Unsupervised Learning of Echo State Networks: A Case Study in Artificial Embryogeny

Alexandre Devert, Nicolas Bredeche, and Marc Schoenauer

TAO team - INRIA Futurs - LRI/CNRS, Bat 490 - Université Paris-Sud - France

Abstract. Echo State Networks (ESN) have demonstrated their efficiency in supervised learning of time series: a "reservoir" of neurons provide a set of dynamical systems that can be linearly combined to match the target dynamics, using a simple quadratic optimisation algorithm to tune the few free parameters. In an unsupervised learning context, however, another optimiser is needed. In this paper, an adaptive (1+1)-Evolution Strategy is used to optimise an ESN to tackle the "flag" problem, a classical benchmark from multi-cellular artificial embryogeny: the genotype is the cell controller of a Continuous Cellular Automata, and the phenotype, the image that corresponds to the fixed-point of the resulting dynamical system, must match a given 2D pattern. This approach is able to provide excellent results with few evaluations, and favourably compares to that using the NEAT algorithm (a state-of-the-art neuro-evolution method) to evolve the cell controllers. Some characteristics of the fitness landscape of the ESN-based method are also investigated.

1 Introduction

Neural Networks (NNs) can be used to tackle a variety of problems. In classification or regression problems, some examples of inputs/outputs of the network are available during the learning phase: the training is *supervised*, and the fitness function is generally some Mean Square Error (MSE) between the network outputs and the actual outputs over the know examples. On the other hand, *unsupervised learning* regards cases where no such examples are available. When the network is used as a component of some computational model for a physical process, an explicit optimisation criterion (or *oracle*) is nevertheless available: the optimal network is the one for which the model reaches a target behaviour. Typical examples of such situation include control problems (e.g. robotics), and engineering inverse problems. Finally, the optimisation criterion can be implicit, and rely on some internal stabilisation of the network under external stimulation, as for instance in the case of Kohonen maps [24], where the task is to cluster pattern examples.

Two crucial programmer's decisions impact the choice of a learning method to train the network: the type of topology – feedforward or recurrent, i.e. without or with loops in the connection graph – and the choice of what will be learnt – the weights of a fixed topology, or both the topology and the weights.

For static problems, feedforward NNs will more likely be used, because the resulting learning problem is generally easier, while dynamical problems (where data k depends on data $k-1, k-2, \ldots$) will bias the choice toward recurrent NNs, that include some memory of the past in the activations of their internal neurons. However, feedforward NN fed with multiple past inputs can also capture the essence of dynamical systems, sometimes better than recurrent NNs [8].

Regarding the choice of the free parameters, very powerful methods are available to learn the weights of a fixed topology (e.g. for supervised learning, the back-propagation algorithm for feedforward [32] or recurrent [30] NNs), that would favour such approach. However, the choice of the topology itself then only relies on the programmer's expertise, and a poor guess will hinder the whole process. On the other hand, whereas learning both the weights and the topology opens up a much larger search space, the best topology can stay out of reach of the chosen learning method. The versatility and robustness of Evolutionary Algorithms make them perfect candidates for this latter task (see Section 2 for a brief survey).

Recently, however, an alternative to topology learning for recurrent NNs was proposed in the context of supervised learning of dynamical systems, namely the prediction of times series: the Echo State Network (ESN) approach [19] turns the search for the best topology into the search for the best combination of many randomly connected neurons: if sufficiently many different dynamics are present in the *reservoir* of neurons, then any dynamical system can be approximated by a linear combinations of those dynamics [29]. The learning problem is thus quadratic, and can be solved rapidly and efficiently by any standard optimisation method (more details in Section 3). However, this straightforward approach is limited to supervised learning, and very few attempts, if any, have tried to use ESNs in other contexts.

This paper proposes to use Evolutionary Computation to train an ESN in the context of unsupervised learning with explicit optimisation criterion, more precisely, to find the best controller of the cells of a multi-cellular approach to embryogenic design. As said above, training an ESN amounts to learning a vector of real parameters. However, because the context is unsupervised, no gradient-based approach applies. Moreover, because of the huge number of different dynamics that exist in the reservoir, it is expected that the fitness landscape will be quite rough. Hence Evolutionary Algorithms seem to be a good choice here. Unfortunately, an additional difficulty arises from the number of weights to learn: even though only the output weights are to be learnt (see again Section 3), the need for a large reservoir usually requires dozens or hundreds of internal neurons, resulting in as many weights per output of the network to be adjusted. Hence special care must be taken when choosing the optimisation method that will adjust those weights.

