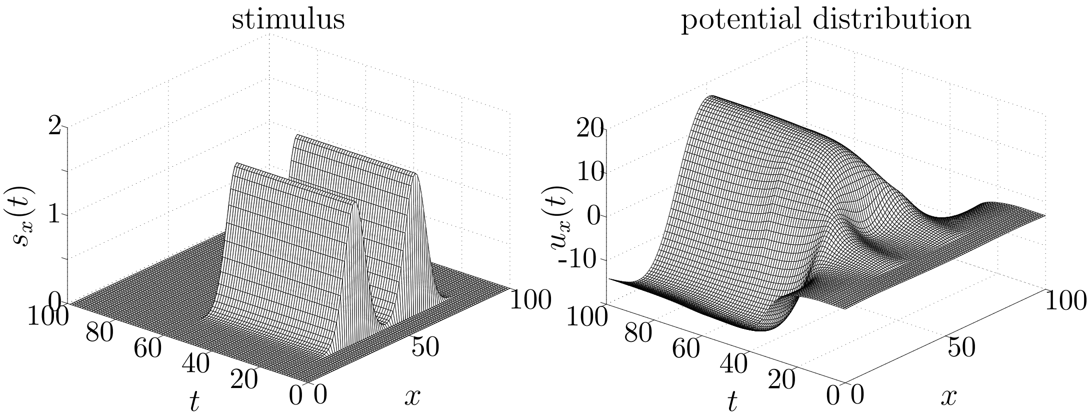

Fig. 1. A bimodal input pattern (left) presented for 50 time steps merges to a monomodal potential distribution (right) due to the cooperative interactions in the field.

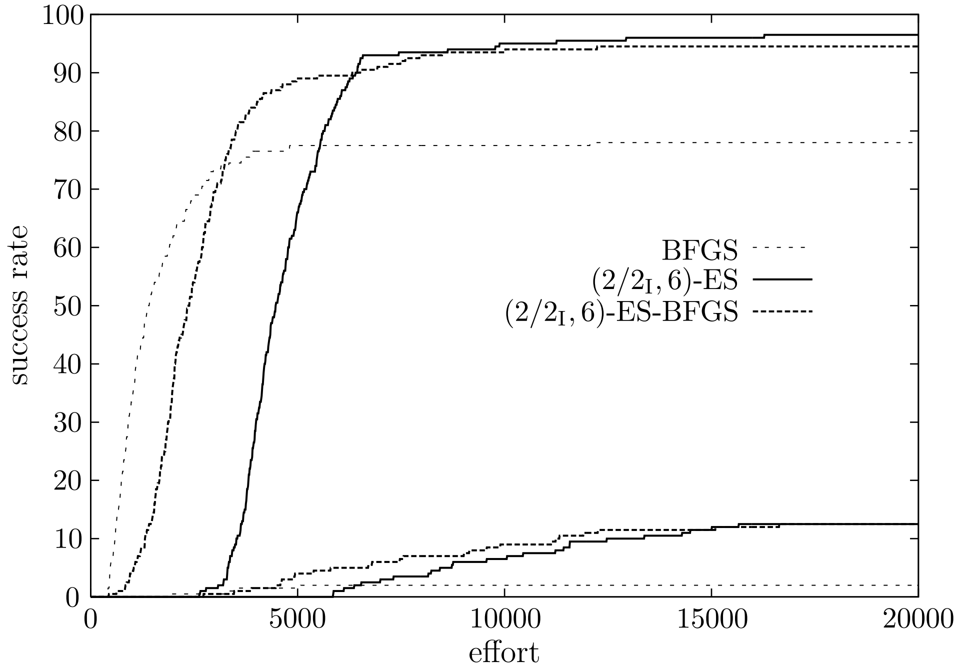

Fig. 2. Success rates versus computational effort for the different optimization algorithms, where the upper lines correspond to trials initialized in the vicinity of the optimum and the lines below to random initialization.

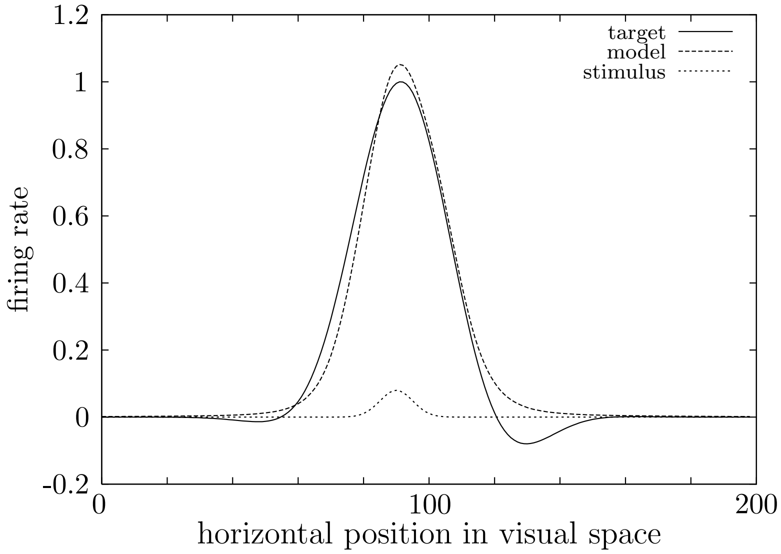

Fig. 3. A Gaussian stimulus with $g_s = 1$ and $\sigma_s = 5$ triggers the evolution of an activity pattern. After a fixed time interval, the activity of the NF model is compared with the reconstructed neural activity distribution.