Aproximación de la Dinámica Inversa de un Manipulador Robótico mediante una Red Neuronal entrenada con un Algoritmo Estocástico de Aprendizaje

On the Approximation of the Inverse Dynamics of a Robotic Manipulator by a Neural Network Trained with a Stochastic Learning Algorithm

Artículo de Investigación Científica - Fecha de Recepción: 27 de Septiembre de 2013 - Fecha de Aceptación: 17 de Noviembre de 2013

Enrique Carlos Segura
Doctor en Ciencias Matemáticas, Universidad de Buenos Aires. Buenos Aires, Argentina. esegura@dc.uba.ar

Para citar este artículo / To reference this article:
E. C. Segura, "On the Approximation of the Inverse Dynamics of a Robotic Manipulator by a Neural Network Trained with a Stochastic Learning Algorithm", *INGE CUC*, vol. 9, no. 2, pp. 39–43, 2013.

Resumen: **Se utiliza el algoritmo SAGA para aproximar la dinámica inversa de un manipulador robótico con dos juntas rotacionales. SAGA (Simulated Annealing + Gradiente + Adaptación) es una estrategia estocástica para la construcción aditiva de una red neuronal artificial de tipo perceptrón de dos capas, basada en tres elementos esenciales: a) actualización de los pesos de la red por medio de información del gradiente de la función de costo; b) aceptación o rechazo del cambio propuesto por una técnica de recocido simulado (simulated annealing) clásica; y c) crecimiento progresivo de la red neuronal, en la medida en que su estructura resulta insuficiente, usando una estrategia conservadora para agregar unidades a la capa oculta.**
Se realizan experimentos y se analiza la eficiencia en términos de la relación entre error relativo medio -en los conjuntos de entrenamiento y de testeo-, tamaño de la red y tiempos de cómputo. Se hace énfasis en la habilidad de la técnica propuesta para obtener buenas aproximaciones, minimizando la complejidad de la arquitectura de la red y, por lo tanto, la memoria computacional requerida. Además, se discute la evolución del proceso de minimización a medida que la superficie de costo se modifica.

Palabras clave: **red neuronal, manipulador robótico, perceptrón multicapa, aprendizaje estocástico, dinámica inversa.**

Abstract: **The SAGA algorithm is used to approximate the inverse dynamics of a robotic manipulator with two rotational joints. SAGA (Simulated Annealing Gradient Adaptation) is a stochastic strategy for additive construction of an artificial neural network of the two-layer perceptron type based on three essential elements: a) network weights update by means of the information from the gradient for the cost function; b) approval or rejection of the suggested change through a technique of classical simulated annealing; and c) progressive growth of the neural network as its structure reveals insufficient, using a conservative strategy for adding units to the hidden layer. Experiments are performed and efficiency is analyzed in terms of the relation between mean relative errors -in the training and testing sets-, network size, and computation time. The ability of the proposed technique to perform good approximations by minimizing the complexity of the network's architecture and, hence, the required computational memory, is emphasized. Moreover, the evolution of minimization processes as the cost surface is modified is also discussed.**

Keywords: **neural network, robotic manipulator, multilayer perceptron, stochastic learning, inverse dynamics.**

I. Introduction

One of the best known features of the classical Simulated Annealing (SA) and, at the same time, more often referenced in the literature [6], [8], [11], [13], [15] is its ability to provide, in complex optimization problems (i.e. domains with many variables and cost function with many local minima and high-valued derivatives), good "coarse" approximations, i.e., approach to an optimal or sub-optimal solution in reasonable times. In opposition, the SA frequently fails or results very slow at the "fine tune", i.e., the accurate detection of the desired optimal value. On the other hand, gradient-based minimization (optimization) algorithms (and, in the particular case of neural networks, the Error Backpropagation algorithm [12], [4]) are commonly unable to escape a local minimum but very suitable to get the exact minimum, provided that its basin of attraction has been previously located. In this way, the idea of a hybrid algorithm which combines cooperatively the good properties of both approaches arises naturally.