The paper is organised as follows : section 2 gives an overview on the state of the art of NN learning, focusing more precisely on evolutionary methods, and detailing in particular the NEAT algorithm [33], that constructively evolves the topology of a recurrent NN. Section 3 briefly introduces the Echo State

Networks and their (supervised) training before detailing the proposed approach for unsupervised optimisation of the weights of the ESN using an adaptive (1+1) Evolution Strategy. Section 4 describes the multi-cellular artificial embryogeny benchmark problem. Such problem has been addressed by the authors in a recent work using NEAT [9]. Those latter results are the baseline for the experimental validation of the proposed ESN-based approach, in section 5, where the fitness landscape is also studied in more detail. Finally, conclusions and further directions of research are given in section 6.

2   Artificial Evolution of Neural Network

The start of the Golden Age for Neural Networks was the invention of the backpropagation algorithm for supervised learning of feed-forward networks in the early 80's [32]. Several improvements have been proposed since then (e.g. variants of gradient methods for accelerated convergence, modification of the method for supervised learning of recurrent networks [30]). However, while theoretical results proved the representation power of such networks even with a single hidden layer [17], the issue of setting the correct number of hidden neurons for a given task remains open.

Some works addressed this issue by modifying the structure of the neural network prior to, or during, the learning process. Proposed approaches rely on hand-crafting the NN topology [25], pruning arcs or nodes from fully connected NN [26] or even growing NN by adding new nodes during the course of learning [11,2]. In this context, Evolutionary Algorithms (EAs) quickly appeared as a relevant approach towards NN learning (see [35] for a detailed survey). And though they can also be useful for the optimisation of the weights of a feed-forward NN, because backpropagation, as a gradient-based method, can easily get stuck in some local optima, EAs have been mainly used for their flexibility to handle complex search spaces: variation operators acting on the topology can easily be designed.

But evolutionary learning of NNs can be applied as soon as some performance measure for a given network is available, i.e. in both supervised or unsupervised (with an explicit optimisation criterion) context, or for feed-forward as well as for recurrent network topologies. Indeed, evolutionary learning of both the topology and the weights of recurrent NNs has been widely adopted in domains that could benefit from the wide variety of rich dynamical behaviours they offer, e.g. for control problems, in evolutionary robotics . . .

Notable works in this field include: the GNARL approach [1] which uses direct encoding of a neural network for building a robot controller; SANE [28] (Symbiotic Adaptive Neuro-Evolution), ESP [13] (Enforced Sub-population) evolve a population of neurons (rather than network) and combine these neurons to form effective neural networks; GASNET [18] combines optimising the position of neurons in an euclidean space with diffusion of chemicals; Gruau's Cellular Encoding [15,34] and his followers use indirect encoding, evolving a set of instructions that creates the network.

More recently, NEAT (Neuro Evolution of Augmenting Topologies) [33] and AGE (Analog Genetic Encoding) [10] have both been able to provide some relevant results, both in pure performance and speed of convergence, on classical benchmarks such as the double pole balancing. While AGE relies on an approach inspired from Genetic Regulatory Networks [5] where (regulatory) part of the genome encodes information on how to interpret (coding) parts, NEAT uses a direct encoding, as detailed in next section.

2.1 NEAT: Revolution of Augmenting Topologies

The *NEAT* algorithm [33], is an evolutionary neural network optimisation algorithm. It evolves both the topology and the weights of a neural network, either feedforward or recurrent. It relies on a direct encoding of neural network topologies and is based on a specific evolutionary scheme, using different subpopulations to preserver diversity along evolution. The main feature of NEAT is that it explores the topologies from the bottom-up: it starts from an empty network (direct connections from inputs to outputs), and proceeds constructively, using several mutation operators (and no crossover operator) to stochastically add neurons and connections to the networks while preserving as much as possible the behaviour of the network (e.g. new connections have very small weights, new neurons have no connections to start with, ...). Some Gaussian mutation operator modifies the weights so as to fine tune the network.

