Proceeding Series of the Brazilian Society of Computational and Applied Mathematics
Preprint

Data Assimilation on CoLab Environment: Methods and Application on Two Dynamical Systems

Roberto P. Souto[1], Maria Eugenia S. Welter[2], Carla Osthoff F. de Barros[3]
LNCC, Petrópolis, RJ, Brazil
Helaine C. M. Furtado[4], Julio T. Silva[5]
UFOPA, Santarém, PA, Brazil
Juliana A. Anochi[6], Marcelo P. Ramos[7], Haroldo F. de Campos Velho[8],
INPE, São José dos Campos, SP, Brazil
Luiz A. V. Dias[9]
ITA, São José dos Campos, SP, Brazil
Sabrina B. S. Sambatti[10]
Independent researcher, São José dos Campos, SP, Brazil

Abstract. Data assimilation (DA) is an essential process to identify the best initial conditional by combining data from an observation system with a previous prediction from a numerical simulation of a given dynamical system. This paper describes the effort to develop a framework for testing different methods applied to two dynamical systems. The framework was implemented using the Google CoLab platform, and Octave free mathematical software. The dynamic systems used for testing is the Lorenz system under the chaotic regime, and 2D shallow water — for ocean circulation modeling.

Key-words. Data assimilation methods, Lorenz system, Shallow water 2D, Octave package, Google Colab platform.

1 Introduction

Applied mathematics and scientific computing are key factors for the development of science and technology of the XX century. An important example of this advance is the modern numerical weather prediction (NWP). Indeed, weather prediction was one of the dreams for the humanity. The Norwegian physicist and meteorologist Vilhelm Bjerknes, around the year 1904, stated that weather prediction can be formulated as an initial value problem. The mathematical equations to be solved were the Navies-Stokes equations modeling the fluid flow dynamics.

However, the first attempt to address the problem figured out by Bjerknes was the English scientist Lewis Fry Richardson. Richardson published a book describing a numerical process to

[1]rpsouto@lncc.br
[2]marowelter@gmail.com
[3]Oosthoff@lncc.br
[4]helaine.furtado@gmail.com
[5]julio.tota@ufopa.edu.br
[6]juliana.anochi@inpe.br
[7]marcelo.paiva@inpe.br
[8]haroldo.camposvelho@inpe.br
[9]vdias@ita.br
[10]sabrinabms@gmail.com

predict weather dynamics [16]. But, the forecasting shown in Richardson's book failed. There are, at least, two reasons to understand why that prediction did not work: there was no constrain between time and space discretizations (Richardson used an explicit method for time integration), and the pressure and wind atmospheric fields were out of balance [15]. The second issue shows the necessity to focus on estimation of the initial condition [6]

Only with the development of the numerical analysis, it was possible to apply stable numerical algorithms for solving time evolution partial differential equations. Charney, Fjørtoft and Von Neumann [4] showed sthe first successful weather prediction by using computer methods. Nowadays, very sophisticated numerical software is employed for the NWP. Beyond to solve the Navier-Stokes equations, many physical processes need to be modeled, such as radiation, cloud formation, precipitation, surface-atmosphere interaction, turbulence, for citing few effects. In fact, the NWP is in a permanent development [1].

Better prediction is linked to calculate a good initial condition. The procedure to compute the best initial condition as possible is called *data assimilation* (DA). The DA is a process for data fusion between observation and previous forecasting [11]. With the amount of data to be processed, the DA becomes the more intense computer task for the NWP into operational centers. This motivated the development of new algorithms based on artificial neural networks for speeding up the DA procedure [5]. A multi-institutional effort to develop a framework for testing new algorithms and software/hardware strategies was established by using cloud computing (Google CoLab platform), and Octave free software. This paper describes the scientific software development of the platforms developed for testing new algorithms for DA applying self-configuring neural networks. Two dynamical systems can be used for testing.

2 Dynamical models

Two dynamical systems were selected for testing methods for data assimilation: Lorenz system [13], and a simplified model to simulated ocean circulation on limited area [2].