The other element that we intend to include is related to architecture optimality. There exist well studied [2], [3] minimal conditions that must be imposed on the structure of a feedforward neural network in order to assure its universality as a continuous functions approximator. Taking that into account, a mechanism for adding processing units was incorporated to the network learning strategy, so as to make the system able to obtain structures of minimal or quasi-minimal complexity to approximate with acceptable errors the objective function.

On the base of all these elements we proposed the SAGA algorithm (Simulated Annealing + Gradient + Adaptation). A preliminary version of such a technique has already been successfully applied to the problem of video camera calibration for autonomous robots [10], [14].

II. Toward A Simple Structure

Many researchers have looked for reducing wide classes of mathematical functions to simple combinations of simple functions. One of the first precedents is that of Kolmogorov [7] who (apparently) far from connectionist thinking, proved the (exact) representability of any continuous function in many variables by means of sums and compositions of one-variable function. This result, as well as several other some years after [1],[16], may be translated, in neural networks terms, as the demonstration that any continuous function can be represented as a three layer neural network (although what Kolmogorov intended was, in fact, to answer negatively the thirteenth Hilbert' problem). But these results are presented in a fully non-constructive way.

Since then, many researchers worked on this subject. In 1989, two mathematicians proved, almost simultaneously, that three layer perceptrons (i.e. one hidden layer) with sigmoid activation functions in the hidden units can approximate, to an arbitrarily small error, in any L^p metrics, an arbitrary continuous function [2],[3]. Here we present the result in the version given in [3].

Theorem (Funahashi)

Let Φ be a monotonic increasing and bounded function, $C \in R^N$ a compact set, $f: C \rightarrow R^L$ a continuous function. For any arbitrary $\varepsilon > 0$, there exist $K \in N$ and real numbers W_{ij}, u_j $(j=1,\ldots,K, i=1,\ldots,L)$ and w_{jk} $(j=1,\ldots,K; k=1,\ldots,N)$ such that if

$$\tilde{f}(\xi_1,\ldots,\xi_N) = \sum_{j=1}^{K} W_{ij}\,\phi\Big(\sum_{k=1}^{N} w_{jk}\xi_k - u_j\Big)$$

then

$$\left\| f - \tilde{f} \right\|_{L^\infty(C)} = \max_{\xi \in C} \left| f(\xi_1,\ldots,\xi_N) - \tilde{f}(\xi_1,\ldots,\xi_N) \right| < \varepsilon$$

This result also holds for any L^p metric ($p \geq 2$), particularly the quadratic ($p=2$). More recent results [5], [9] have precised as a necessary and sufficient condition on f that it be not a polynomial.

The resulting neural network will take the form depicted in figure 1. Activation is sigmoidal for hidden units and linear for input and output units.

From now on, we will consider (without any loss of generality) the case of a single output. Then, the parameter set $\{W_{ij}\}$ becomes simply $\{c_i\}$.

The idea of obtaining, with the cheapest possible structure, approximations of a certain utility, is inspired in this fact. The algorithm we will propose performs,

specifically, a simple strategy -¿the simplest?- for the addition of hidden units.

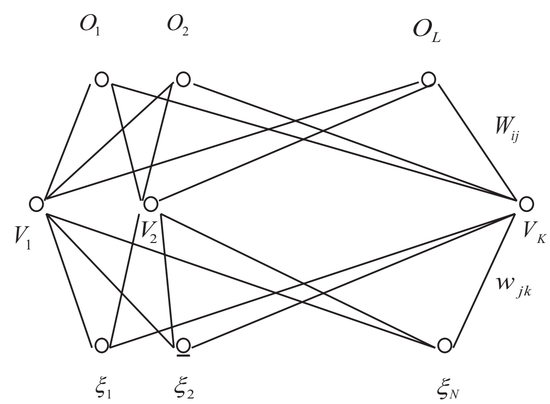

Figure 1. The basic structure of a three layer perceptron (one hidden layer)
Source: Author's own research.

III. The Algorithm

A brief sketch of the SAGA algorithm would be as follows (with /* and */ enclosing comments).