NEAT has been applied successfully to a wide range of problem, from the classical two pole balancing problem to particle systems rendering [16]. However, it should be noted that direct encoding methods poorly scale up, and sometimes have trouble to catch problem regularities (such as symmetries). Extensions of the original NEAT implementation have been proposed to tackle this issue, such as HyperNEAT [7], which has been successfully applied in an autonomous robotic context. Nevertheless, NEAT is currently one of the (if not "the") state-of-the-art NN optimisation algorithm and will be considered in this paper as the reference algorithm.

3 Echo State Networks

Echo state networks (ESN) have been proposed by Jaeger in 2001 [19,20] with the objective to endow a neural network with rich dynamics behavioural patterns while keeping learning complexity at a low level. An ESN is a discrete time, continuous state, recurrent neural network using a sigmoidal activation function for all neurons. A typical ESN is shown in figure 1, and will be used in this paper: the input layer is totally connected to the hidden layer, both the hidden and input layers are totally connected to the output layer. Moreover, the output layer is connected backward to the hidden layer. In this setup, the hidden layer, or *reservoir*, is randomly generated: N neurons are randomly connected up to a user-defined density of connections ρ. The weights of those connections are randomly set uniformly in $[-1, 1]$, and are scaled so that the spectral radius of
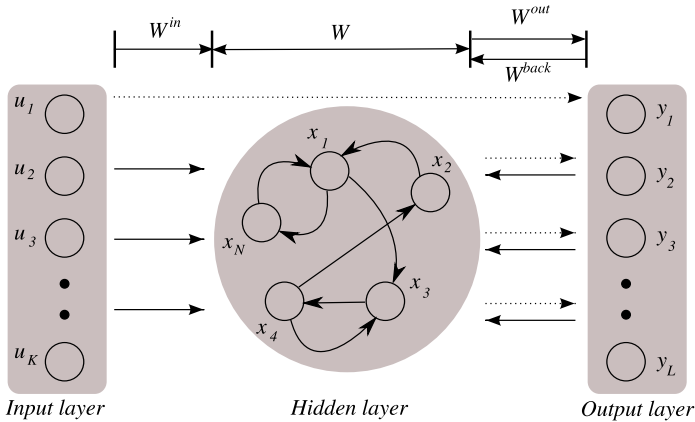
Fig. 1. Schematic view of an Echo State Network. Plain arrows stand for weights that are randomly chosen and remain fixed, while dashed arrows represent the weights to be optimised ($(K + N) \times N$) if N is the number of neurons in the reservoir.

the connection matrix is less than a given value $\alpha < 1$, ensuring that the network exhibits the "echo state property", i.e. stays out of the chaotic behaviour zone whatever the input sequence (see e.g. [21]). The random construction of an ESN is thus determined by the 3 parameters N, ρ and α.

The main point in ESN is that only the weights going from the input and hidden nodes to the output nodes are to be learnt. If the problem has K inputs and L outputs and a reservoir of size N, this amounts to $(K + N) \times L$ free parameters. Moreover, the learning problem is reduced to a quadratic optimisation problem that can be efficiently and quickly solved by any deterministic optimisation procedure, even for very large values of N. In some sense, an ESN can be seen as a universal dynamical system approximator, which linearly combines the elementary dynamics contained in the reservoir [29]. ESN have been shown to perform surprisingly well in the context of supervised learning, in particular for problems of prediction of times series, though it has also been successfully used in the context of (supervised) robot control learning (see [22] for an overview of ESN applications).

3.1   Unsupervised Learning of ESN

It is rather straightforward to replace the default learning algorithm of ESN with any derivative-free optimisers, such as Evolution Strategies or Simulated Annealing, to train an Echo State Network for supervised learning [4]. Yet, to date, and despite its intrinsic properties, ESN has not been applied to unsupervised problem with explicit criteria.

In order to address this class of problems, this unsupervised learning task is turned into an optimisation task: optimising an ESN amounts to optimising a real-valued vector representing the plastic weights of the network (from inputs and reservoir to outputs, see Figure 1). In such situation, Evolution Strategies

(ES) [31,6] provide an efficient and well-grounded framework. However, although only a limited amount of weights have to be optimised, the size of the reservoir may quickly lead to a high dimensional search space depending on the number of outputs, which impacts on the type of ES to be chosen.

