2014 International Joint Conference on Neural Networks (IJCNN)
July 6-11, 2014, Beijing, China

A Monte Carlo strategy for structured multiple-step-ahead time series prediction

Gianluca Bontempi
Machine Learning Group
Interuniversity Institute of Bioinformatics in Brussels, $(IB)^2$
Université Libre de Bruxelles, ULB
Brussels, Belgium
mlg.ulb.ac.be, ibsquare.be

***Abstract*—Forecasting a time series multiple-step-ahead is a challenging problem for several reasons: the accumulation of errors, the noise, and the complexity of the dependency between past and far future which has to be inferred on the basis of a limited amount of data. Traditional approaches to multi-step-ahead forecasting reduce the problem to a series of single-output prediction tasks. This is notably the case of the Iterated and the Direct approaches. More recently, multiple-output approaches appeared and stressed the multivariate and structured nature of the output to be predicted. This paper intends to go a step further in this direction by formulating the problem of multi-step-ahed forecasting as a problem of conditional multivariate estimation which can be addressed by a Monte Carlo importance sampling strategy. The interesting aspect of the approach is that this probabilistic formulation allows a natural integration of the traditional Iterated and Direct approaches. The extensive assessment of our algorithm with the NN5, NN3 and a synthetic benchmark shows that this approach is promising and competitive with the state-of-the-art.**

I. Introduction

Forecasting the continuation of an observed time series multiple steps forward is a relevant and challenging problem in data mining. The complexity of this problem is due to several aspects: the potential nonlinearity of the dependency between the past and the future, the lack of a priori knowledge, the large noise and the small amount of samples. Two well-known strategies are commonly used to tackle such task: the Iterated (also known as recursive) strategy which iterates a one-step-ahead predictor but suffers of error accumulation and the Direct strategy which decomposes the prediction in a set of independent tasks [1].

In order to improve the accuracy of forecasting strategies, a specific aspect of time series has been recently pointed out: a time series is the realization of a sequential set of random variables with a specific structure deriving by a complex pattern of temporal dependencies [2], [3]. As a consequence such pattern of dependencies can be exploited as an inductive bias to regularize the estimation process and better adapt to the data distribution.

The exploitation of structure in data to impose constraints to the learning process and obtain improved generalization is nowadays common in regression and classification (see for instance the work in multi-task learning [4] or prediction of structured data [5]) but only scarcely adopted in time series prediction.

Also, so far, the use of the data structure in time series prediction has been limited to multi-response regression strategies like the Joint method based on multiple output neural networks proposed by [6] or the LL-MIMO method based on Lazy Learning proposed by [2], [3]. Here we propose an alternative to the multi-response regression approach which is based on the use of a Monte Carlo strategy. The rationale is to use Monte Carlo importance sampling to sample the conditional distribution of the multivariate vector representing the continuation of the time series in a a way that takes into account the structural dependency of the series. What results is a multi-step-ahead forecasting method which could be considered as a probabilistic combination of the Iterated approach and the Direct approach. This is due to the fact that the outcome of the Monte Carlo importance sampling strategy reweighs the set of bootstrap predictions obtained by the Direct method by taking into consideration the constraint represented by the one-step-ahead predictor.

Note that the use of importance sampling in graphical models is well-known, for instance as a way to perform approximate inference [7] or sequential filtering [8]. However in those cases the technique is used for computing the conditional probability of a probabilistic model (e.g. a Bayesian Network) whose structure and parameters are known. In our paper, importance sampling is used instead as a way to enforce a probabilistic constraint (or bias) in a learning algorithm.

After motivating and presenting the algorithm in Section II and III, we perform in Section IV an assessment of its accuracy by means of three common benchmarks in multiple-step-ahed forecasting. The experimental comparison relies on local learning techniques to estimate the conditional distributions.

II. The multiple-step-ahead forecasting problem

In probabilistic terms a time series is the realization of a stochastic process, that is a sequence of random variables indexed by a variable t. A stochastic process is completely determined by the joint distribution of all the variables

$$\{\dots, \mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_{t+1}, \mathbf{y}_{t+2}, \dots\}$$

This distribution summarizes all the dependencies between the past and the future values of the series. In the case of a strictly stationary process the joint distribution of two variables $\mathbf{y}_t$ and $\mathbf{y}_{t+h}$ depends only on h and not on t. Forecasting at time t the next $h > 0$ values of the time series is then possible since

978-1-4799-1484-5/14/$31.00 ©2014 IEEE

the observed data $\{y_1, \ldots, y_t\}$ can provide information about the stochastic dependencies between the past and the future realizations and these dependencies are preserved with time.

