Prediction of Hilbertian autoregressive processes : a Recurrent Neural Network approach

André Mas, Clément Carré

Pub. Inst. Stat. Univ. Paris
63, fasc. 2-3, 2019, 115-128
Numéro spécial en l'honneur des 80 ans de Denis Bosq /
Special issue in honour of Denis Bosq's 80th birthday

Prediction of Hilbertian autoregressive processes : a Recurrent Neural Network approach

Clément Carré* **and André Mas*,†**

*BionomeeX, Montpellier, France * and IMAG, Univ. Montpellier, CNRS, Montpellier, France†*

Abstract: The autoregressive Hilbertian model (ARH) was introduced in the early 90's by Denis Bosq. It was the subject of a vast literature and gave birth to numerous extensions. The model generalizes the classical multidimensional autoregressive model, widely used in Time Series Analysis. It was successfully applied in numerous fields such as finance, industry, biology. We propose here to compare the classical prediction methodology based on the estimation of the autocorrelation operator with a neural network learning approach. The latter is based on a popular version of Recurrent Neural Networks : the Long Short Term Memory networks. The comparison is carried out through simulations and real datasets.

In memory of Besnik Pumo

1. Introduction

The contribution of Denis Bosq to functional data analysis and modeling is major for several reasons. First of all his work has the flavor of pioneering steps. The article [4] dates back to the very beginning of functional data. Of course some earlier papers investigate functions as statistical observations such as [12], [17] but these authors usually confine themselves to infer without modeling, restricting to first and second order moment estimation or to correlation analysis. The second interesting point comes from the model itself : the ARH(1) as defined in [4] and soon studied by late Besnik Pumo and Tahar Mourid in their PhD thesis. It is seemingly the first model acting on functional data. The linear regression model appears only a few years later in [7]. The ARH(1) will pave the way towards a specific domain of FDA which reveals fruitful: functional time series and processes. The reader interested with this topic is referred to [1, 16] and references therein for instance. At last the works by Denis Bosq had a clear methodological impact by introducing tools from fundamental mathematics and connecting statistics with functional analysis and operator theory.

The ARH model has been widely investigated and generalized in several directions. The underlying Hilbert space was replaced by a space of continuous functions in [24] then generalized to Banach spaces in [22]. The autoregressive operator was

Keywords and phrases: Functional Data, ARH process, LSTM networks, Prediction

extended to linear processes with early results in [21]. The celebrated monograph [5] sums up the main results on the topic (see also later [6]). At last some authors proposed variants of the original ARH by including derivatives (see the ARHD model [20]) or by adding exogenous variable ([10] and their ARHX model).

The outline of the paper is the following. The ARH(1) model is introduced in the next section as well as the classical estimation procedure. Then we summarize Long Short Term Memory blocks that were selected for a numerical comparison. The results of our experiments are given in the last section. We first treated a large simulated dataset, then compared two temperature datasets and finally focused on a synthetic non-linear process.

2. The framework and the model

Let $\mathbf{H}$ be a real separable Hilbert space endowed with an inner product $\langle .,. \rangle$ and a norm derived from it denoted $\|.\|$. In the rest of the paper the space $\mathbf{H}$ is the function space $L^2(\Omega)$, where Ω is assumed to be a real compact interval, usually $[0,T]$ for $T > 0$. The space $\mathbf{H}$ could as well be of $W^{m,2}(\Omega)$ a Sobolev space with regularity index m.

$$W^{m,2} = \left\{ f \in L^2(\Omega) : f^{(m)} \in L^2(\Omega) \right\}.$$

We will consider in the sequel a sample $(X_1, ..., X_n) \in \mathbf{H}^n$. When X_1 is of functional nature its whole path is assumed to be observed. The expectation $\mathbb{E}X$ is a vector of $\mathbf{H}$ whenever it exists, satisfying $\mathbb{E}(< X, u >) =< \mathbb{E}X, u >$ for all $u \in \mathbf{H}$. The covariance operator of X is denoted Γ. It is a positive, symmetric linear operator from $\mathbf{H}$ to $\mathbf{H}$ defined by :

$$\Gamma = \mathbb{E}\left[(X - \mathbb{E}X) \otimes (X - \mathbb{E}X)\right]$$

where $u \otimes v = \langle u, \rangle v$ is the tensor product notation for rank-one operators. The operator Γ is trace-class and self-adjoint whenever $\mathbb{E}\|X\|^2 < +\infty$. The centered autoregressive Hilbertian model reads :