This is why an adaptive $(1+1) - ES$ has been chosen here: this simple yet efficient ES scales up well with the dimension of the search space, in term of memory and computation time needed. In this algorithm, the "population" contains only a single individual X_t, that generates a single offspring Y_t using Gaussian mutation as follow :

$$Y_t = X_t + \sigma_t \mathcal{N}(0,1)$$

$\mathcal{N}(0,1)$ is a realisation of the normal probability distribution, and σ_t is the *mutation strength* at time t. The selection is deterministic: X_{t+1} is the best performing individual among $\{X_t, Y_t\}$

The idea behind this $(1+1) - ES$ is to adapt the σ value during the course of evolution with regards to the success of creating better offspring. Following [23], the σ value is updated according to the so-called *one-fifth rule*:

$$\sigma_{t+1} = \begin{cases} 2\sigma_t & \text{if } f(Y_t) < f(X_t) \\ 2^{-\frac{1}{4}}\sigma_t & \text{otherwise} \end{cases}$$

4   Multi-cellular Artificial Embryogeny

The model for Artificial Embryogeny considered here was originally proposed in [9]. It can be viewed as a continuous state discrete space and time cellular automata. *Cells* are placed on a two dimensional regular square grid (the whole grid is filled with cells, no cell division or migration is used). The state of each cell is a continuous value, representing here a grey level. The whole grid, or *organism*, can hence be interpreted as a grey image. Each cell communicates with its 4 neighbours by exchanging some "chemicals": each cell has an internal *controller* (a neural network) that determines its state as well as the amount of chemical it emits at time t toward its neighbours, according to the amount of chemical it received from its neighbours at previous time step $t-1$ (cells on the boundary of the grid don't receive anything from the external environment). Starting from a given state for all cells at time 0, this developmental process is repeated until some stopping condition is reached. The goal is here to reach a target 2D image when the development stops.

The original feature of the proposed model lies in the stopping criterion for the development: whereas previous works used a fixed number of development iterations, this model waits for the organism to stabilise (and penalises individuals whose organism doesn't stabilise within a prescribed number of iterations). The controller used in [9] was a neural network, evolved using the NEAT approach (described in section 2.1). Thanks to the stopping criterion, the evolved organisms exhibited very strong robustness to perturbation: the target image seemed to be the only fixed point of the best organisms considered as dynamical systems,
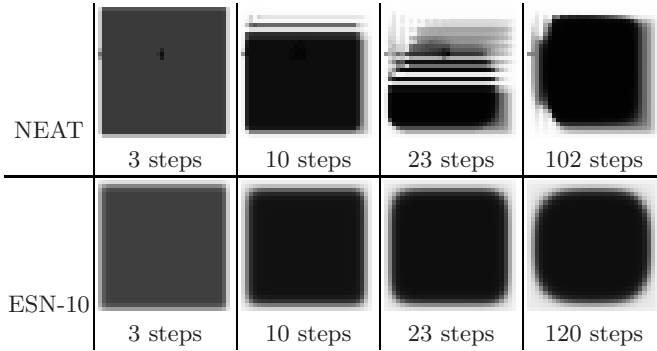
**Fig. 2.** Developmental stages for the *disc* problem. The right-most plots show the fixed-point image.

even though a single starting point was used during evolution. Figure 2 shows an example of a complete development of such result toward the fixed-point shape (the target shape was a black disc on white background). In the following, a maximum number of iterations of 1024 steps is fixed. If the organism has not reached a stable pattern before this limit, its fitness is set to the worst value.

4.1   The Flag Problems

In order to evaluate multi-cellular approaches, it is common to consider matching with simple geometric 2-dimensional images, like "flags" (French and Norwegian flags are quite popular in the literature [27,12]) or other regular patterns [14]). Figure 3 shows the two 32×32 target grey-level images used in this work, respectively called the disc and the half-disc.

