4th Inverse Problems, Design and Optimization Symposium (IPDO-2013)
Albi, France, June 26-28, 2013

DATA ASSIMILATION BY NEURAL NETWORK ON HARDWARE DEVICE

Vitor C.S. Gomes[a], Sabrina B.M. Sambatti[a], Helaine C.M. Furtado[a], Eduardo F.P. Luz[a], Andrea S. Charão[b], Haroldo F. de Campos Velho[a]
[a] *National Institute for Space Research (INPE), São José dos Campos (SP), Brazil,*
[vconrado, sabrinabm]@gmail.com , [helaine.furtado, eduardo.luz, haroldo]@lac.inpe.br ;
[b] *Federal University of Santa Maria (UFSM), Santa Maria (RS), Brazil, andrea@inf.ufsm.br ;*

Abstract

Data assimilation combines observation and data from a mathematical model to compute the best initial condition. Inverse problems can be classified according to the nature of the estimated property type: source term, system property, boundary condition, and initial condition. Extended Kalman filter (EKF) and four dimensional variational method (4D-Var) are two procedures employed to perform DA. Artificial neural networks are applied here for reducing the computational complexity for a given DA scheme. The supervised multi-layer perceptron is used to emulate the Kalman filter. The designed neural network was implemented on a FPGA (Field-programmable Gate Array), employed as a co-processor. The linear 1D wave equation is the dynamical system used for testing the framework. A good performance was obtained with neural network emulating the Kalman filter, with an average error 5.18E-05 (software and FPGA comparison) on the FPGA implementation.

Introduction

Inverse problems can be classified according to the nature of the method (explicit or implicit), nature of the unknown set (function or parameter), or the nature of the estimated property type: source term, system property, boundary condition, and initial condition. Data assimilation combines observation and data from a mathematical model to compute the best initial condition. The estimation of initial condition is an essential issue for operational prediction centers for weather forecast [1, 2], ocean circulation [3], space weather [4], hydrology [5], and environmental prediction [6].

Modern techniques as ensemble Kalman filter (EnKF) [7, 8], particle filter (PF) [9, 10] – two Bayesian techniques, and 4D variational methods [1, 2, 11] represent high level standards for data assimilation process. However, these schemes are very expensive, from computational point of view.

Nowosad et al. [13] employed a multi-layer perceptron neural network (MLP-NN) by data assimilation emulating the extended Kalman filter. In the approach suggested by Härter and Campos Velho [14], a local NN was employed, instead of the use the strategy used before [13]. The application of neural networks for data assimilation was also tested to emulate particle filter [14], and variational method [15]. Recently, the neural network was applied to a 3D general circulation atmospheric model (SPEEDY: Simplified Parameterizations PrimitivE-Equation Dynamics), where the MLP-NN was able to produce a similar analysis than the local ensemble transform Kalman filter (LETKF) [16], but the neural network was 75 times faster than LETKF [17].

However, a difficult task is to find out the best configuration of the ANN for a given application. Some studies are addressed to develop automatic schemes for configuring an ANN. In our approach, the configuration of the multi-layer perceptron (MLP) is formulated as an optimization problem. A new meta-heuristic, the multi-particle collision algorithm (MPCA) [19], is used to identify the best configuration [20]. The MPCA is inspired to emulate the behavior of neutrons transport in a nuclear reactor, where absorption and scattering are considered.

Data assimilation is performed by MLP-NN by using the FPGA (Field-programmable Gate Array. In the 1980s, many studies were carried out to design hardware neurocomputers. However, the development was based on application-specific integrated circuit (ASIC). At that time, there is no demand for large-scale production. On the other hand, FPGAs worked with relatively small memory, and they were not fast enough for real applications of ANN [21]. The technology changed, and the current performance for FPGA allows a configuration to be neurocomputers for facing applications with intensive computation.

The experiment for data assimilation by a FPGA-neurocomputer is carried out with linear 1D wave equation [3, 22]. A comparison between data assimilation using neural network by software and hardware is shown, where small differences are reported.

