Application of a model-free based control algorithm to neural networks updating

Loic MICHEL
École Centrale de Nantes (France)
Laboratory of Digital Sciences of Nantes - UMR 6004 CNRS

July 2019

Outline

- Para-model control
- Application to neural networks
  - Basic network supervised training
  - Short-term memory network updating

Derivative-free & model-free control : Para-model control

Based on the original model-free control...

(Michel - 2011)

Para-model control definition

For any discrete moment t_k, $k \in \mathbb{N}^*$, one defines the discrete controller $\mathcal{C}_\pi : (y, y^*) \mapsto u_k$ such as :

$$u_k = \Psi_k \, . \int_0^t K_i(y^*_{k-1} - y_{k-1}) \, d\tau$$

with

$$\Psi_k = \Psi_{k-1} + K_p(k_\alpha e^{-k_\beta k} - y_{k-1})$$

where : y^* is the output reference trajectory ; K_p, K_I, k_α and k_β are real positive tuning gains

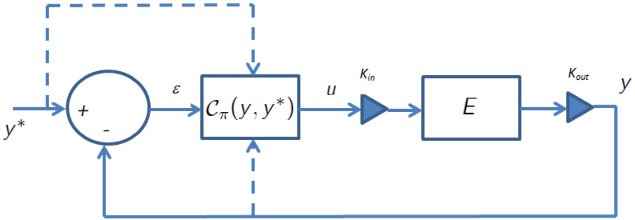
- Equivalent to a product of an integrator and a series $(\Psi_n)_{n \in \mathbb{N}}$

Para-model control

Structural properties

Given an output reference y^* and a nonlinear dynamical system E, it is *a priori* possible :

- to *control* E (track y^*) in a *robust* manner
- to *optimize* E (look for extremum)



Para-model control

Example of code

- Only few lines of basic operations :

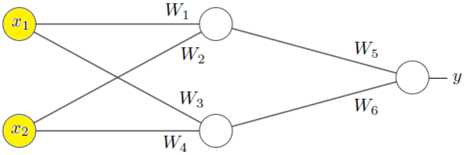
```
y_int(i) = M_alpha*exp(-M_beta*tt(i));
para_exp_err = y_int(i-1) - y(i-1);
para_stand_err(i) = y_ref(i) - y(i-1);
para_u(i) = para_u(i-1) + Kp*para_exp_err;
para_G(i) = Kint*para_stand_err(i);
para_tr(i) = para_tr(i-1) + h*(para_G(i) + para_G(i-1))/2;
para_u_final = para_u(i)*para_tr(i);
```

Outline

- Para-model control
- Application to neural networks
  - Basic network supervised training
  - Short-term memory network updating

Application to a basic neural network training

Considering a neural network defined as an "unknown" system $E : (x_1, x_2) \mapsto y$ and given a data training set $\{x_1^{train}, x_2^{train}, y^{train}\}$,

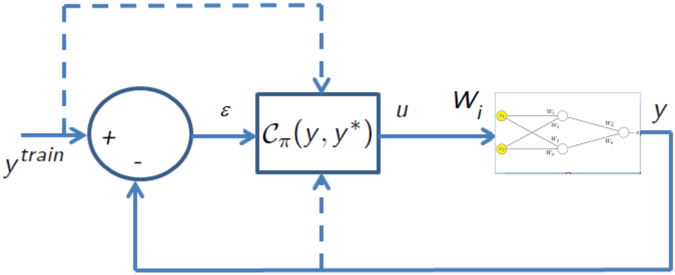


the goal is to adjust the set of the weights W_i to fit with the data training using the proposed algorithm such as :

$$\boxed{\text{for all } i, \quad W_i = \mathcal{C}_\pi(y, y^{train})}$$

Application to a basic neural network training

Proposed control scheme of the trained neural network

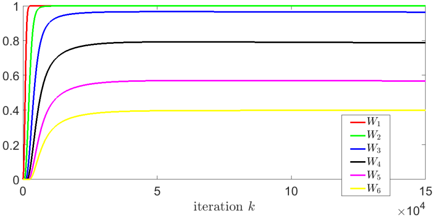


- Allows *a priori* online updates of the network according to topological network modifications / training data changes