2.1 Lorenz System

Edward Lorenz studying the behavior of the atmosphere presented a system of three non-linear coupled differential equations [13]

$$\begin{aligned} dx/dt &= \sigma\,(y-x)\ , & (1)\\ dy/dt &= \rho\,x - y - x\,z\ , & (2)\\ dz/dt &= x\,y - \beta\,z\ . & (3) \end{aligned}$$

The system above shows a *chaotic dynamics* with the parameter values: $(\sigma, \rho, \beta) = (10, 28, 8/3)$. Due to non-linear system of equations, with strong dependence of the initial conditions, the Lorentz system has been used for testing DA algorithms [9].

2.2 Shallow Water Equations

For multi-dimensional system, the model used in the Bennett's book [2] is also employed here. The ocean circulation is simulated by a shallow water 2D (SW-2D) approach, for $t > 0$:

$$\begin{aligned} \partial u/\partial t - f v + g\,(\partial q/\partial x) + r_u u &= -C_d\,\rho_a\,u_a^2/(H\,\rho_w) & (4)\\ \partial v/\partial t + f v + g\,(\partial q/\partial y) + r_v v &= 0 & (5)\\ \partial q/\partial t + H\,[(\partial u/\partial x) + (\partial v/\partial y)] + r_q q &= 0 & (6) \end{aligned}$$

where (u, v) are the ocean stream components, q is the ocean surface depth, H is the average ocean depth, f is the Coriolis parameter, and (r_u, r_v, r_q) are damping coefficients [2]. In equation (4), C_d is the drag parameter, (ρ_a, ρ_w) are air and water (ocean) densities, respectively, and u_a is the surface zonal wind. The space domain is a limited area: $\Omega(x, y) \equiv (0, L_x) \times (0, L_y)$, with appropriated boundary conditions on $\partial\Omega$ [2].

3 Data Assimilation Methods

3.1 Kalman filter

Kalman filter (KF) is well known scheme to estimate a dynamical state by a weighted average between the model prediction and measured data. The dynamical system is assumed to be a Gaussian stochastic process. The weighted average is computed by the *Kalman gain* (K). The algorithm for DA can be summarized as follows:

1. Prediction and observation equations:

$$\mathbf{x}^p_{n+1} = \mathbf{F}_n\,\mathbf{x}^p_n + \mu_n \tag{7}$$
$$\mathbf{y}^p_{n+1} = \mathbf{H}_{n+1}\,\mathbf{x}^p_{n+1} + \nu_n \tag{8}$$

2. Covariance matrix for the prediction, and Kalman gain:

$$\mathbf{P}^p_{n+1} = \mathbf{F}_n\,\mathbf{x}^p_{n+1}\,\mathbf{F}^{\mathrm{T}}_n + \mathbf{W}^{\mathrm{Mod}}_n \tag{9}$$
$$\mathbf{K}_{n+1} = \mathbf{P}^p_{n+1}\,\mathbf{H}^{\mathrm{T}}_{n+1}\,\left[\mathbf{W}^{\mathrm{Obs}}_n + \mathbf{H}^{\mathrm{T}}_{n+1}\,\mathbf{P}^p_{n+1}\,\mathbf{H}^{\mathrm{T}}_{n+1}\right] \tag{10}$$

3. Compute the *analysis* (DA process), and covariance matrix for the analysis:

$$\mathbf{x}^a_{n+1} = \mathbf{x}^p_{n+1} + \mathbf{K}_{n+1}\,\left[\mathbf{y}^{\mathrm{Obs}}_{n+1} - (\mathbf{H}_{n+1}\,\mathbf{x}^p_{n+1})\right] \tag{11}$$
$$\mathbf{P}^a_{n+1} = \left[\mathbf{I} - \mathbf{K}_{n+1}\,\mathbf{H}_{n+1}\right]\,\mathbf{P}^p_{n+1} \tag{12}$$

In the Kalman filter process, μ_n and ν_n are random white Gaussian noises, matrices $\mathbf{W}^{\mathrm{Mod}}_n$ and $\mathbf{W}^{\mathrm{Obs}}_n$ are covariances for model and observation errors, respectively, $\mathbf{K}_n$ is the Kalman gain, $\mathbf{x}^p_{n+1}$ and $\mathbf{x}^a_{n+1}$ are predicted and analysis state vectors, the first equation (7) represents our discrete dynamical system. The KF algorithm can be adapted for non-linear dynamical systems [7].