2. Forward model: Linear 1D wave equation

The dynamical system used in our tests is a linear, first-order partial differential equation, called here linear 1D wave equation:

$$\frac{\partial \eta}{\partial t} + c\frac{\partial \eta}{\partial t} = F(x,t) \qquad (1)$$

where $\eta(x,t)$ is the unknown variable to be estimated by data assimilation, c is the constant phase speed, $F(x,t)$ is the external forcing, t is time, and x is space. Periodic boundary conditions are assumed. The initial condition is given by the following equation:

$$\eta(x,0) = \eta_0 \frac{1}{\cosh^2[(x-\nu)/\Delta]} \qquad 0 \le x \le L_x \qquad (2)$$

with $\eta_0 = -60$, $\nu = c + \alpha\eta_0/3$, $\alpha = -1.62\times10^{-2}$, $c = 2.42$, and $\Delta = 1340$. The equation (1) is integrated using finite difference (for space), and Crank-Nicholson (for time).

3. Data assimilation sche**mes**

3.1 Kalman filter

The Kalman filter is a well established statistical estimation process under a stochastic Gaussian process. The algorithm for the cycle of data assimilation when the observation is available can be summarized as following:

1. Forecast model for state vector: $\eta^f_{v,n+1} = M_{n+1}\eta^a_{v,n}$, with: $\eta^f_{v,n} = [\eta^f_1(t_n) \ \dots \ \eta^f_{N_x}(t_n)]^T$.
2. Up-date the covariance matrix: $P^f_{n+1} = M_{n+1}P^a_{n+1}M^T_{n+1} + W^{\mathrm{Mod}}_n$
3. Compute the Kalman gain: $K_{n+1} = P^f_{n+1}H^T_{n+1}[W^{\mathrm{Obs}}_n + H_{n+1}P^f_{n+1}H^T_{n+1}]^{-1}$
4. Compute the analysis (data assimilation): $\eta^a_v = \eta^f_v + K_{n+1}[\eta^{\mathrm{Obs}}_v - H_{n+1}\eta^f_{n+1}]$
5. Up-date the analyis covariance: $P^a_{n+1} = [I - K_{n+1}H^T_{n+1}]P^f_{n+1}$

The state value $\eta(x,t)$ is discretized: $\eta(x_i,t_n)$, and the matrix M_n represents the state transition matrix from the state η_n up to η_{n+1} for the discrete dynamical system. Matrices P, H, W^{Obs}, W^{Mod} are the state covariance matrix, observation system matrix, and error covariance matrices for observations and modeling, respectively. The superscript f and a are the predicted values (forecasting, or also background), the *analysis*. Subscripts ν and n identifies the grid point (x_i) and discrete time (t_n), respectively. Finally, the matrix K is the Kalman gain. Several studies are dedicated to evaluate the matrix W^{Mod}. The ensemble Kalman filter [8, 16] is a procedure to identify this matrix. For non-linear and non-Gaussian stochastic process, the particle filter is an alternative estimation procedure [9, 14].

3.2 Neural network: Multi-layer perceptron

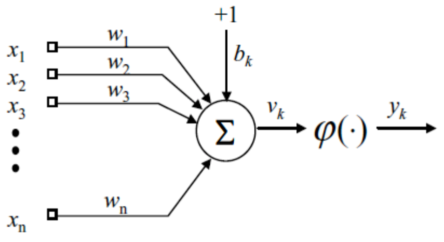
Artificial Neural Network (ANN) are distributed parallel systems, they are composed by simple processing units (node) that calculate certain mathematical functions (typically nonlinear), emulating the human brain function to accomplish specific tasks. These units are arranged in one or more interconnected layers by a large number of connections. These connections are associated with weights in various models, which store the knowledge represented in the model and serve to balance the input received by each network neuron [23].

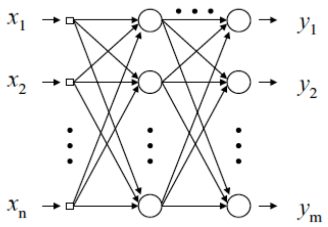
Multilayer Perceptron (MLP) is one of the most commonly used topologies, they have at least one intermediate layer or hidden, MLP have been applied successfully to solve several and difficult problems by training them with a supervised manner with a highly popular backpropagation algorithm.