- Get a training (TRS) and a testing (TES) sets from f (with a single output variable)
- Initialize the net at random with a proper initial number K of hidden units

/* $c_i(0), w_{ij}(0), u_i(0) \approx U(0,1)$

$i = 1,\ldots,K; j = 1,\ldots,N$ */

- While t < tmax

/* or $E_{TRS} > \varepsilon$ or $E_{TES} > \varepsilon$ */

{

T ← T(t); /* tipically T(t) = A/log(1+t) */

propose_change (W,W*);

generate p ≈ U(0,1)

If exp{-Δ/T} > p; /* $\Delta E = E_x(W^*) - E_x(W)$ */

W ← W*;

Else

rejections++;

If rejections > tol and K< Kmax

add(W,K);

t++;

}

propose_change (W,W*)

{

Choose x ϵ TRS at random;

Choose r ≈ U(0,R);

/* R maximum step fixed heuristically */

$$W^* \leftarrow W - r\frac{\nabla E_x(W)}{\left|\nabla E_x(W)\right|};$$

/* E_x *(W)* term of the error

function due to x */

}

add (W,K)

{

K++; /* creates a new hidden unit */

Choose m between 1 and K-1;

/* tipically m such that $\left|C_m\right|$ is maximized */

$$u_K \leftarrow u_m;$$

$$w_{Kj} \leftarrow w_{mj} \; for\, j = 1,\ldots,N;$$

$$c_K = c_m \leftarrow \frac{c_m}{2};$$

rejections ← 0;

}

Although the algorithm looks highly parameterized, relevant parameters are three: one for the cooling schedule of the SA, the maximum step R and the tolerance value (tol) which, if overcome by the number of consecutive rejections, calls the routine which adds a hidden unit to the net (it increases in N+2 the dimension of the weight space, but preserving the error obtained up to that moment).

IV. An Application

We applied *SAGA* to the approximation of the inverse dynamics of a manipulator with two degrees of freedom. Given a control scheme as shown in fig. 2, where y(t) is the output at time t, y*(t) the desired output and u(t) the control signal, we are interested in modelling the inverse application: what should have been the control signal in order to obtain the desired output. In our case, the inverse model relates accelerations $\ddot{\theta}_1$ y $\ddot{\theta}_2$ in the joints with torques τ_1 and τ_2 to be applied (voltages on the motors), given angular positions θ_1 and θ_2 and velocities $\dot{\theta}_1$ and $\dot{\theta}_2$.

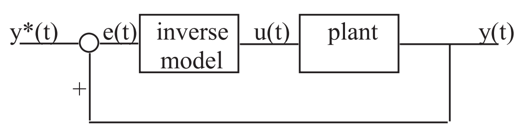

Figure 2. Control scheme with an inverse model
Source: Author's own research.

Although an analytic expression for such inverse is well known ((1) and (2)), it does not represent an exact model of the dynamics for any real manipulator, which should be modelled on the base of measurements on the plant. Anyhow, in order to evaluate our method, simulations were performed using such equations (where l_i and m_i are, respectively, the length and the mass of link i).

$$\tau_1 = m_2 l_2^2(\ddot{\theta}_1+\ddot{\theta}_2) + m_2 l_1 l_2 \cos(\theta_2)(2\ddot{\theta}_1+\ddot{\theta}_2)$$
$$+ (m_1 + m_2) l_1^2 \ddot{\theta}_1 - m_2 l_1 l_2\, sen(\theta_2)\,\dot{\theta}_2^{\,2} - 2 m_2 l_1 l_2\, sen(\theta_2)\dot{\theta}_1\dot{\theta}_2$$
$$+ m_2 l_2\, g \cos(\theta_1+\theta_2) + (m_1+m_2) l_1\, g \cos(\theta_1) \quad (1)$$

$$\tau_2 = m_2 l_1 l_2 \cos(\theta_2)\ddot{\theta}_1 + m_2 l_1 l_2\, sen(\theta_2)\dot{\theta}_1^{\,2}$$
$$+ m_2 l_2\, g \cos(\theta_1+\theta_2) + m_2 l_2^2 (\ddot{\theta}_1+\ddot{\theta}_2) \quad (2)$$