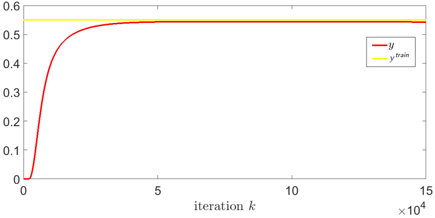
Application to a basic neural network training

Case 1

- Online tuning w.r.t. an <u>initial</u> set of training data



Evolution of the weights W_i



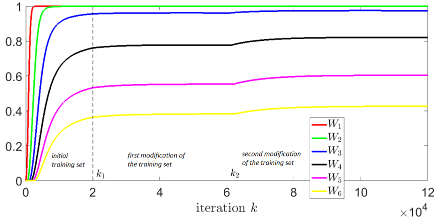
Evolution of the output y

$\Rightarrow$ *Allows a priori stabilization of the weights W_i according to the training data*

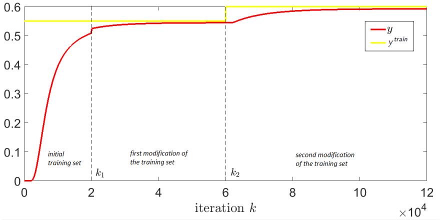
Application to a basic neural network training

Case 2

- Online tuning w.r.t. changes of the training data



Evolution of the weights W_i



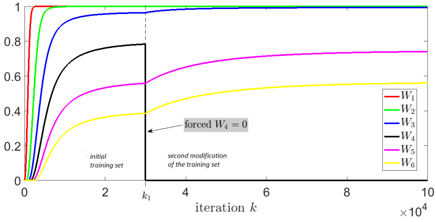
Evolution of the output y

$\Rightarrow$ *Allows a priori re-stabilization of the weights W_i according to the training data changes*

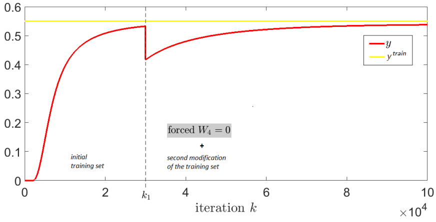
Application to a basic neural network training

Case 3

- Online tuning w.r.t. changes of the training data and the network topology



Evolution of the weights W_i



Evolution of the output y

$\Rightarrow$ *Allows a priori re-stabilization of the weights W_i according to the different changes*

Outline

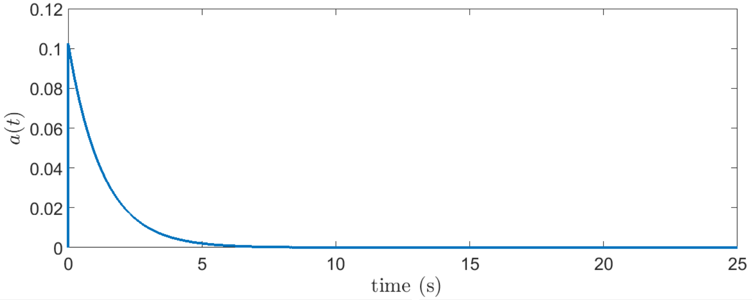
- Para-model control
- Application to neural networks
  - Basic network supervised training
  - Short-term memory network updating

Application to the self-org. short-term memory network

(Federer, Zylberberg - 2018)

Considering firstly a single neuron model :

$$\begin{cases} \tau \dfrac{d\,a(t)}{d\,t} = -a(t) + Lr(t) \\ r(t) = \mathrm{relu}(a(t)) \end{cases}$$



Evolution of the free response $a(t)$ with $a(0) = 0.1105$

Application to the self-org. short-term memory network

Controlling the single neuron to retain stimulus

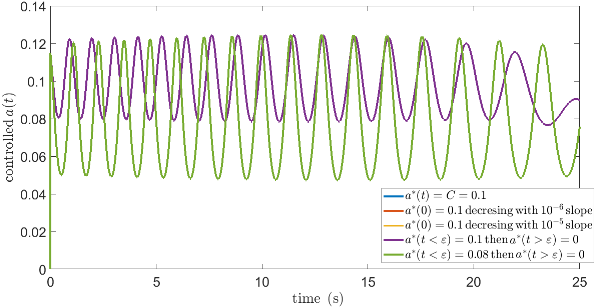
First, try controlling a single neuron using Para-model to 'simulate' a stimulus $s = a^*(0)$ retention :

$$\begin{cases} \tau \dfrac{d\, a(t)}{d\, t} = -a(t) + r(t) u(t) \\ r(t) = \mathrm{relu}(a(t)) \\ u(t) = \mathcal{C}_\pi(a(t), a^*(t)) \quad \text{under different "shapes" of } a^*(t) \end{cases}$$