3.2 Self-configuring neural network

There are many successful applications with the use of artificial intelligence (AI). One of the most employed AI approaches is the artificial neural network (NN). However, applying NN for DA is relatively recent. Cintra and Campos Velho applied a supervised multilayer perceptron (MLP), using back-propagation algorithm for the learning process, to emulate the Kalman filter – with a significant reduction of the CPU-time [5].

Basically, NN is a non-linear mapping between inputs and outputs. In our application, the inputs are the prediction state ($\mathbf{x}^p_{n+1}$) and the observation vector ($\mathbf{y}_{n+1}$), with the output being the analysis ($\mathbf{x}^a_{n+1}$). One challenge is to determine the best architecture to the NN to be employed. There are several hyperparameters to be selected: number of hidden layers, number of neurons for each hidden layer, type of activation function, number of epochs, and the training phase parameters – η and α, learning and momentum rates, respectively.

Our approach is to formulate the NN configuration as an optimization problem. The objective function to be minimized is given by:

$$J(\mathbf{Q}) = penalty \times \left[\frac{\rho_1 E_{Train} + \rho_2 E_{Gen}}{\rho_1 + \rho_2} \right] \quad (13)$$

$$penalty = c_1 \exp(\#\{\text{neurons}^2)\} + c_2 \{\#(\text{epochs})\} + 1 \quad (14)$$

vector $\mathbf{Q}$ denotes the NN hyperparameters to be identified, (E_{Train}, E_{Gen}) are learning and generalization errors, respectively, (ρ_1, ρ_2) are parameters to indicate our attention to the error to be minimized (we adopt: $\rho_1 = \rho_2 = 1/2$). The penalty term indicates that we are looking for the simplest architecture to the MLP-NN, with the smallest number of artificial neurons and the fastest convergence during the learning process for computing the connection weights. The optimization problem (13) is solved by the MPCA (multi-particle collision algorithm) meta-heuristic [14].

4 Octave package in the Google-CoLab platform

Originally, we start to develop our computer codes using Fortran language. When more people were joining this research, applying NN to the DA, one strategy to amplify and make easier the development/testing of algorithms was to use modern platforms using computing resources available on the internet. Such a strategy allows the employment of different computer languages, hardware resources, and cloud computing environments. As a first comment, *Google-Colab* [10] is a free cloud computing environment, where the users have access to the CPU multi-core, GPU and TPU co-processors. Code executions on Google-Colab use Python language.

Beyond Python language, other computer languages can also be used in the Colab platform, as `R` and Octave languages. GNU Octave [8] is a programming standard to deal with scientific and numerical computations, and it is mostly compatible with MATLAB. Part of our development was migrate to the Octave language.

The MPCA Fortran code for self-configuring neural network architecture, with worked example, is available by accessing the internet address: https://github.com/sabrinabms/RNA-MPCA. There is MPCA Python version to the Colab, accessing: https://colab.research.google.com/github/sabrinabms/RNA-MPCA/blob/main/rna%26mpca_local.ipynb. Shallow water system (see Section 2) executions applying KF and NN approaches for DA can be run accessing Fortran code by https://github.com/robertopsouto/kfs2d_rna_mirror. Executions for shallow water system by the Octave language on the Colab can be accessed at: https://colab.research.google.com/drive/1IBLHruTzAuMqKLfZLTkJH575FCnjOhVk .

5 Data assimilation results

Figure 1 illustrates the relevance of the DA process. In the numerical experiment, the dotted blue line represents the *true world* (reference), where the initial condition for the prediction has a smaller difference related to true values. The green square boxes are observations – the true values added to a small random forcing. Figure 1 Figure 1 shows three time periods: during the first and the last ones, there is no visual difference between the simulated and the reference dynamics, while in the middle part of the plot – without DA procedure – there is a full disagreement between the two dynamics (reference and simulation (red line)). Therefore, an acceptable prediction under a chaotic dynamical regime is only possible by applying a data assimilation scheme.