The backpropagation learning error consists to compute the weight connections (synaptic weights) in order to produce the best fitting with the target values. The synaptic weights are all adjusted in accordance with an error correction rule [23].

MLP with backpropagation learning algorithm are feedforward networks composed of an input layer, an output layer, and a number of hidden layers, whose aim is to extract high order statistics from the input data [23]. Figure 1a is an outline of an artificial neuron, and Figure 1b depicts a multilayer neural network with one hidden layer. Neural networks will solve nonlinear problems, if nonlinear activation functions are used for the hidden and/or the output layers. From several activation functions, the sigmoid are commonly used (a is a free parameter):

$$\varphi(v) = \frac{1-\exp(-av)}{1+\exp(-av)} \quad . \tag{3}$$





(a) (b)

Figure 1: (a) single neuron, (b) multi-layer neural network.

3.2.1 MPCA for ANN self-configuring

For finding the best configuration for an ANN for a given application, many parameters must be evaluated: number of hidden layers, number of neurons for each layer, type of activation function, learning ratio, and momentum rate. These parameters are identified by minimizing a cost function. The optimization problem is solved by the MPCA (Multi-Particle Collision Algorithm) [19] meta-heuristic.

The standard PCA starts with a selection of an initial solution (Old-Config), and it is modified by a stochastic perturbation (Perturbation {.}), leading to the construction of a new solution (New-Config). The new solution is compared (function Fitness {.}), and the new solution can or cannot be accepted. If the new solution is not accepted, the scheme of scattering (Scaterring {.}) is used. The exploration around closer positions is guaranteed by using the functions Perturbation {.} and Small-Perturbation {.}. If the new solution is better than the previous one, this new solution absorbed. If a worse solution is found, the particle can be sent to a different location of the search space, it enables the algorithm of escaping a local minimum [19].

The implementation of the MPCA algorithm is similar to PCA, but it uses a set with N particles, where a mechanism to share the particle information is applied. A blackboard strategy is adopted, where the best-fitness information is shared among all particles in the process. This process was implemented in Message Passing Interface (MPI), looking for application into a distributed memory machine. The pseudo-code for the MPCA is presented in Figure 2.

The best configuration for the ANN is obtained by minimizing the following objective function [18]:

$$J(X) = penality \times \frac{\rho_1 \varepsilon_1 + \rho_2 \varepsilon_2}{\rho_1 + \rho_2} \tag{4}$$

where ρ_1 and ρ_2 are learning and generalization weights, with ε_1 and ε_2 errors for each process cited. The vector X is the unknown parameter set: number of hidden layers and neurons per hidden-layer, type of activation function, learning and momentum ratios. The penality function is given by [18]:

$$penality = C_1 (e^{\#neurons})^2 \times (C_2 (\#epochs)) + 1 \,. \qquad (5)$$

Therefore, simpler ANN are chosen, i.e., ANN with few neurons and fast convergence. The values for constants C_1 and C_2 are the same used in reference [18].

```
Generate an initial solution: Old-Config
Best-Fitness = Fitness{Old-Config}
Update Blackboard
For n = 0 to # of particles
    For n = 0 to # iterations
    Update Blackboard
    Perturbation{.}
        If Fitness{New-Config} > Fitness{Old-Config}
            If Fitness{New-Config} > Best-Fitness
                Best-Fitness = Fitness{New-Config}
            End If
            Old-Config = New-Config
            Exploration{.}
        Else
            Scattering{.}
        End If
    End For
End For
```

Figure 2. MPCA pseudo-code.

