Final Project - Computational Neuroscience
Coherent Patterns of Neural Activity

From "Generating Coherent Patterns of Activity from Chaotic Neural Networks" by D. Sussillo and L.F.Abbott (2009)

Joséphine Raugel

June 2020

1 Abstract

In 2009, David Sussillo and L.F. Abbott published an article [1] presenting a procedure called FORCE learning, used to study and take advantage of the relationship between spontaneous and stimulus-triggered neural activity. In this project, we model this procedure on a network controlled only by feedback.

2 Introduction

The strength of FORCE learning resides in the fact that the only way it modifies an initially chaotic generating model network is by a feedback that changes the synaptic strength within or external to the network. In this project, we use the procedure on an network-external synaptic strengths modification model only, with a chaotic network.

3 Architecture of the model

In the model we use, a recurrent generator network with firing rates $\mathbf{r}$ drives a linear readout unit with output z through weights $\mathbf{w}$, see figure 1.

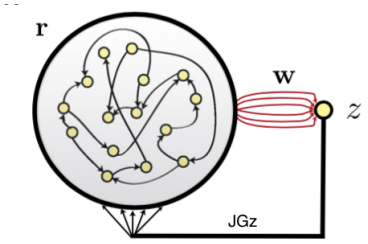

Figure 1: The activation function f

The weights **w**, in red, are the only values modified during training. The only feedback to the generator network is provided by the readout unit z. Firing rates **r** and weights **w** are column vectors that vary over time. The output z is defined by:

$$z(t) = \mathbf{w}^T \mathbf{r}(t) \tag{1}$$

All the network models used are based on firing-rate descriptions of neural activity.The firing rate r_j of each neuron i of the generator network (one out of N_G neurons) is given by:

$$\tau \dot{x}_i = -x_i + g_{GG} \sum_{j=1}^{N_G} J_{ij}^{GG} \underbrace{\tanh(x_j)}_{=r_j} + g_{G_z} J_i^{G_z} z \tag{2}$$

With:

- $r_j = \tanh(x_j)$
- τ =10 ms is the time constant of the units dynamics
- J^{GG} is the connection matrix inside of the generator network G, it contains the strength of each synapse inside G. The values of J^{GG} are much smaller than the ones of J^{Gz}, to allow a rapid learning.
- J^{Gz} is the connection matrix between the feedback provided by z and the generator network G.
- $g^{GG}$is the scaling factor of J^{GG}
- g^{Gz} is the scaling factor of the feedback loop : an increased g^{Gz} allows learning process in the network chaotic activity.
- dt = 1ms is the time step of integration

4 First-Order Reduced and Controlled Error (FORCE) Learning procedure

Force Learning does not focus on suppressing any error, instead this procedure allows output errors small enough to be fed back into the network without disrupting learning. It drastically reduces the output error right away, and keeps it small enough by making rapid and effective weight modifications. This method avoids over-fitting, which would cause stability issue, as the activity could diverge at the first non-zero error fed back into the system.
At time t, the training procedure starts by sampling the network output every Δt, the learning time step. The integration time step dt must be much smaller than the fastest timescale of the dynamics, in our case τ, and Δt must be much larger than dt. Consequently, we choose dt = 1 ms and Δt = τ = 10 ms. At each Δt, the output z(t) is compared to a target function f(t) as follow:

$$e_{-}(t) = \mathbf{w}(t - \Delta t)^{\mathbf{r}}(t) - f(t) \tag{3}$$

Then, $\mathbf{w}$ is modified as a function of the result:

$$\mathbf{w}(t) = \mathbf{w}(t - \Delta t) - e_{-}(t)P(t)\mathbf{r}(t) \tag{4}$$

with the NxN matrix P(t) beeing a running estimate of the inverse of the correlation matrix of the network rates $\mathbf{r}$ plus a regularization term αI. P(t) is initialized with :

$$P(0) = \frac{1}{\alpha}I \tag{5}$$

where α is the learning rate : small α values allow a fast learning but can make the algorithm unstable by changing too quickly the weights, too large α values can cause the fail of the learning process, by not keeping the output close to the target function for a long enough time. In our case, $\alpha = 1.0$.
P(t) is updated at the same time as the weights according to the rule :

$$P(t) = P(t - \Delta t) - \frac{P(t - \Delta t)\mathbf{r}(t)\mathbf{r}^{T}(t)P(t - \Delta t)}{1 + \mathbf{r}^{T}(t), P(t - \Delta t)\mathbf{r}(t)} \tag{6}$$

The system ultimately converges to a solution in which the weight vector is no longer changing, the training can be terminated.