The fitness function (to be minimised) is based on the MSE between two images. It takes value in $[0, 1]$, the optimal value is 0 when both images are identical:

$$s(A, B) = \frac{1}{wh} \sum_{i=0}^{h-1} \sum_{j=0}^{w-1} (A(i,j) - B(i,j))^2$$

disc   half-discs
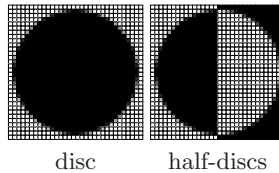
Fig. 3. The two target pictures (with grid lines)

5   Experimental Results

Three algorithms are compared on the two target flags for the embryogenic approach described in previous section: NEAT (results from [9]), and two Echo State Networks with different reservoir sizes.

5.1   Settings

The NEAT implementation used here includes the latest features from the literature. Our implementation has been validated with regards to the original results presented in [33]. NEAT explores recurrent topologies without constraints using a population of 500 individuals while all other parameters are set to default values (see [9] for a detailed information).

Two variants are tested for the Echo State Networks: reservoirs of sizes 10 and 50, with connection factor of respectively 50% and 10%. In both cases, the damping factor (spectral radius) was set to 0.9. These are refered to as 10- and 5-ESN respectively.

The settings for the $(1+1) - ES$ optimiser are as follows : σ_0 is set to 10^{-1}, and the starting point x_0 has all weights set to 0. The algorithm is stopped and restarted (with the same reservoir) whenever $\sigma_t < 10^{-8}$. In any case, the run is stopped when the total number of evaluations reaches 250000. Figure 4 displays the evolution of the fitness for a typical run: the restarts are clearly visible (note that it is a coincidence that the best fitness is reached after the final restart).

Note that the CPU cost of a single evaluation cannot be estimated alone, whatever the algorithm: it of course depends on the reservoir size for ESN, and on the (dynamic) number of neurons for NEAT, but also heavily on the number of developmental steps before stabilisation. Globally, the 16 ESN runs lasted around 2 days for reservoir size 10 and 5 days for reservoir size 50. In contrast, NEAT needed 7 days for the same experimental conditions.
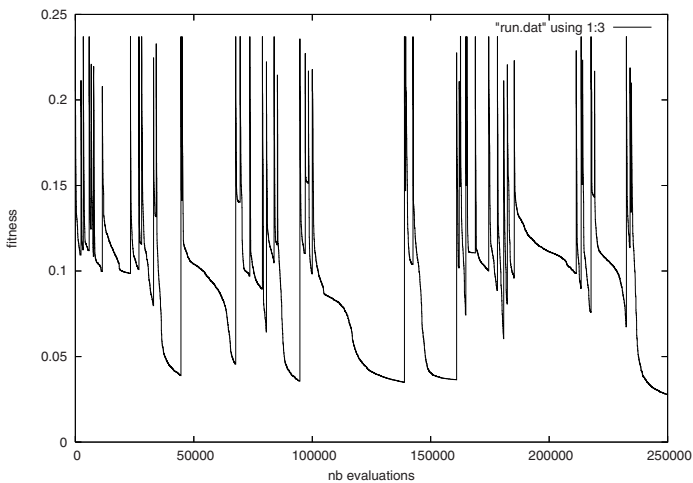

Fig. 4. A typical run for ESN with 10 neurons on the disc problem

## 5.2   Results

On-line results of best-so-far fitness averaged over 16 independent runs are displayed on figures 5. Note that NEAT plot starts after the evaluation of the initial population (of size 500). The corresponding off-line results (i.e. after 250 000 evaluations) are detailed in Table 1.

It is clear that the ESNs outperform NEAT on the *disc* problem, and confirmed by a two-tailed t-test at 99% confidence level. Furthermore, though a bigger reservoir gives more parameters to optimise, it also makes the problem easier: the performances of ESN-10 and ESN-50 are not statistically distinguishable. An important remark is that the results of ESN are much more stable, as witnessed by the standard deviations in Table 1, one order of magnitude smaller for ESN (whatever the reservoir size) than for NEAT.