V. Experiments

SAGA was applied to the function $\tau=(\tau_1,\tau_2)$ on the domain of R^6 defined by $\theta_1,\theta_2 \in [0,\pi]$, $\dot{\theta}_1,\dot{\theta}_2 \in [-0.5,0.5]$ and $\ddot{\theta}_1,\ddot{\theta}_2 \in [-0.1,0.1]$ with l_1= 0.4 m., l_2= 0.2 m., m_1=1 kg. and m_2= 0.5 kg. The neural net has the form

$$\tilde{\tau}_i(\vec{x}) = \sum_{j=1}^{K} c_i\, \phi\left(\sum_{k=1}^{6} w_{jk} x_k - u_j\right)$$

for i = 1,2, being $\vec{x} = (\theta_1,\theta_2,\dot{\theta}_1,\dot{\theta}_2,\ddot{\theta}_1,\ddot{\theta}_2)$ and $\phi(h) = \tanh(\beta h)$

Here we show some results obtained for the case of τ_1. Figure 3 shows the evolution of the mean relative error for independent training sets. Good results can be observed for 300 (6.6 %) and for 400 examples (7.4 %). Figure 4 shows, for the same sets, the evolution of the training error as new units were added to the hidden layer. Fixing a limit of 30 units, the approximations reported above are obtained. However, it must be emphasized that not always a larger number of examples produces a larger approximation error. For example, if we had stopped the process at 20 000 iterations (figure 3) or at a maximum of 12-14 hidden units (figure 4), clearly it would have resulted more successful for the set of 400 examples. Anyhow, it is worth noting the great economy and simplicity of the resulting net structure. Figure 5 presents, for the same set of 400 examples, and a testing set of 1500 examples, the time evolution of both errors (always mean and relative). Here is clearly illustrated the overtraining phenomenon: from the 50 000 iterations on, the testing error grows up, which suggests that the useful stage of the process is over.

VI. Conclusions

A stochastic technique for the construction of two-layer perceptron type ANN's was presented. It provides good approximations of any continuous function by means of very simple architectures and a minimal or quasi-minimal number of processing units.

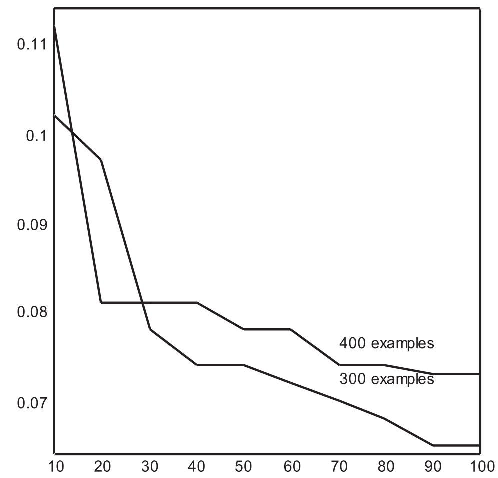

Figure 3. Training error (mean relative) vs. time (x1000 iterations)
Source: Author's own research.

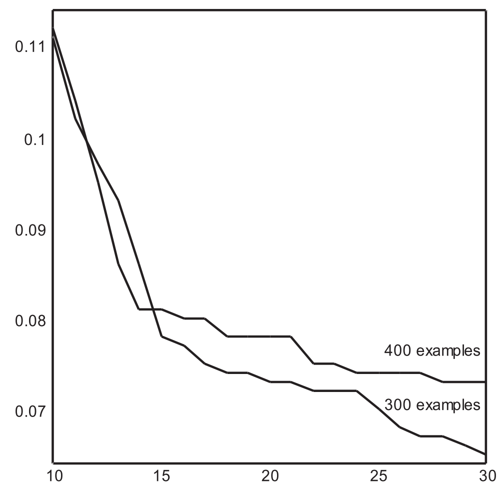

Figure 4. Training error (mean relative) vs. nr. of hidden units
Source: Author's own research.