$$X_{n+1} = \rho(X_n) + \varepsilon_{n+1}, \qquad n \in \mathbb{Z} \tag{2.1}$$

where $(\varepsilon_n)_{n\in\mathbb{N}}$ is a Hilbertian strong white noise and ρ is a bounded linear operator acting from $\mathbf{H}$ to $\mathbf{H}$. The model is studied in detail in [5]. Let $\|\rho\|_{\mathcal{L}}$ denote the classical -uniform- operator norm of ρ. We remind the reader this basic but crucial fact (see ibidem Theorem 3.1 p 74) : if $\|\rho\|_{\mathcal{L}} < 1$ then the process X_n solution of (2.1) is uniquely defined and stationary. In the sequel we assume that $(X_n)_{n\in\mathbb{Z}}$ is both stationary and centered.

Estimation of ρ is a difficult problem. Due to the functional framework, likelihood approaches are untractable in a truly infinite-dimensional framework. It can

be shown that ρ is the solution of a specific inverse problem. Namely if $D = \mathbb{E}\left[\langle X_n, \cdot \rangle X_{n+1}\right]$ is the cross-covariance of order 1 of the process :

$$D = \rho \cdot \Gamma. \tag{2.2}$$

The trouble with the above equation is that Γ^{-1} does not exist unless Γ is one-to-one. Then it is an unbounded linear operator, though measurable, and is defined on a domain $\mathcal{D} \subsetneq \mathbf{H}$. This domain is Borel-measurable but neither open nor closed and dense in $\mathbf{H}$. As a consequence deriving from (2.2) $\rho = D \cdot \Gamma^{-1}$ is not correct since ρ is defined on the whole $\mathbf{H}$ whereas Γ^{-1} is not.

Any reasonable estimation procedures should simultaneously estimate D and Γ and regularize the latter in order to define say "$\hat{\Gamma}^{\dagger}$", approximation of Γ^{-1}. The estimation of D and Γ is usually simple though their empirical version :

$$\hat{\Gamma}_n = \frac{1}{n}\sum_{i=1}^{n-1} X_i \otimes X_i \qquad \hat{D}_n = \frac{1}{n-1}\sum_{i=1}^{n-1} X_i \otimes X_{i+1}.$$

At this point note that two smoothing strategies may be applied to stabilize the previous estimates : either smoothing the data (e.g. spline smoothing or decomposition in a basis of smooth function space) or smoothing the covariance operators only.

Approximation of Γ^{-1} is usually more tricky and requires the computation of a regularized inverse denoted $\hat{\Gamma}^{\dagger}$ above. This may be done directly by methods that are classical in inverse problem solving. For instance a ridge estimate provides then $\hat{\Gamma}_n^{\dagger} = (\Gamma_n + T_n)^{-1}$ where T_n is a regularizing (Tikhonov) matrix usually taken as $\alpha_n \mathbf{I}$ where $\alpha_n > 0$, $\alpha_n \downarrow 0$ and $\mathbf{I}$ denotes the identity matrix. Spectral (PCA based) regularization involve the random eigenelements of Γ_n, denoted $(\lambda_{i,n}, \phi_{i,n}) \in \mathbb{R}^+ \times \mathbf{H}$ where $\Gamma_n \phi_{i,n} = \lambda_{i,n} \phi_{i,n}$. A classical output is then :

$$\hat{\Gamma}_n^{\dagger} = \sum_{i=1}^{k_n} \frac{1}{\lambda_{i,n}} \phi_{i,n} \otimes \phi_{i,n}.$$

where k_n must be selected accurately.

Following again (2.2) an estimate of ρ then writes :

$$\hat{\rho}_n = \hat{D}_n \cdot \hat{\Gamma}_n^{\dagger}$$

The predictor is $\hat{\rho}_n\left(X_{n+1}\right)$ and stems from the preceding equation. Note that the evaluation of $\hat{\rho}_n$ at X_{n+1} simplifies the object under concern (the predictor is in $\mathbf{H}$ whereas ρ is an operator on $\mathbf{H}$) and has a smoothing effect on the inverse problem mentioned above. Other results and further details may be found in [5, 19].