Table 1. Off-line results out of 16 runs: minimum – average(std. deviation)

	NEAT	10-ESN (1+1)ES	50-ESN (1+1)ES	10-ESN CMAES
Disc	$0.076 - 0.105(0.135)$	**0.021** – **0.030**(0.009)	$0.027 - 0.033(0.008)$	
Half-disc	**0.135** $- 0.201(0.171)$	$0.205 - 0.207(0.002)$	$0.206 - 0.209(0.002)$	$0.184 - $**0.194**$(0.004)$

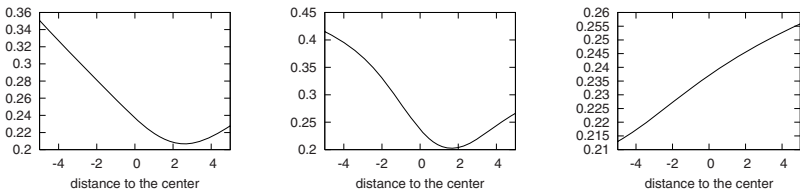(a) The disc problem   (b) The half-disc problem
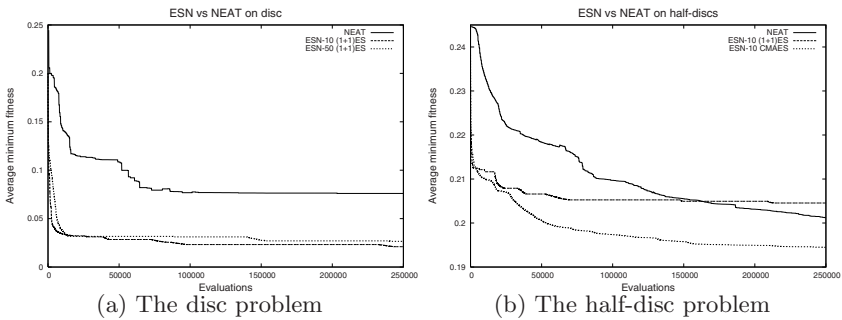
Fig. 5. On-line average minimal fitness reached by NEAT and both 10- and 50-ESN in 250000 evaluations, on the disc and half-disc problems with one chemical. Y-axes have different scales for both problems.

The picture is somewhat different for the *half-discs* problem. The first remark is that the best fitness is much worse for all algorithms than for the disc problem. Moreover, there is no statistically significant difference (whatever the confidence level in a 2-tailed t-test) among the 3 results. However, here again, the standard deviation among the NEAT runs is much larger than among the ESN runs . . . and this does make a difference here when considering the best fitness reached among the 16 runs: NEAT reaches 0.135 while no ESN run can find a better fitness than 0.206. Additional experiments are needed to give this some statistical significance. Nevertheless, this is a typical "design" situation where a large variance is a better indicator of possible good performance for equivalent averages.

5.3   The Fitness Landscape

In order to investigate the characteristics of the fitness landscape, projections on random directions of $\mathbb{R}^{(N+K)\times L}$ (see Figure 1) have been plot, both around the initial point of all optimisation, i.e. with all weights set to 0 (Figure 6) and the best point reached by one of the ESN-10 runs on the disc problem (Figure 7). Whereas the landscape around the initial point seems very smooth and almost convex, that around the final solution looks much more rough. In particular, there exist points very close to the solution that have the worst possible fitness value of 1, i.e. whose development never reaches a fixed point. This suggests that some gradient-based algorithm could be used at the beginning of evolution, but will rapidly stop being efficient when reaching lower-fitness regions, with rougher landscapes.

Fig. 6. Typical sections of fitness landscape for ESN-10 on the disc problem around the initial null point (centre of x-axis)
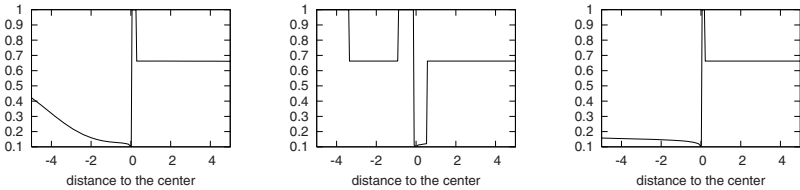
Fig. 7. Typical sections of fitness landscape for ESN-10 on the disc problem around the best individual of one of the successful runs (centre of the x-axis)

6   Conclusion

Echo State Networks are able to perform rich dynamic behavioural patterns with only few real-valued parameters to optimise. This makes ESN a very good choice for classification, regression and time-series prediction, even compared to more complex approaches such as NN weight and topology optimisation algorithms. Yet, to date, applications of ESN have been limited to supervised learning tasks.