- $a^*(t)$ is the output reference for which it is expected that ideally : $\boxed{\text{for all } t,\ a(t) \to a^*(0)}$

Application to the self-org. short-term memory network

Controlling the single neuron to retain stimulus - Simulation results



- Oscillations around $a^*(0)$ but *a priori* invariance of the transient response whatever the "shape" of $a^*(t > 0)$

$\Rightarrow$ *Allows a priori to retain stimulus without any external signal*

Application to the self-org. short-term memory network

Inclusion of the Para-model algorithm in a 100-neuron network

Considering a 100-neuron network updated thanks to the gradient descent plasticity rule

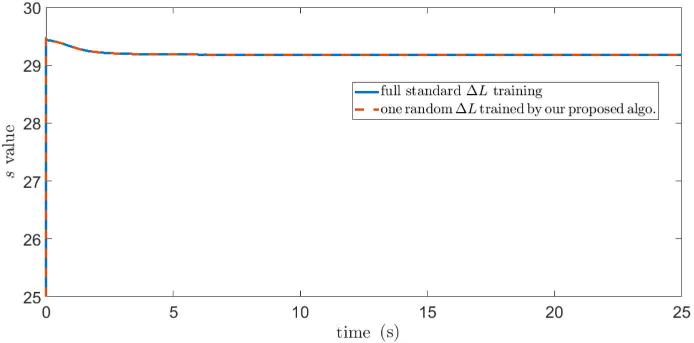
Goal : Preliminary (exploratory) results to observe the behavior of the proposed Para-model control considered as an update law

Working Assumption : A simple initial pulse s defined at $t = 0$ can *a priori* determine the behavior of the controlled neuron transient

Application to the self-org. short-term memory network

Inclusion of the Para-model algorithm in a 100-neuron network

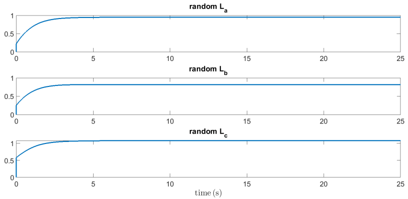
- A single neuron - randomly chosen - is updated with our proposed Para-model algorithm



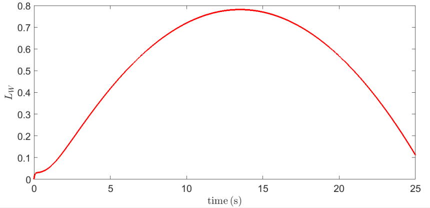
Evolution of the retained stimulus value s according to plasticity rules in comparison with our proposed algorithm (with $s(0) = 29.4$)

Application to the self-org. short-term memory network

Algorithms in interaction in a 100-neuron network



Plasticity rules ΔL for three (randomly chosen) neurons



Evolution of the controlled ΔL with Para-model algorithm

$\Rightarrow$ *First observation of an a priori non divergence of our Para-model algorithm (and stability of the s value) over a long period of time*

Perspectives

Future works include investigation of the interactions between our proposed algorithm and the plasticity update rules in multiple operating conditions as well as study of the general stability properties

References

- J.E. Gaudio *et al.*, "Connections between adaptive control and optimization in machine learning", preprint arXiv, Apr. 2019.
- M. Fliess and C. Join, "Model-free control", Int. J. Control, vol. 86, issue 12, pp. 2228-2252, Jul. 2013.
- L. Michel, "Online tuning of artificial neural networks using para-model algorithm – A preliminary study", preprint arXiv, May 2018.
- C. Federer, J. Zylberberg, "A self-organizing short-term dynamical memory network", Neural Networks, vol. 106, pp. 30-41, Oct. 2018.