3. Long Short Term Memory Networks in a nutshell

The question of predicting time series from neural networks is absolutely not new, see [2]. When addressing the specific issue of prediction in time series, especially

functional time series, Recurrent Neural Networks (RNN) appear as a natural and potentially effective solution. The basic RNNs architecture links a sequential input X_n -typically with a stochastic dependence between X_n and its past- with an output Y_n through an hidden layer H_n. The sequence H_n is often compared with the hidden state in Hidden Markov chain modeling. We refer to the beginning of [18] for a nice presentation of RNN's. The system is driven by the two following equations :

$$\begin{cases} H_n = \sigma_h\left(\mathbf{A}X_n + \mathbf{B}H_{n-1}\right) \\ Y_n = \sigma_y\left(\mathbf{C}H_n\right) \end{cases} \tag{3.1}$$

where $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$ are matrices and σ_h and σ_y are two sigmoidal activation functions. Note that the previous matrices are fixed and not updated in the learning step. This specific structure enables the hidden layer to keep a memory of the past. As a consequence RNNs were successfully applied in speech recognition and more generally in treating dependent data indexed by time. Numerous variants of the RNN were proposed, many of them trying to make the network deeper, see [23].

One of the most efficient variants of RNN are Long Short Term Memory units, trying to overcome the relative unability of RNN to capture long term dependence. They were introduced in the late 90's in [14]. Several tutorials may be found on the internet about LSTM. We give a sketch of the way LSTMs run but we refer the reader to [13] for a formal description. A key improvement in LSTMs over RNN relies on the addition of a cell state to the hidden space H_n that appears in (3.1). Figure 1 shows the architecture of the single block LSTM which was used in this work. The cell state for unit n is a vector denoted C_n. The cell state and the hidden state influence each other through three channels, also called gates. Roughly speaking C_n updates H_n within the LSTM block and will keep along the different layers the truly important information. The three gates may be described in a few words. A first "Forget" gate sweeps off the unimportant coordinates in the new input and in the current hidden state. Then the "Input" gate aims at updating the cell state from C_n to C_{n+1}. It applies a filter similar to the Forget gate on the concatenated vector (H_{n-1}, X_n). In parallel a tanh activation function is applied to the same vector, exactly like a single layer neural network. Then an Hadamard-product (coordinate-wise multiplication) merges the two preceding vectors. The by-product is added to the cell state posterior to the Forget Gate. The last step is the "Output Gate" that first scales the current C_n then filters (H_{n-1}, X_n) through a last sigmoid function. The resulting two vectors are Hadamard-multiplied, simultaneously generating the output and updating H_{n-1} to H_n.

LSTMs gave rise to numerous variants. For instance some connections may be added between the three gates mentioned above and the current value of the cell state (referred to as "peephole" connections). Conversely the LSTM architecture may be simplified like in the Gated Recurrent Unit ([8]).

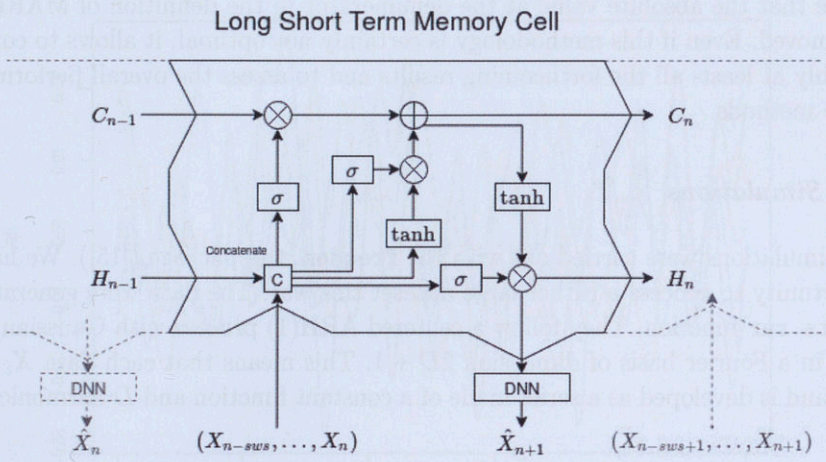

FIGURE 1. *Architecture of a LSTM block/unit*

4. Numerical Experiments

Below we consider only the one-step (functional) predictor : $\hat{\rho}_n(X_{n+1})$. Keep in mind that this predictor provides a forecast of the whole path of the functional data on its period typically. All this comes finally down to a multi-step prediction in terms of univariate time-series.