4. Data assimilation by hardware device: FPGA as a neurocomputer

The Cray XD1 is a hybrid computer system combining CPU and FPGA in the same computer node. Our implementation was in a machine with 6 interconnected nodes (blades). Each blade has two AMD Opteron 2.4 GHz processors and one FPGA Xilinx Virtex II. Data transfer between the CPU and FPGA is performed by the API (application programming interface) RapidArray transport core. This API allows the CPU sending and receiving data to the FPGA, and the FPGA can read and write on the shared memory regions with the CPU.

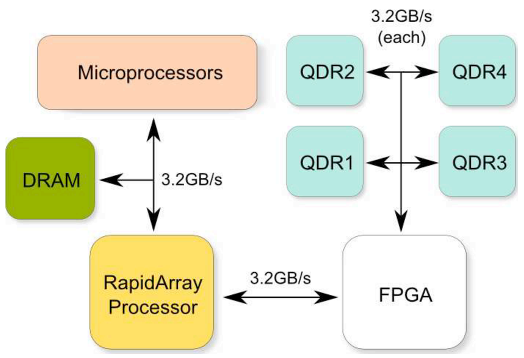


Figure 3. Cray XD1 blade.

In the Cray XD1, the FPGA can access to different levels of memory. The Dynamic random-access memory (DRAM) has up to 8GB and variable reading latency. A board with QDR II SRAM (QDR: Quad Data Rate; SRAM: Static Random-Access Memory) – it is a type of static RAM computer memory with capacity to transfer up to four words of data in each clock cycle – is available. Four independent

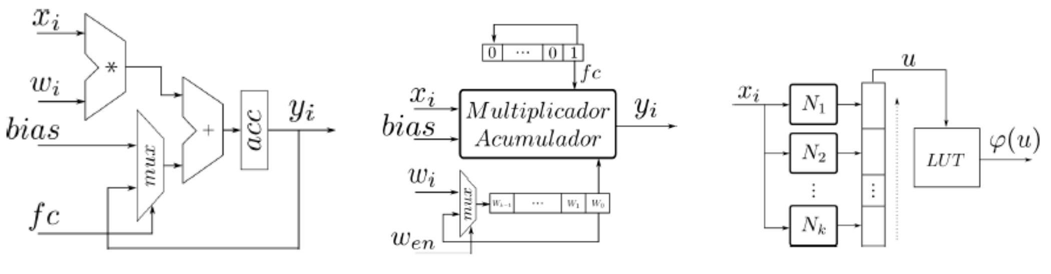
QDR II SRAM with 4MB can be directly accessed by FPGA, with latency of 8 cycles. There is an internal FPGA memory, with smaller capacity, and it can be accessed every cycle. Figure 3 shows the architecture for the Cray XD1 blade.

4.1 MLP-NN for the Cray XD1

The implementation of the MLP-NN on FPGA, designed for the data assimilation, has different modules. Each module is embedded into other modules – as computation components.

The MAC (Multiplier and ACcumulator) unit – see Fig. 4a – storages the result of the product between inputs and synaptic weights, adding the bias. Each input x_i is multiplied by a weight w_i and added to the register *acc* or to the *bias*. For selecting the operation to be done, the signal *fc* is provided. The next module is the artificial neuron, and it uses a MAC and control structures – see the diagram in Fig. 4b. In this module, the *fc* signal is generated by a shifting register with only one bit. For the weights management, interconnected registers are used on the circular queue. The weights are shifted at each x_i input. The last computational module is a combination of neurons, with the inputs are connected by a unique bus. The output of each neuron is connected to a position of a shifting register with parallel loading – see Fig. 4c. The neurons can receive data, and the results (outputs) are flowing to the Lookup Table (LUT) unit: this is operation to simulate the activation function.

The MLP-NN design is complete with serial concatenating of layers forming a neural network. The input of each layer is directly concatenated to the output of the previous layer. Considering a layer as a computation module, a pipeline of a operation sequence is performed. The computation for each layer can be independently executed, allowing multiples data set can be computed with a sequential delay for each computation layer.



(a) (b) (c)

Figure 4: ANN on FPGA: (a) MAC unit: Multiplier and ACumulator, (b) neuron (using MAC), (c) ANN implemented: the pipeline.