This paper has introduced ESN in the context of an unsupervised learning task. The proposed approach combines ESN with a simple yet efficient Evolution Strategy algorithm (a $(1+1)-ES$ implementing the 1/5 rule). Experiments conducted on two benchmark problems from Multi-Cellular Artificial Embryogeny have shown that the proposed approach is competitive with that using NEAT, a state-of-the-art Neural Network topology optimisation algorithm.

ESN clearly outperform NEAT in one of the two problems, have similar performance on the other, and converge much faster in both cases: this confirms both their ability to model complex dynamics and the possible gain due to optimising in a smaller search space. Furthermore, NEAT results have a much larger variance. Whereas this can be thought as a defect demonstrating some lack of robustness, it can also turn out to be an advantage when the average values are comparable, as in the second experiment, as it witnesses the ability for the algorithm to find some very good solution, though very rarely. A deeper statistical study is required to assess (or not) this property.

Further analysis showed that the fitness landscape is very smooth around the initial solution, which might explain the good results in terms of speed of minimisation obtained with such a simple optimiser. This also suggests to try other optimisation strategies, using, or at least starting with, gradient based method in the first steps. At the other end of the process, it seems that the landscape is rather rough close to the (local) optima reached on the flag problems, in part due to the non-stabilisation of the developmental dynamical systems very close to the solution. This in turn suggests to use an adaptive stopping criterion rather than a fixed user-defined number of iterations.

Finally, all experiments have been performed with default parameters, both for generating ESN and tuning the $(1+1)-ES$ optimiser. Results might possibly be improved with a fine tuning of these parameters. Future directions include optimising the meta-parameters implied in the generation of the ESN reservoir, as well as relying on more powerful Evolution Strategy algorithms, such as the state-of-the-art CMA-ES algorithm [3], whenever the size of the reservoir is small enough to make possible.

References

1. Angeline, P.J., Saunders, G.M., Pollack, J.P.: An evolutionary algorithm that constructs recurrent neural networks. IEEE Transactions on Neural Networks 5(1), 54–65 (1994)
2. Ash, T.: Dynamic node creation in backpropagation networks. Connection Science 1(4), 365–375 (1989)
3. Auger, A., Hansen, N.: A restart cma evolution strategy with increasing population size. In: Proceedings of the IEEE Congress on Evolutionary Computation (CEC 2005), pp. 1769–1776 (2005)
4. Babinec, S.: Evolutionary optimization methods in echo state networks. In: 6th Czech-Slovak Workshop on Cognition and Artificial Life (2006)
5. Banzhaf, W.: On the dynamics of an artificial regulatory network. In: Banzhaf, W., Ziegler, J., Christaller, T., Dittrich, P., Kim, J.T. (eds.) ECAL 2003. LNCS (LNAI), vol. 2801, pp. 217–227. Springer, Heidelberg (2003)
6. Beyer, H.-G., Schwefel, H.-P.: Evolution strategies: A comprehensive introduction. Natural Computing: an international journal 1(1), 3–52 (2002)
7. D'Ambrosio, D.B., Stanley, K.O.: A novel generative encoding for exploiting neural network sensor and output geometry. In: Thierens, D., et al. (eds.) GECCO 2007, ACM Press, New York (2007)