After testing different stategies to evaluate our numerical results we adopted the following methodology. First we decided not to consider the rough Mean Square Error since it depends on the data's range and does not allow a comparison between datasets and methods. The MSE may also be hard to interpret. The Mean Absolute Relative Error (MARE) will be defined this way in our framework (we are aware that concatenating Absolute and Relative in the same acronym is not especially elegant) :

$$\text{MARE} = \frac{1}{n_{te}} \sum_{i=1}^{n_{te}} \text{Abs}\left(X_i, \hat{X}_i\right)$$

where $\hat{X}_i = \hat{\rho}_{n_{tr}}(X_i)$, n_{te} is the size of the testing set and :

$$\text{Abs}\left(X_i, \hat{X}_i\right) = \sum_{j=1}^{T} \frac{\left|X_i(t_j) - \hat{X}_i(t_j)\right|}{|X_i(t_j)|}.$$

The integer T is the (time-)grid size and the t_j's are the discretization times . One of the problems with the above definition is that the denumerator may be null or very small. In order to avoid this problem all datasets were normalized to $[0.01, 1]$. Others normalizations were tested without any clear impact on the stability of the results.

Notice that the absolute value at the denumerator in the definition of MARE may be removed. Even if this methodology is certainly not optimal, it allows to compare -roughly at least- all the forthcoming results and to assess the overall performances of the methods.

4.1. Simulations

The simulations were carried out with the `freqdom.fda` package ([15]). We had the opportunity to process a rather large dataset this way. The data were generated by the `fts.rar` function. They follow a centered ARH(1) process with Gaussian white noise in a Fourier basis of dimension $2D+1$. This means that each data X_i obeys (2.1) and is developed as a series made of a constant function and D harmonics such as :

$$X_i(t) = a_0 + \sum_{k=1}^{D} \left\{ a_k^{(i)} \cos(2\pi k t) + b_k^{(i)} \sin(2\pi k t) \right\}$$

where $a_k^{(i)}$ and $b_k^{(i)}$ are sequences of real random variables. In order to ensure stationarity 50 burning iterations of the processes were conducted. The scenarii for the simulations depend on :

- Three different values for the Fourier basis dimension D,
- Two different autocorrelation operators ρ described just below.

The default autocorrelation operator of `fts.rar` was used. It is a large dimensional matrix whose row i-column j cell $\rho_{i,j}$ is proportional to $exp(|i-j|)$ hence rapidly decreasing out of the diagonal. We also investigated the situation when the cells decrease more slowly : $\rho_{i,j} \propto \frac{1}{1+|i-j|^2}$. In both cases the Hilbert-Schmidt norm of ρ was fixed so that $\|\rho\|_{HS} = 0.5$.

Once generated in the basis the data were evaluated on a regular grid of size 500 in order to draw them and to compute their norms. The sample size was 1000. The data are consequently collected in a (500×1000) matrix. An overview of the data is given at Figure 2.

For both methodologies the initial dataset was first split in three subsets of size $n_{tr} = 600$, $n_v = 200$ and $n_{te} = 200$ respectively for training, validation and test. In the classical approach, the data matrix was processed as an fdata object by the `far` function of the package `far` by S. Guillas and J. Damon ([11]). The previous package carries out the estimation of ρ by the spectral cut (PCA) methodology and the prediction. The cross-validation step correctly determines the optimal value of k_n, around $2D+1$ in all situations.

The Neural Network part was conducted under Keras (see [9]) with TensorFlow 2.0 as backend. The LSTM unit is followed by a dense layer whose output size equals the grid size (here 500). The learning rate for gradient descent was fixed to

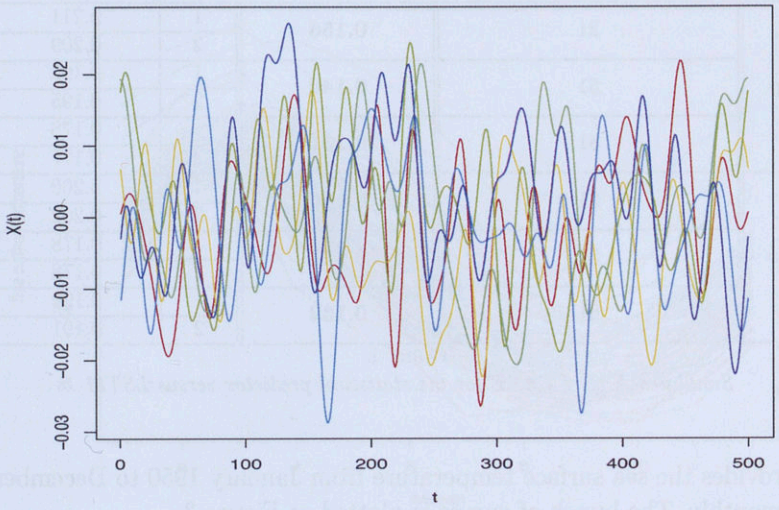

FIGURE 2. *Sample of size 5 of the simulated dataset with $D = 21$ and ρ of exponential type*