5 Results

For the first simulation, we use a triangular function as the target function f. The neurons' firing rates dynamic, initially chaotic, becomes periodic during learning, and stays this way after learning, see figure 2.

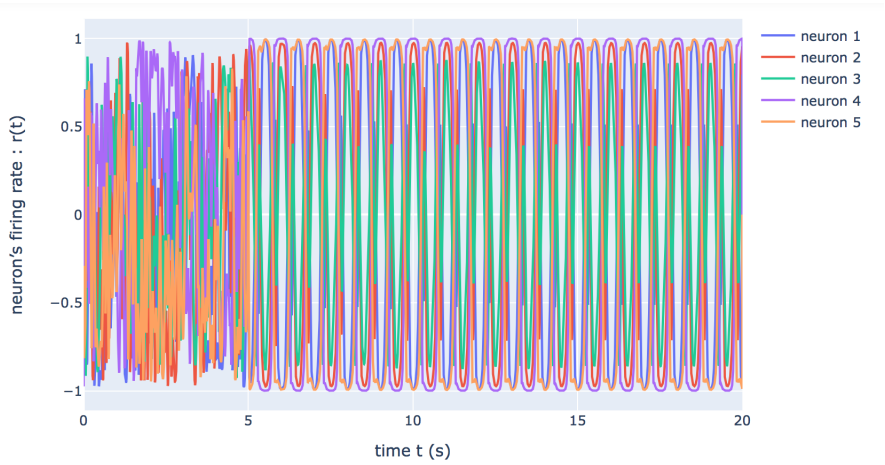

Figure 2: Evolution of the network activity — chaotic before FORCE learning — during and after the procedure

The output z (see figure 3, in green), initially chaotic during pre-learning (0-5s), matches the target function (in purple) during learning (5-15s). The rapid changes of $\mathbf{w}$ are shown by $|\frac{d\mathbf{W}}{dt}|$ (in red). $|\frac{d\mathbf{W}}{dt}|$ diminish during learning, as $\mathbf{w}$ needs less and less modification for the output to match z. After training (15-20s) the network activity is periodic and the output z matches the target without requiring any weight modification.

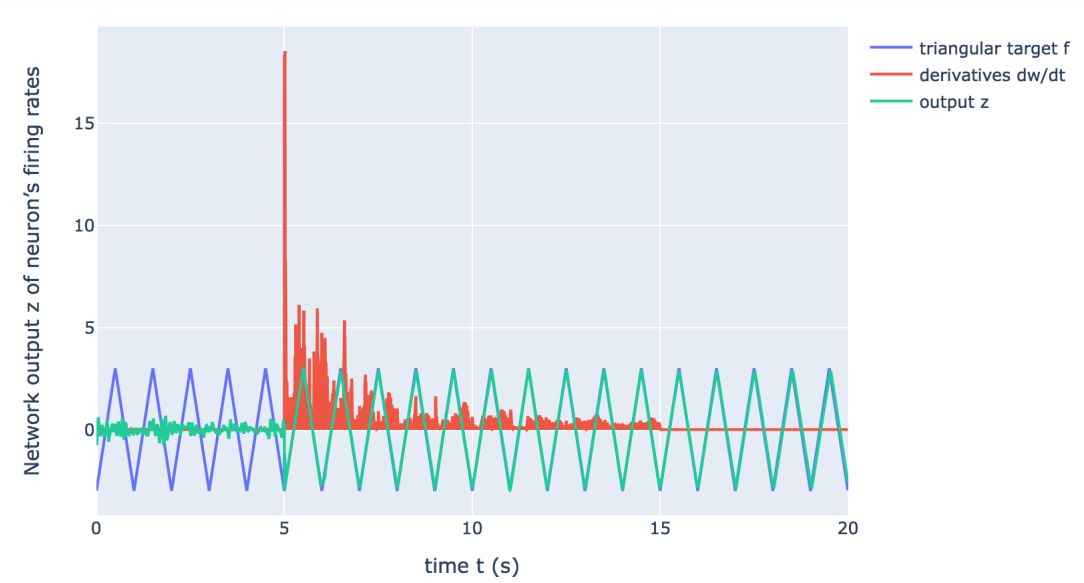

Figure 3: Evolution of the output z before, during and after FORCE learning, with a triangular target

FORCE learning procedure also allows to match the output z with a periodic but not triangular target function f, such as in figure 4, where f is composed of 4 sinusoids. The learning phase begins at t = 5s and finishes at t = 20s.