The result of applying the MPCA for self-configuring the MLP-NN to emulate the KF approach to the shallow water 2D model is displayed in Figure 2. Simulations were executed considering 25 measures in the domain for the q-variable only. It is easy to realize that the analysis produced by the

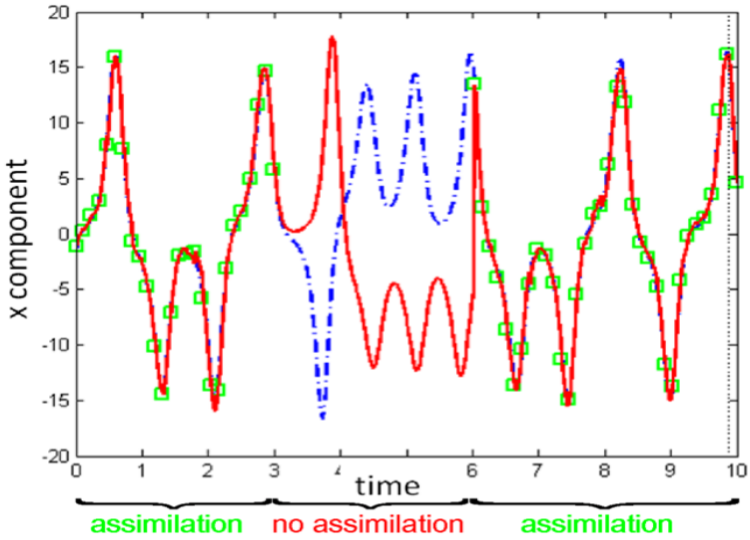

Figura 1: Lorenz system: prediction (red line) with DA and without DA – see [7].

MLP-NN with optimal architecture (*ANN*: dotted lines) is closer to the reference values (*TRUE*: strong continuous lines) than the analysis computed with KF (*KF*: weak continuous lines). These results were obtained with Fortran code. The CPU-time for KF was approximately 42 minutes, while NN spends about 1 minute and a half, indicating that NN was more than 25 times faster than KF.

For numerical experiments with Lorenz and shallow water systems, the assimilation cycle is activated after each 10 time-steps. Table 1 shows the root least mean square error (RMSE) from the experiments with Lorenz system and shallow water equations.

Tabela 1: DA error for KF and NN assimilation schemes.

Lorentz system	KF error	NN error	Shallow water	KF error	NN error
x	0.3391	0.3781	u	0.0296	0.0199
y	0.3862	0.3423	v	0.8554	0.7794
z	0.9757	0.3850	q	0.5969	0.1460

6 Final Remarks

The goal of the present paper is to share the information of a scientific computing environment for developing/testing new algorithms for data assimilation applied to two dynamical models. Other results using the TPU (Tensor Processing Unit) co-processor were obtained with the Colab platform – it is not shown here: partial result for the D.Sc. thesis of one of the authors. The environment for DA was used during the short course held in February 2023, organized and promoted by the National Laboratory of Scientific Computing (LNCC: Laboratório Nacional de Computação Científica), Petrópolis (RJ, Brazil) [12].

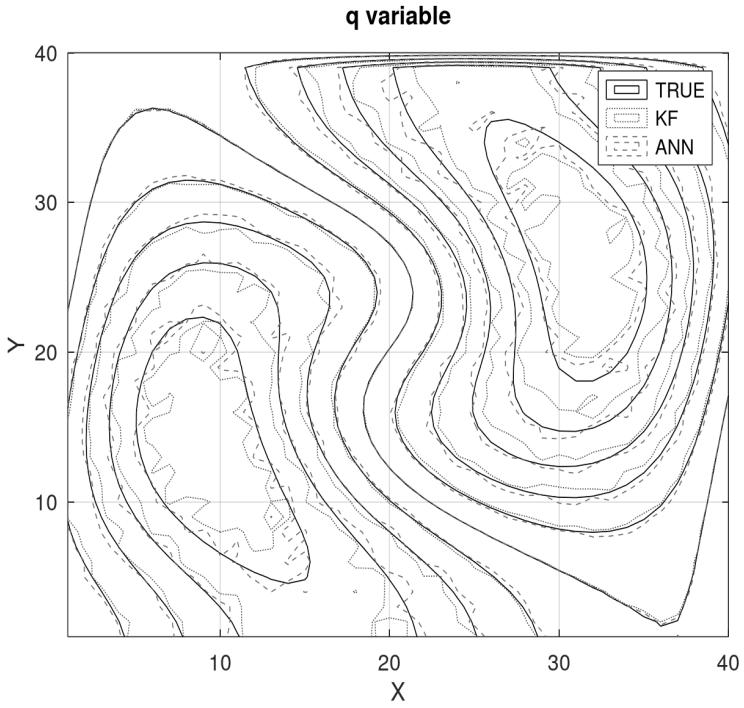

Figura 2: Isovalue curves for ocean surface level $q(x, y)$ – see [3].