1e-4. The training step is stopped when the MARE does not decrease anymore on 5 consecutive epochs on the validation set. The best epoch is then used for the testing step. The LSTM was carried out by taking into account the data in a sliding window of varying size (denoted SWS below). Here since the data are simulated according to an ARH(1) this optimal SWS is 1. In the case of an ARH(p) it would be obviously p.

Table 4.1 displays the MARE values. We notice first that the autoregression operator structure (exponential or power 2) does not seem to have a clear impact. The MARE generally decreases when the latent dimension D increases. A penalization term should certainly be added to balance this side-effect. Remind however that our goal here is to compare two methodologies. The ARH model was always optimally calibrated and provides the best results which is not surprising. We checked that the MARE decreases logically when the noise level in the model shrinks. Conversely the LSTM cell was not specifically designed for this data. The gap is not wide and seems rather promising in view of application on real data.

4.2. Real data

4.2.1. El Niño

The El Niño dataset is one of the first which was studied in the framework of dependent functional data (see e.g. [3]). Our version comes from the `rainbow` package

ρ type	Effective Dimension (2D+1)	Stat pred MARE	SWS	LSTM MARE
exp	21	**0.156**	1	0.211
			2	0.209
	51	**0.140**	1	0.193
			2	0.195
	81	**0.131**	1	0.178
			2	0.177
pow	21	**0.156**	1	0.200
			2	0.202
	51	**0.141**	1	0.178
			2	0.178
	81	**0.132**	1	0.192
			2	0.191

TABLE 1
Simulated ARH : MARE for the statistical predictor versus LSTM

in R. It provides the sea surface temperature from January 1950 to December 2018 observed monthly. The bunch of curves is plotted at Figure 3.

Out of 69 curves-data, 40 were used for training, 15 for validation and 14 for test. The modest size of the dataset restricted our study to SWS of size 1 and 2 only. The summary of MARE is given in Table 4.2.1.

Stat. Predict MARE	SWS	LSTM MARE
0.226	1	0.301
	2	0.308

TABLE 2
El Niño Dataset : MARE for the statistical predictor versus LSTM

The LSTM is again outperformed by the statistical predictor, but the MARE range, above 20% is not good. At this point we must mention that we were faced with two main numerical issues concerning this meteorological dataset.

First of all, even if we do not aim here at proving (again) the global warming, it seems that this fact could be retrieved from observations of the ten first versus the ten last curves-data as plotted on Figure 4. The ten first are black-solid, the ten last are red-dashed. It is plain that sea temperature for the six first months of observations tend to be higher for recent years. As a consequence the basic assumption on stationarity of the data is not clearly fulfilled.

Second we need to underline the problems encountered when applying the usual strategy based on training, validation and testing for such dependent functional data. As explained earlier the training test is separated from the testing set by a validation interval containing 15 years of data. This strategy is clearly more sensitive to potential non-stationarity or slight perturbation in the model than in the situation where the sample is i.i.d. It results in a potential overfitting. Ideally, validation, training and test should be performed continuously along the sample. But the model

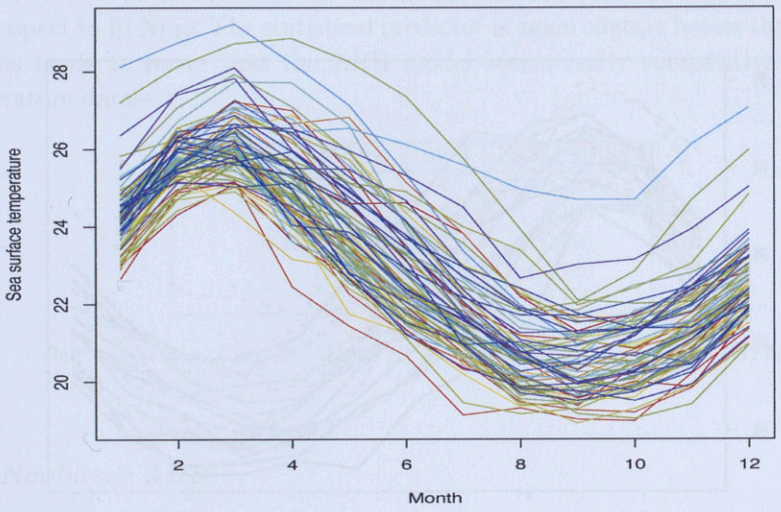

FIGURE 3. *The Sea Surface Temperature in El Niño dataset*

is not adapted to such strategies. Even if these results are not given here we noticed a substantial improvement of the MARE when using only a training and a testing set (without folding) plus a simple grid-search on k_n.