5. Results

The linear 1D wave equation was discretized using N_x = 128 grid points for the space mesh ($L_x = N_x\Delta x = 128\times150 = 19200$ m), with time integration up to 200 time-steps ($t_{\max} = N_t\Delta t = 3000\times10 = 30000$ seconds). For the Kalman filter implementation, the following values were assumed:

$$W_n^{\text{Mod}} = 0.1I \ , \quad W_n^{\text{Obs}} = 0.5I \ , \quad H_n = I \ ,$$

with I being the identity matrix. The observations were generated synthetically, and the assimilation cycle is carried out every 10 steps. The synthetic observations were computed from the time integration of Eq. (1), and adding to the result a Gaussian random noise with 5% for level of noise. In general, data assimilation is a two steps process:

Forecast step: $\eta_{n+1}^f = M[\eta_n^f]$ (6)

Analysis step: $\eta_{n+1}^a = \eta_{n+1}^f + d_{n+1}$ (7)

where d_{n+1} is the innovation. For the Kalman filter, the innovation is computed using the Kalman gain. In this paper, the analysis η^a is calculated by

$$\text{Analysis step:} \quad \eta_{n+1}^a = F_{NN}(\eta_{n+1}^{\text{Obs}}, \eta_{n+1}^f) \qquad (8)$$

being $F_{NN}(.)$ the output from the multilayer perceptron neural network.

5.2 Data assimilation with neurcomputer (FPGA)

The ANN implementation on FPGA is evaluated with the MLP-NN described by [22], with two inputs (forecasting and observation), three neurons in the hidden layer, and one neuron for the output layer. The VHDL (VHSIC Hardware Description Language) description for the mentioned NN was compiled. The activation function is not fully implemented: its values are computed and storage in a table (look-up-table procedure).

The results for assimilation process with FPGA are shown in Figure 6 (software, FPGA, and square error). The mean square error was 5.18×10^{-5}, with variance 1.79×10^{-8}. The largest square error was 1.30×10^{-3}.

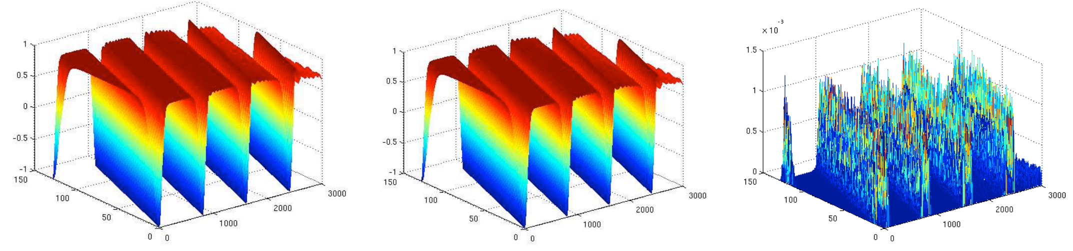


Figure 6: Data assimilation using MLP-NN: (a) Software, (b) FPGA, (c) square error.

5.1 MLP-NN design for data assimilation using MPCA

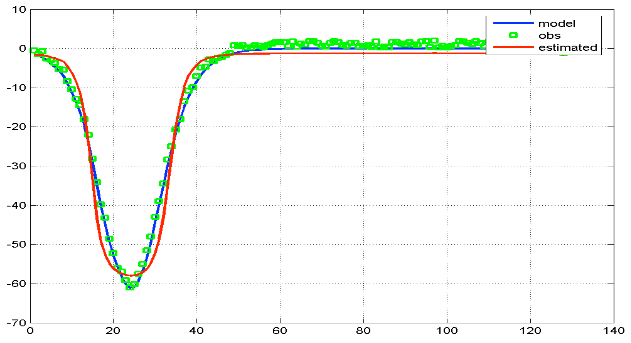
The MPCA was executed with: 6 particles, one particle per processor, total 6 processors. The maximum number of evaluations of the objective function was adopted as stopping criterion. The results are presented considering the average of 4 experiments. The numerical experiments were performed with synthetic observational data. The MPCA was applied to optimize the parameters of ANN following: number of neurons in the hidden layer, activation function, learning and momentum ratios.