A well-known family of forecasting approaches represents the dependency between the value of a series at time $t+h$ and a finite set of p previous values by using the nonlinear autoregressive NAR(p) notation

$$\mathbf{y}_{t+h} = F_h(y_t, y_{t-1}, \ldots, y_{t-p+1}) + \mathbf{w}_h = F_h(q) + \mathbf{w}_h$$

where p is the order of the model and the vector q of length p is commonly denoted as the *embedding vector*. This model implies that the stochastic process is a Markov process, that means that, once the embedding vector is known, the variable $\mathbf{y}_{t+h}$ is conditionally independent of more ancient measures. This model induces two properties of the conditional distribution of $\mathbf{y}_{t+h}$ given the observed values: the expected value of $\mathbf{y}_{t+h}$ is given by

$$E[\mathbf{y}_{t+h}|y_t, y_{t-1}, \ldots,] = E[\mathbf{y}_{t+h}|y_t, \ldots, y_{t-p+1}] = \\ = F_h(y_t, y_{t-1}, \ldots, y_{t-p+1})$$

and $\mathbf{w}_h$ denotes the conditional distribution of $\mathbf{y}_{t+h} - E[\mathbf{y}_{t+h}|y_t, \ldots, y_{t-p+1}]$.

In case of linear F and $h = 1$ this formulation boils down to the conventional linear autoregressive model AR(p) for one-step-ahead prediction. The same formalism underlies the two most common techniques for multistep-ahead forecasting where the goal is to forecast the next $H > 1$ values of a series observed up to time t: the Iterated and the Direct approach. In the Iterated approach the data are used to infer the one-step-ahead dependency

$$\mathbf{y}_{t+1} = F_1(y_t, y_{t-1}, \ldots, y_{t-p+1}) + \mathbf{w}_1$$

and the estimated model $\hat{F}_1$ is used iteratively to provide the set of H predicted values $\hat{y}_{t+1}, \hat{y}_{t+2}, \ldots, \hat{y}_{t+H}$.

In the Direct approach the multiple-step-ahead forecasting task is decomposed in a set of H independent single-output tasks

$$\mathbf{y}_{t+h} = F_h(y_t, y_{t-1}, \ldots, y_{t-p+1}) + \mathbf{w}_h, \quad h = 1, \ldots, H.$$

A detailed discussion of the pros and cons of these two approaches is presented in [2], [3]. To resume, we can say that if on one hand the Iterated approach is more exposed to the accumulation of errors, on the other the Direct approach ignores the conditional dependency between the future values of the series.

In other terms, the two approaches tend to ignore the structured property [5] of a multi-step-ahead forecasting task: indeed when we want to predict the next H values of a time series we aim to solve a learning problem where the nature of the stochastic process implies a dependency between inputs (i.e. the terms of the embedding vector), between outputs (i.e. the H future values to be predicted) and between inputs and outputs.

A way to deal with such dependency is to include a term either in the parametric fitting or the hyperparameters selection of the learner. Examples of how to incorporate correlations into linear regression and SVM are discussed in [9]. The proposed techniques were used in tasks of mass spectrometry prediction and image transformation. In a time series task, the use of structural information to improve the selection of hyper parameters in a nearest neighbor approach has been discussed in [2]. This led to the definition of a multiple input multiple output (MIMO) strategy for long term prediction which has been validated in several contexts [10], [3].

These techniques rely in modifying the conventional cost functional related to prediction accuracy by including a quantitative terms encoding some prior knowledge about the structure of the problem. Such generalization of existing learning techniques demands a sufficiently detailed knowledge about the structure of the dependency in order to introduce it in the learning process.

In the following section we propose an original strategy based on sampling to take into consideration the structural dependencies of a time series in the forecasting procedure.

III. The Structured Monte Carlo forecasting method

In a NAR(p) setting, the prediction of a time series H steps forward demands the estimation of the H conditional expectation terms

$$E[\mathbf{y}_{t+h}|q] = \int y_{t+h} p(y_{t+h}|q) dy_{t+h}, \quad h = 1, \ldots, H \quad (1)$$

where the H variables are distributed according to the conditional and multivariate distribution

$$p(y_{t+H}, \ldots, y_{t+1}|y_t, y_{t-1}, \ldots, y_{t-p+1}) = \\ = p(y_{t+H}, \ldots, y_{t+1}|q) \quad (2)$$

Among the H variables the following structural relation holds for each $h = 2, \ldots, H$ and $j = 1, \ldots, h-1$

$$p(y_{t+h}|q) = \\ \int p(y_{t+h}|y_{t+j}, \ldots, y_{t+1}, q) p(y_{t+j}, \ldots, y_{t+1}|q) dy_{t+j} \ldots dy_{t+1}, \quad (3)$$