4.3. Bale temperature dataset

It may be interesting to compare the previous popular El Niño file with another temperature dataset retrieved freely from the website `https://www.meteoblue.com/fr/historyplus` and ranging from 1985/1/1 to 2020/12/31. The temperatures are recorded hourly in the city of Bale, Switzerland. We decided to consider the daily agregated data (the daily mean was used) in order to reduce drastically the ratio between the ambient dimension and the sample size. The data matrix is (35×365) because all February 29th records were removed. The reader must notice that the sample size here is $n = 35$ hence the half of El Niño's but the time frequency is the day (against the month). A sample of curves is given in Figure 5.

The learning strategy was similar to El Niño. The prediction error is provided in Table 4.3. Learning and calibration is performed on curves 1 to 30 and prediction on curves 31 to 35. The optimal dimension choice for the ARH predictor is $k_n = 3$. Conversely to El Niño the high sampling frequency of data allowed to explore SWS

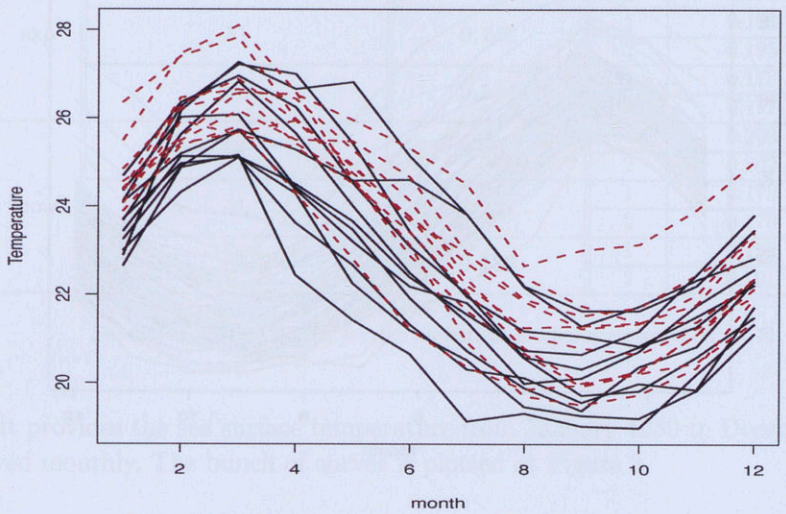

FIGURE 4. *The ten first (black solid) and ten last(red dashed) curves-data in the El Niño dataset*

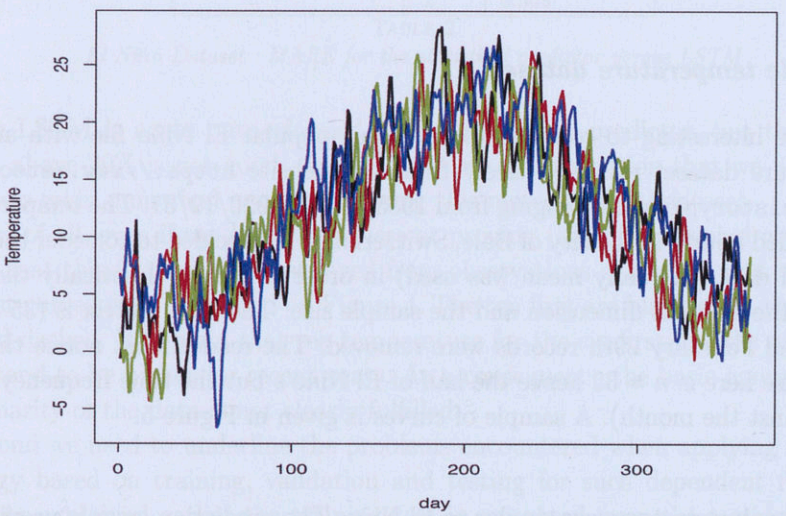

FIGURE 5. *A sample of 4 curves from the Bale temperature dataset. All data were picked in the testing set.*

from 1 to 5. It is noticeable that the MARE are between 10% and 15% and improved with respect to El Niño. The statistical predictor is again slightly better than LSTM. All this tends to prove that the ARH model seems really competitive for these temperature datasets.