The numerical experiment used different initial guess for the ANN connection weights and bias. In the first experiment, the first guess was the fixed value of 0.5 for all weights and bias. In the second experiment, random values were used to weights and bias. The best architectures for both experiments found with MPCA are shown in Table 1, and the parameters selected by an expert [22] are also shown.

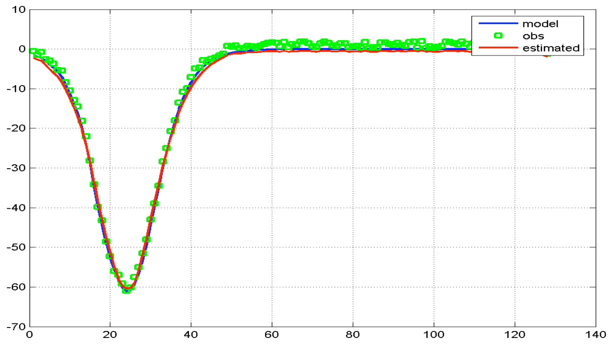
Table 3: ANN topologies

Parameters	MPCA-ANN Exp-1	MPCA-ANN Exp-2	Empirical-ANN
Hidden layers	1	1	1
Hidden layers neurons	2	23	1
Learning ratio	0.53	0.4	0.9
Momentum rate	0.2	0.6	0.0
Activation function	tanh	tanh	tanh
Square error	0.5264	**0.1583**	0.4212

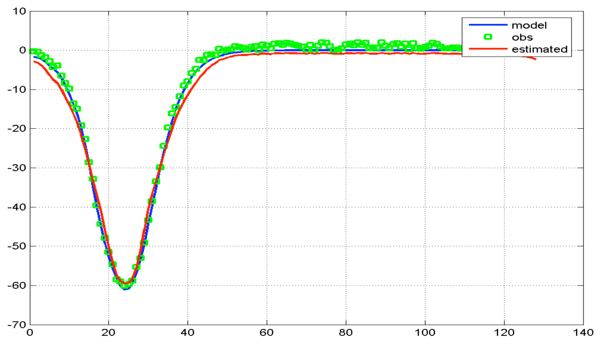
Figure 5a shows the results for the wave profile at time-step 3000. The blue curve correspond to the true state, the red curve is the result of the analysis (ANN emulating the Kalman filter), and the observation are represented by green square. Data assimilation with empirical ANN (Fig. 5c) presented a better performance than ANN-1 (configured by MPCA). The assimilation with the ANN2 (Fig. 5b) has the best performance of all ANNs, the analysis reproduced with great fidelity the real dynamics.



(a)



(b)



(c)

Figure 5: (a) MPCA-ANN1: fixed weight-bias; (b) MPCA-ANN2: random weight-bias; (c) ANN [22]

6. Conclusions

The self-configuring strategy of the MLP-NN using MPCA meta-heuristic, applied to emulate the Kalman filter for data assimilation, was effective. Actually, one of the found topologies produced better results than a topology selected by an expert. This scheme allows the use of ANN for a larger community. The implementation on FPGA works well, where the fixed point arithmetic was adopted for avoiding memory constraints. In a future work, we want to repeat the experiment, but employing the floating-point arithmetic.

REFERENCES

1. R. Daley, *Atmospheric Data Analysis*. Cambridge University Press, 1993.

2. E. Kalnay, *Atmospheric modeling, data assimilation and predictability*, Cambridge Univ. Press. Cambridge, 2003.

3. A. Bennett, *Inverse Modeling of The Ocean and Atmosfere*, Cambridge University Press, 2002.

4. J. S., Guo, S. P. Shang, J. K. Shi, M. L. Zhang, X. G. Luo, H. Zheng, H., Optimal assimilation for ionospheric weather—theoretical aspects. *Space Science Reviews*, **107**, 229 (2003).