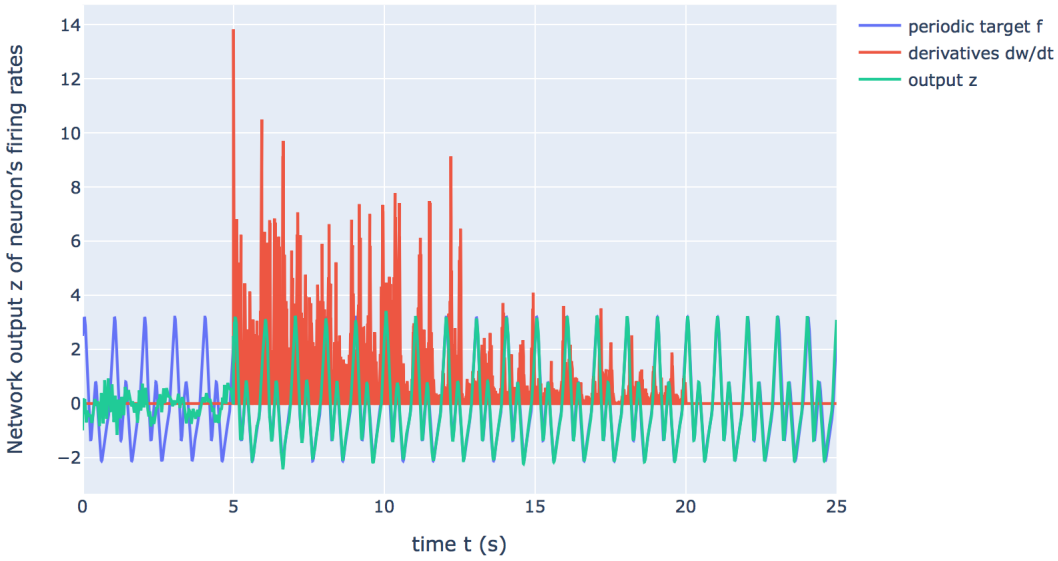

Figure 4: Evolution of the output z before, during and after FORCE learning, with a periodic target

FORCE learning also allows to match the output z with a non periodic target function, such as the Lorenz attractor. But, as the target is itself a chaotic process, a precise match between the output and the target does not last long.

FORCE learning makes it possible to match the output with a discontinuous target, see figure 5, even if the match is not as perfect as for a triangular target.

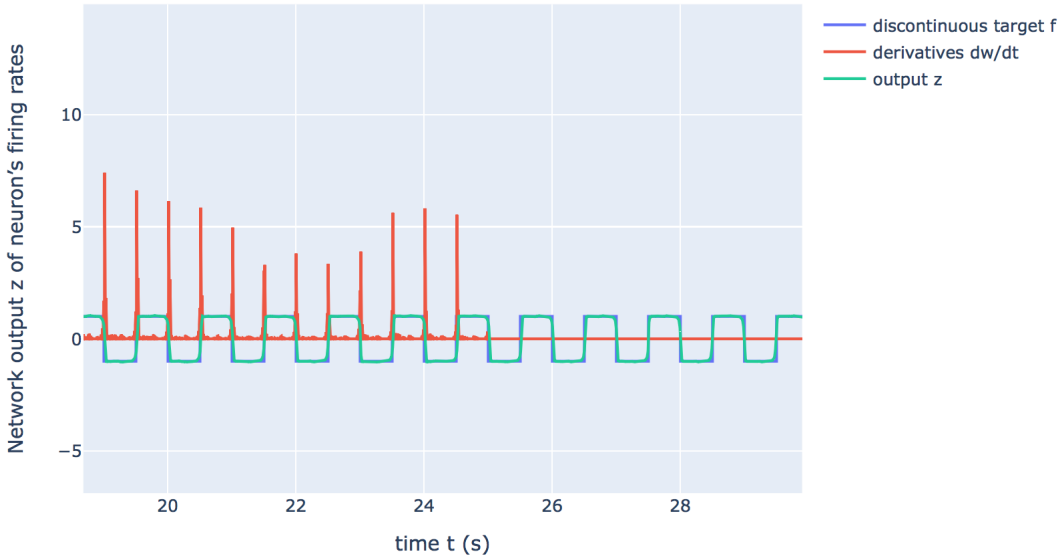

Figure 5: Evolution of the output z before, during and after FORCE learning, with a discontinuous target

Finally, FORCE learning allows to match the output z with a significantly more complicated target : a noisy target. After learning, the output z seems similar to the trace that had the target f before being blurred by noise, see figure 6.