Stat. Predict MARE	SWS	LSTM MARE
0.116	1	0.245
	2	0.133
	3	0.126
	4	0.140
	5	0.125

TABLE 3
Bale temperature Dataset : MARE for the statistical predictor versus LSTM

4.4. Nonlinear ARH

Following the remark of a referee we investigated a situation which is less favorable to the ARH predictor and simulated a basic nonlinear functional autoregressive process. Start from a basic ARH equation simulating $X_n = \rho_0(X_{n-1}) + \epsilon_n$. Then construct the nonlinear process the following way :

$$X_n^{n.l}(t) = 3cos(10\pi \cdot X_n(t)) - 2exp(-X_n(t))$$

A sample of four successive curves is plotted on Figure 6. For a fair comparison with previous results, the dimension and sample size are the same as in section 4.1, respectively 500 and 1000.

Stat. Predict MARE	SWS	LSTM MARE
1.235	1	0.647
	2	0.661
	5	0.655

TABLE 4
Nonlinear autoregressive process : MARE for the statistical predictor versus LSTM

The highly non-linear behaviour of $X_n^{n.l}$ is confirmed by the results in Table 4.4. A "quick and dirty" search gives an optimal k_n around 70. The MARE are very high for this synthetic dataset close to a white noise. Anyway, despite this fact, the LSTM performs almost twice better than the statistical predictor.

5. Conclusion

This work attempts to compare the historical/statistical track for prediction in ARH models with a Neural Network approach centered on LSTMs. Data and code are

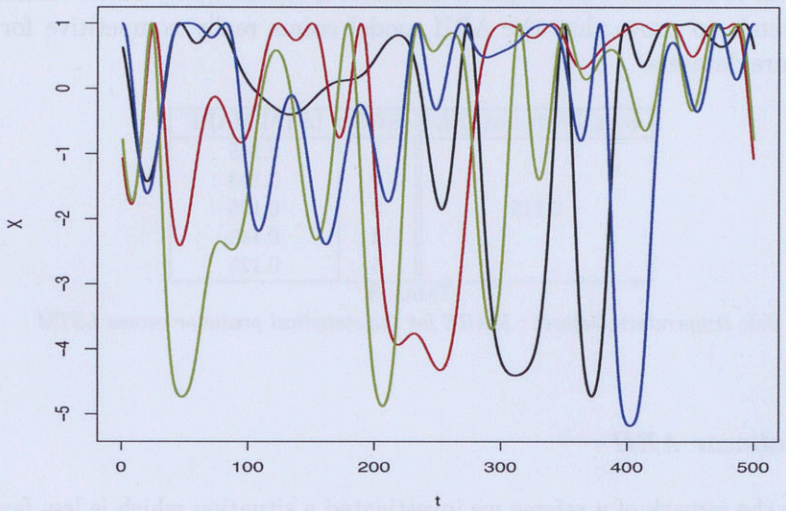

FIGURE 6. *Plot of* $X^{n,l}_{200:203}$ *for an overview of the nonlinear ARH process simulated on a grid of 500 time points. The whole sample size is 1000.*

available at `https://gitlab.com/arh-lstm/`. Several facts should be underlined in order to show the limits of our results.

- We did not study here the impact of the sampling frequency i.e. the size of discretization grid for the functional data. We noticed however some improvement between the El Niño and the Bale dataset. On this basis nothing solid should be stated however. We could have also focused on the effect of the sample size or of the ρ operator norm on the accuracy of the results.
- The architecture used here is simplistic because based on a single LSTM block. Introducing some depth by adding several layers of LSTM should certainly improve the predictions of the simulated dataset. El Niño is certainly not suited to a sequence of cells.
- We used the discretized version of the functional data coming down to a large dimensional input vector (up to size 500 here). Clearly feeding the network with the Fourier coefficient instead leads to a more compact entry and paves the way to another approach.