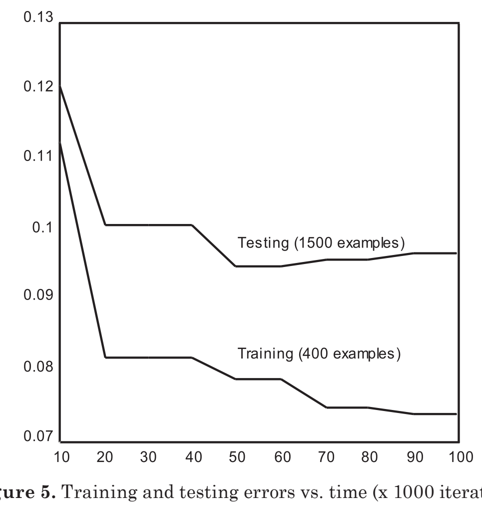

Figure 5. Training and testing errors vs. time (x 1000 iterations)
Source: Author's own research.

EDUCOSTA

The proposal was tried in the case of the inverse dynamics of a robotic manipulator with two degrees of freedom (rotational joints), which implied the approximate modelization of a function with six input and two output variables. Although the obtained results can be considered very good, it would be interesting to extend the experiments to more degrees of freedom, as well as to testing and training sets with higher cardinality. As for the domain over which we worked, it may be considered close enough to a real case.

References

[1] V. I. Arnold, "On Functions of three Variables", *Dokl. Akad. Nauk*, no.114, pp. 679-681, 1957.

[2] G. Cybenko, "Approximation by superpositions of a sigmoidal function", *Math. Control, Signals and Systems*, vol.2, no.4, pp. 303-314, 1989.

[3] K. Funahashi, "On the approximate realization of continuous mappings by neural networks", *Neural Networks*, vol.2, no.3, pp. 183-92, 1989.

[4] S. Haykin, *Neural Networks and Learning Machines.* Upper Saddle River, Pearson-Prentice Hall, 2009.

[5] Y. Ito, "Extension of Approximation Capability of Three Layered Neural Networks to Derivatives", *Proc. IEEE Int. Conf. Neural Networks*, San Francisco, 1993, pp. 377-381.

[6] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by Simulated Annealing", *Science*, vol. 220, pp. 671-680, 1983.

[7] A. N. Kolmogorov, On the Representation of Functions of Many Variables by Superposition of Continuous Functions of one Variable and Addition (1957), *Am. Math. Soc. Tr.,* vol.28, pp. 55-59, 1963.

[8] P. J. Van Laarhoven and E. H. Aarts, Simulated Annealing: Theory and Applications. Dordrech: Kluwer, 2010.

[9] M. Leshno, V. Y. Lin, A. Pinkus and S. Schocken, "Multilayer Feedforward Networks with a Nonpolynomial Activation Function Can Approximate Any Function", *Neural Networks*, vol.6, no 6, pp. 861-867, 1993.

[10] A. B. Martínez, R. M. Planas, and E. C. Segura, "Disposición anular de cámaras sobre un robot móvil", en *Actas XVII Jornadas de Automática Santander96*, Santander, 1996.

[11] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. H. Teller, and E. Teller, "Equation of State Calculations by Fast Computing Machines", *J. Chem. Phys,* vol. 21, no 6, pp. 1087-91, 1953.

[12] D. E. Rumelhart, G. E Hinton, and R. J. Williams, "Learning representations by back-propagating errors", *Nature* no.323, pp. 533-536, 1986.

[13] P. Salamon, P. Paolo Sibani, and R. Frost, *Facts, Conjectures and Improvements for Simulated Annealing.* SIAM Monographs on Mathematical Modeling and Computation, 2002.

[14] E. C. Segura, *A non parametric method for video camera calibration using a neural network*, *Int. Symp.* Multi-Technology Information Processing, Hsinchu, Taiwan, 1996.

[15] E. C. Segura, *Optimisation with Simulated Annealing through Regularisation of the Target Function*, Proc. XII Congreso Arg. de Ciencias de la Computación, Potrero de los Funes, 2006.

[16] D. A. Sprecher, "On the Structure of Continuous Functions of Several Variables", *Tr. Am. Math. Soc.,* vol.115, pp. 340-355, 1963.