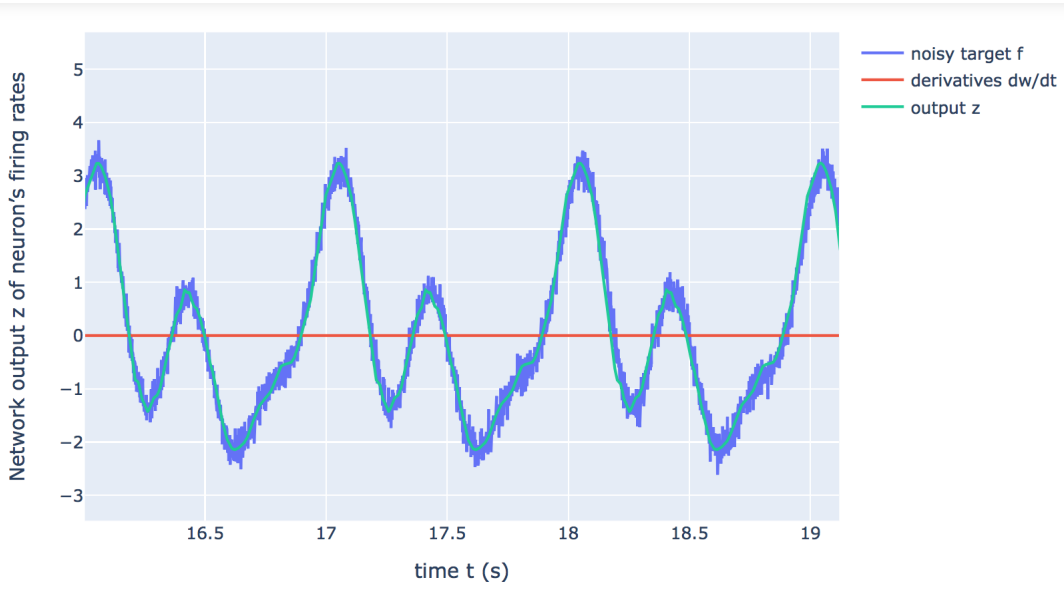

Figure 6: Evolution of the output z before, during and after FORCE learning, with a noisy target

6 Principal Component Analysis (PCA)

In this section, we try to reproduce the triangular output post - FORCE learning with a triangular function, with only the 8 principal components of the network activity. To obtain this pseudo-output, we apply principal component analysis (PCA) - diagonalization of the inverse of P, as P post-learning is considered to have converged to the inverse correlation matrix of $\mathbf{r}$. We then project the network activity on the 8 principal components (with the highest eigenvalues). We obtain the z' pseudo-output, result of the combination of the projected $\mathbf{r}$ with the post-learning (constant) projected $\mathbf{w}$. We see on figure 7 that we can reconstitute nearly perfectly the output z with z', pseudo-output formed with only the 8 principal components of the signal.