Our framework was centered on the functional autoregressive process of order 1 and may be restrictive in some way. The design of LSTM is general enough to foster a wider investigation : autoregressive processes of order $p > 1$ or even more general functional times series with linear or non-linear dependence structure. Further work is in progress in order to compare the numerical performance of Neural Networks strategy against functional non-parametric techniques such as kernel-regression in

this setting of dependent functional data.

6. Acknowledgements

We are grateful to an anonymous referee for helpful comments that improved the paper and for suggesting the simulations in section 4.4.

References

[1] Aue, A., Norinho, D. D., and Hörmann, S. (2015). On the prediction of stationary functional time series. *Journal of the American Statistical Association*, 110(509):378–392.

[2] Bengio, S., Fessant, F., and Collobert, D. (1995). A connectionist system for medium-term horizon time series prediction. In *In Proc. Intl. Workshop Application Neural Networks to Telecoms*, pages 308–315.

[3] Besse, P. C., Cardot, H., and Stephenson, D. B. (2000). Autoregressive forecasting of some functional climatic variations. *Scandinavian Journal of Statistics*, 27(4):673–687.

[4] Bosq, D. (1991). Modelization, nonparametric estimation and prediction for continuous time processes. In *Nonparametric Functional Estimation and Related Topics*, pages 509–529. Springer Netherlands.

[5] Bosq, D. (2000). *Linear Processes in Function Spaces*. Springer New York.

[6] Bosq, D. (2007). General linear processes in Hilbert spaces and prediction. *Journal of Statistical Planning and Inference*, 137(3):879–894.

[7] Cardot, H., Ferraty, F., and Sarda, P. (1999). Functional linear model. *Statistics and Probability Letters*, 45(1):11–22.

[8] Cho, K., van Merrienboer, B., Gülçehre, a., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning phrase representations using rnn encoder-decoder for statistical machine translation. In Moschitti, A., Pang, B., and Daelemans, W., editors, *EMNLP*, pages 1724–1734. ACL.

[9] Chollet, F. et al. (2015). Keras. `https://keras.io`.

[10] Damon, J. and Guillas, S. (2002). The inclusion of exogenous variables in functional autoregressive ozone forecasting. *Environmetrics*, 13(7):759–774.

[11] Damon, J. and Guillas, S. (2015). *far: Modelization for Functional AutoRegressive Processes*. R package version 0.6-5.

[12] Dauxois, J., Pousse, A., and Romain, Y. (1982). Asymptotic theory for the principal component analysis of a vector random function: Some applications to statistical inference. *Journal of Multivariate Analysis*, 12(1):136–154.

[13] Greff, K., Srivastava, R. K., Koutnik, J., Steunebrink, B. R., and Schmidhuber, J. (2017). LSTM: A search space odyssey. *IEEE Transactions on Neural Networks and Learning Systems*, 28(10):2222–2232.

[14] Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8):1735–1780.

[15] Hörmann, S. and Kidzinski, L. (2017). *freqdom.fda: Functional Time Series: Dynamic Functional Principal Components.* R package version 0.9.1.

[16] Hörmann, S. and Kokoszka, P. (2010). Weakly dependent functional data. *Ann. Statist.*, 38(3):1845–1884.

[17] Kleffe, J. (1973). Principal components of random variables with values in a separable Hilbert space. *Mathematische Operationsforschung und Statistik*, 4(5):391–406.

[18] Li, S., Li, W., Cook, C., Zhu, C., and Gao, Y. (2018). Independently recurrent neural network (IndRNN): Building a longer and deeper RNN. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition.* IEEE.

[19] Mas, A. (2007). Weak convergence in the functional autoregressive model. *Journal of Multivariate Analysis*, 98(6):1231–1261.

[20] Mas, A. and Pumo, B. (2009). Functional linear regression with derivatives. *Journal of Nonparametric Statistics*, 21(1):19–40.

[21] Merlevède, F. (1996). Central limit theorem for linear processes with values in a Hilbert space. *Stochastic Processes and their Applications*, 65(1):103–114.

[22] Mourid, T. (2002). Estimation and prediction of functional autoregressive processes. *Statistics*, 36(2):125–138.

[23] Pascanu, R., Gulcehre, C., Cho, K., and Bengio, Y. (2014). How to construct deep recurrent neural networks. In *Proceedings of the Second International Conference on Learning Representations (ICLR 2014).*

[24] Pumo, B. (1998). Prediction of continuous time processes by C[0,1]-valued autoregressive process. *Statistical Inference for Stochastic Processes*, 1(3):297–309.

Corresponding author
andre.mas@umontpellier.fr