8. Dematos, G., Boyd, M., Kermanshahi, B., Kohzadi, N., Kaastra, I.: Feedforward versus recurrent neural networks for forecasting monthly japanese yen exchange rates. Asia-Pacific Financial Markets 3(1), 59–75 (1996)
9. Devert, A., Bredeche, N., Schoenauer, M.: Robust multi-cellular developmental design. In: Thierens, D., et al. (eds.) GECCO 2007, ACM Press, New York (2007), http://hal.inria.fr/inria-00145336/en/
10. Durr, P., Mattiussi, C., Floreano, D.: Neuroevolution with analog genetic encoding. In: PPSN IX, pp. 671–680 (2006)
11. Fahlman, S.E., Lebiere, C.: The cascade-correlation learning architecture. In: Touretzky, D.S. (ed.) Advances in Neural Information Processing Systems, Denver, vol. 2, pp. 524–532. Morgan Kaufmann, San Mateo (1990)
12. Federici, D., Downing, K.: Evolution and development of a multicellular organism: scalability, resilience, and neutral complexification. Artificial Life 12(3), 381–409 (2006)
13. Gomez, F., Miikkulainen, R.: Incremental evolution of complex general behavior. Technical Report AI96-248, 1 (1996)
14. Gordon, T.G.W., Bentley, P.J.: Bias and scalability in evolutionary development. In: GECCO 2005, pp. 83–90. ACM Press, New York (2005)
15. Gruau, F.: Genetic synthesis of modular neural networks. In: ICGA 1993, pp. 318–325. Morgan Kaufmann, San Francisco (1993)
16. Hastings, E., Guha, R., Stanley, K.O.: Neat particles: Design, representation, and animation of particle system effects. In: IEEE CIG 2007 (2007)
17. Hornik, K., Stinchcombe, M., White, H.: Multilayer feedforward networks are universal approximators. Neural Networks 2, 359–366 (1989)
18. Husbands, P., Smith, T., Jakobi, N., O'Shea, M.: Better living through chemistry: Evolving gasnets for robot control. Connection Science 10(3-4), 185–210 (1998)
19. Jaeger, H.: The Echo State approach to analysing and training recurrent neural networks. Technical Report GMD Report 148, German National Research Center for Information Technology (2001)
20. Jaeger, H.: Short term memory in echo state network. Technical Report GMD Report 152, German National Research Center for Information Technology (2001)
21. Jaeger, H.: Tutorial on training recurrent neural networks. Technical report, GMD Report 159, Fraunhofer Institute AIS (2002)
22. Jaeger, H., Haas, H., Principe, J.C. (eds.): NIPS 2006 Workshop on Echo State Networks and Liquid State Machines (2006)
23. Kern, S., Müller, S.D., Hansen, N., Büche, D., Ocenasek, J., Koumoutsakos, P.: Learning probability distributions in continuous evolutionary algorithms: a comparative review. Natural Computing 3(1), 77–112 (2004)
24. Kohonen, T.: Self-Organizing Maps. Springer Series in Information Sciences, vol. 30. Springer, Berlin, Heidelberg (1997) (Second Extended Edition, 1997)
25. LeCun, Y., Boser, B., Denker, J.S., Henderson, D., Howard, R.E., Hubbard, W., Jackel, L.: Backpropagation applied to handwritten zip code recognition. Neural Computation 1(4), 541–551 (1989)
26. LeCun, Y., Denker, J.S., Solla, S., Howard, R.E., Jackel, L.D.: Optimal brain damage. In: Touretzky, D. (ed.) NIPS 1989, Morgan Kaufman, San Francisco (1990)
27. Miller, J.F.: Evolving a self-repairing, self-regulating, french flag organism. In: Deb, K., al., e. (eds.) GECCO 2004. LNCS, vol. 3102, pp. 129–139. Springer, Heidelberg (2004)
28. Moriarty, D.E.: Symbiotic evolution of neural networks in sequential decision tasks. Technical Report AI97-257, 1 (1997)

29. Ozturk, M.C., Xu, D., Principe, J.C.: Analysis and design of echo state networks. Neural Computation 19(1), 111–138 (2007)
30. Pearlmutter, B.A.: Gradient calculations for dynamic recurrent neural networks: A survey. IEEE Transactions on Neural Networks 6, 1212–1228 (1995)
31. Rechenberg, I.: Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution. Frommann-Holzboog, Stuttgart (1973)
32. Rumelhart, D.E., Hinton, G.E., McClelland, J.L.: Exploration in Parallel Distributed Processing. MIT Press, Cambridge (1988)
33. Stanley, K.O., Miikkulainen, R.: Evolving neural networks through augmenting topologies. Evolutionary Computation 10(2), 99–127 (2002)
34. Whitley, D., Gruau, F., Pyeatt, L.: Cellular encoding applied to neurocontrol. In: ICGA 1995, pp. 460–467. Morgan Kaufman, San Francisco (1995)
35. Yao, X.: Evolving artificial neural networks. Proceedings of the IEEE 87(9), 1423–1447 (1999)