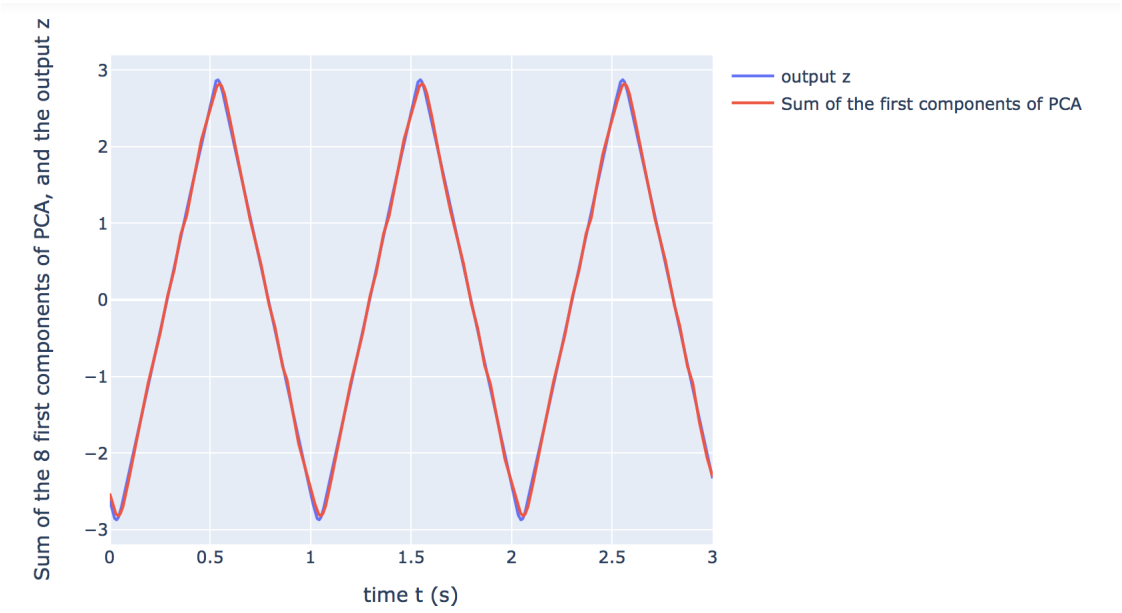

Figure 7: Plot of the output z and the pseudo-output z' from PCA
Post-learning with a triangular target

Figure 8 represents the decomposition of the pseudo-output z', and each component's contribution to the triangular z'.

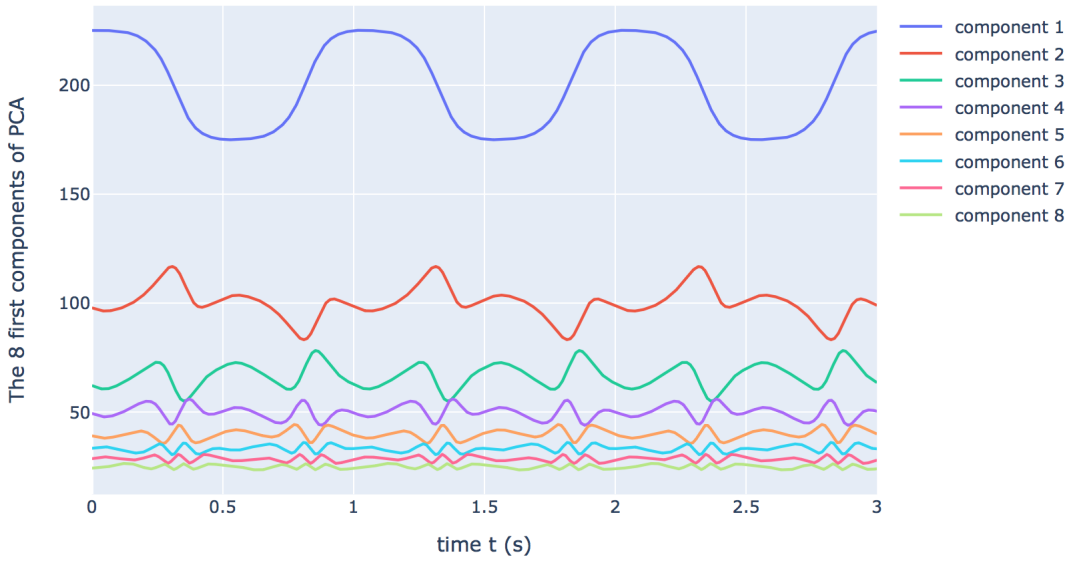

Figure 8: Plot of the 8 principal components of PCA
Post-learning with a triangular target

7 Conclusion

FORCE learning can be used as a model to study learning-induced modification of biological networks, or simply as a method to imitate this kind of networks. This procedure constitutes a real advance in the field of computational neuroscience, as it allows to model stable networks, initially chaotic, without requiring any reconfiguration of the generator network, only external synapses strength are changed, at least in the architecture we study here.
However, FORCE learning is not pretending to be a realistic model of biological networks : for example, the procedure stores and uses an error output while it is not clear how error signals are computed in biological systems, and how, once generated, they control synaptic plasticity. These problems and questions are shared by all models of error- or reward-based learning. What is more, the architecture studied in this project seems less realistic than the other ones presented by the article, where feedback comes from the generator or the feedback networks, and not directly from the readout unit. FORCE-learning is neither completely realistic nor perfectly effective, but it seems to be a real progress towards a better understanding and modelling of neuronal activity.

References

[1] Sussillo D, Abbott LF. Generating coherent patterns of activity from chaotic neural networks. Neuron. 2009;63(4):544-557. doi:10.1016/j.neuron.2009.07.018

[2] Sompolinsky, H., Crisanti, A., and Sommers, H.J. (1988). Chaos in random neural networks. Phys. Rev. Lett. 61, 259–262.