5. W. T. Crow, D. Ryu, A new data assimilation approach for improving hydrologic prediction using remotely-sensed soil moisture retrievals. *Hydrol. Earth Syst. Sci.*, **5**, 2005 (2008).

6. J. Frydendall1, J. Brandt, J. H. Christensen, Implementation and testing of a simple data assimilation algorithm in the regional air pollution forecast model, DEOM. *Atmos. Chem. Phys.*, **9**, 5475 (2009).

7. P. L. Houtekamer, H. L. Mitchell, A sequential ensemble Kalman filter for atmospheric data assimilation. *Mon. Wea. Rev.*, **129**, 123 (2001).

8. A. Evensen, *Data assimilation: the emsemble Kalman filter*. Springer, 2006.

9. N. Gordon, D. Salmond, A. Smith, Novel approach to nonlinear/non-gaussian bayesian state estimation, *IEE Proceedings*, **140**(2) 107 (1993).

10. P. J. van Leewen, Nonlinear data assimilation in geosciences:an extremely efficient particle filter. *Q. J. Royal Meteorol. Soc.*, **136**(B), 1991 (2010).

11. P. Courtier, Dual Formulation of four-dimensional variational assimilation. *Q. J. Royal Meteor. Soc.*, **123**(B), 2449 (1997).

12. A. N. Nowosad and H. F. Campos Velho, Data assimilation in chaotic dynamics using neural networks. *Proceedings of the International Conference on Nonlinear Dynamics, Chaos, Control and Their Applications in Engineering*, Campos do Jordão (SP), Brazil, **6**, 212 (2000).

13. F. P. Härter, H.F. Campos Velho, New approach to applying neural network in nonlinear dynamic model. *Appl. Math. Model.*, **32**, 2621 (2008).

14. H. C. M. Furtado, H. F. Campos Velho, E. E. N. Macau, Data assimilation: particle filter and artificial neural networks. *Journal of Physics. Conference Series (Online)*, **135**, 012073, 2008.

15. H. C. M. Furtado, H. F. Campos Velho, E. E. N. Macau, Neural networks for emulation variational method for data assimilation in nonlinear dynamics. *Journal of Physics. Conference Series (Online)*, **285**, 012036, 2011.

16. T. Miyoshi, S.Yamane, Local ensemble transform Kalman filtering with an AGCM at a T159/L48 resolution, *Mon. Weather Rev.*, **135**, 3841 (2007).

17. R. S. C. Cintra, H. F. Campos Velho, Global data assimilation using artificial neural networks in SEEDY model. *1st International Symposium Uncertainty Quantification and Stochastic Modeling* (Uncertainties 2012), Maresias (SP), Brazil, February 26 up to March 02, 648-654, 2012.

18. A. R. Carvalho, F. M. Ramos, A. A. Chaves, Metaheuristics for the feedforward artificial neural network (ANN) architecture optimization problem, *Neural Comp. Applic.*, **20 (8)**, 1273 (2010).

19. E. F. P. Luz, J. C. Becceneri, H. F. Campos Velho, A new multi-particle collision algorithm for optimization in a high performance environment, *J. Comp. Interdisciplinary Sci.*, **1**(3), 10 (2008).

20. S. B. M. Sambatti, J. A. Anochi, E. F. P. Luz, A. R. Carvalho, E. H. Shiguemori, H. F. Campos Velho, Automatic configuration for neural network applied to atmospheric temperature profile identification. *3rd International Conference on International Conference on Engineering Optimization* (EngOpt), Rio de Janeiro (RJ), Brazil, 1–5 July, 1-9 (2012).

21. A. R. Omondi, J. C. Rajapakse (Editors), *FPGA Implementations of Neural Networks*, Springer, 2006.

22. H. C. M. Furtado, H. F. Campos Velho and E. E. N. Macau, Data assimilation with artificial neural networks on differential equations. *Brazilian Conference on Dynamics, Control, and Application*s (DINCON), 28-August up to 01-September, 595 (2011) – in Portuguese.

23. S. Haykin, *Neural Networks - A Comprehensive Foundation*, 2nd Edition, Prentice-Hall, 2001.