Acknowledgments

Authors wish to thank the Google Colaboratory development platform (Colab), National Council for Research and Development (CNPq) – Brazilian agency for research support – for the financial project support (Grant: 422614/2021-1, and Grant: 314660/2020-8 (author: HFCV)).

References

[1] P. Bauer, A. Thorpe, and G. Brunet. "The quiet revolution of numerical weather prediction". **Nature**, 4.525 (2015), pp. 47–55.

[2] A. F. Bennett. **Inverse Modeling of the Ocean and Atmosphere**. Cambridge University Press, 2002.

[3] H. F. Campos Velho et al. "Data assimilation by neural network for ocean circulation: parallel implementation". **Supercomputing Frontiers and Innovations**, 9.1 (2022), pp. 74–86. URL: https://superfri.org/index.php/superfri/article/view/419.

[4] J. Charney, R. Fjørtoft, and J. von Neumann. "Numerical integration of the barotropic vorticity equation". **Tellus** 2.4 (1950), pp. 237–254.

[5] R. S. C. Cintra, and H. F. Campos Velho. "Data assimilation by artificial neural networks for an atmospheric general circulation model". **Advanced Applications for Artificial Neural Network**. Ed. by A. El-Shahat. InTech, 2008. Chapter 14, pp. 265–285, DOI: 10.5772/intechopen.70791. (URL: https://www.intechopen.com/chapters/57304)

[6] R. Daley. **Atmospheric Data Analysis Atmospheric Data Analysis**. Cambridge University Press, 1993.

[7] H. Furtado, H. F. Campos Velho, and E. E. N. Macau. "Assimilação de dados com redes neurais artificiais em equações diferenciais". **Conferência Brasileira de Dinâmica, Controle e Aplicações**. 2011. DOI: 10.13140/2.1.5055.1687.

[8] GNU-Octave. **Octave programming language**. On-line. Access 30/March/2023, https://octave.org/.

[9] M. Goodliff, J. Amezcua, and P. J. Van Leeuwen. "Comparing hybrid data assimilation methods on the Lorenz 1963 model with increasing non-linearity". **Tellus A: Dynamic Meteorology and Oceanography** 67.1 (2015), p. 26928.

[10] Google-Colab. **Free cloud service by Jupyter notebooks**. On-line. Access 30/March/2023. https://colab.research.google.com/

[11] S. Haykin. **Neural Networks: A Comprehensive Foundation**. Prentice Hall Inc., 1994.

[12] E. Kalnay. **Atmospheric Modeling, Data Assimilation and Predictability**, Cambridge University Press, 2002.

[13] LNCC-MCTI. [MC-A01] **Assimilação de Dados por Redes Neurais Artificiais**. On-line. Access 30/March/2023, colorblue http://veraolncc.kinghost.net/MinicursosAvulsos.php.

[14] E. N. Lorenz. "Deterministic nonperiodic flow". **Journal of the Atmospheric Sciences** 20.2 (1963), pp. 130–141.

[15] E. F. P. Luz, J. C. Becceneri, and H. F. Campos Velho. "A new multi-particle collision algorithm for optimization in a high performance environment". **Journal of Computational Interdisciplinary Sciences** 1.1 (2008), pp. 3–10. DOI: 110. 6062/jcis.2008.01.01.0002. (ULR; https://www.epacis.net/jcis/PDF_JCIS/JCIS11-art.01.pdf)

[16] P. Lynch. **The Emergence of Numerical Weather Prediction: Richardson's Dream**. Cambridge University Press, 2006.

[17] L. F. Richardson. **Weather Prediction by Numerical Processes**. Cambridge University Press, 1992.