Note that if the time series is NAR(p) the expression can be simplified because of the conditional independence properties. For instance if $p = 1$, $h = 3$ and $j = h - 1 = 2$

$$p(y_{t+3}|q) = \\ \int p(y_{t+3}|y_{t+2}) p(y_{t+2}|y_{t+1}) p(y_{t+1}|y_t) dy_{t+2} \ldots dy_{t+1}, \quad (4)$$

The complexity of the estimation problems (1) in real settings is due to several elements: the complex dependency pattern (see for instance the graphical model associated to a NAR(2) in Figure 1), the large dimensionality of the input, the large dimensionality of the output, the potential nonlinearity of the dependencies, the lack of a priori knowledge, the noise and the small amount of samples with respect to the dimensions.

In the Direct approach the estimation of the H terms in (1) is done without taking into account the relation (3).

The Iterated approach instead approximates the relation (3) for $j = h - 1$ by assuming naively that the predictions $\hat{y}_{t+1}, \ldots, \hat{y}_{t+h-1}$ return an accurate estimation of the distribution of $y_{t+1}, \ldots, y_{t+h-1}$ for each $h = 1, \ldots, H$.

In the multi-response approach (e.g. MIMO [2]) the structural constraint (3) is taken implicitly into consideration by fitting a multi-response model and returning a vectorial prediction.

What we propose here is to take explicitly into consideration such constraint by adopting a Monte Carlo sampling strategy. If we were able to generate R samples $y_{t+h}^{(r)}$ according to the conditional distribution $p(y_{t+h}|q)$, the estimation of the quantities (1) would be easy:

$$E[\mathbf{y}_{t+h}|q] \approx \hat{y}_{t+h} = \frac{1}{R}\sum_{r=1}^{R} y_{t+h}^{(r)}$$

Unfortunately the distribution $p(y_{t+h}|q)$ is unknown and we cannot generate samples directly from it. However a possible estimator could be provided by using the Direct strategy. Though such estimator disregards some aspects of the conditional distribution, like the probabilistic constraint (3), we could try to adjust its estimation accordingly in order to take into account the missing information.

The idea of adjusting samples drawn from a *proposal* distribution in order to obtain samples from a *target* distribution, potentially known but impossible to be sampled directly, is the core of the *importance sampling* approach [11]. Suppose we have a target distribution $p(y)$ for which we would like to estimate the expectation by Monte Carlo sampling. Given the impossibility to sample it directly we generate samples from a different *proposal distribution* $\hat{p}(y)$. In general the proposal distribution can be arbitrary with the only requirement that the support of $\hat{p}$ contains the support of p, i.e. $\forall y, \quad p(y) > 0 \Rightarrow \hat{p}(y) > 0$. Since the samples drawn from $\hat{p}$ are incorrect we cannot simply average them to obtain the estimator. The best way to use these samples is to weight them according to the importance they have in representing p, or in other terms, according to their compatibility with p. The resulting normalized importance sampling estimator [7] is

$$E[\mathbf{y}] \approx \hat{y} = \frac{\sum_{r=1}^{R} w^{(r)}\hat{y}^{(r)}}{\sum_{r=1}^{R} w^{(r)}} \tag{5}$$

where $w^{(r)}$ is the importance weight of the rth sample generated according to the proposal distribution $\hat{p}$. Note that this formulation does not require a complete knowledge of the density p but simply a knowledge up to a normalizing constant.

The original idea of this paper is to have recourse to an importance sampling strategy to adjust the Direct approach in order to incorporate the structural constraint (3). So, though in our case $p(y_{t+h}|q)$ is not known, we generate approximate, yet incorrect, samples by the Direct approach (that plays here the role of *proposal distribution* generator) and adjust them by weighting according to their satisfaction of the structural constraint (3). Note that if we consider the two factor terms inside the integral in (3), the first one represents the structural constraint since it imposes that the samples at time $t+h$ and the samples at the previous time instants satisfy the dependency

$$p(y_{t+h}|y_{t+h-1}, \ldots, y_{t+h-p}) \tag{6}$$

in accordance with the NAR(p) setting.

So, if we are able to estimate the conditional distribution (6), as typically done in the Iterated strategy, we can impose a structural constraint by weighting the samples obtained with the Direct method. Once each sample has been weighted accordingly we obtain a structured prediction by computing the weighted average. Note that any supervised learning algorithm (e.g. feedforward neural networks) can be used to estimate (6) from historical data. In Section IV we will have recourse to a local learning algorithm to perform the estimation.

A. The SMC algorithm

Our proposed algorithm, denoted by SMC (Structured Monte Carlo) and detailed in Algorithm 1 is composed of three main parts.

In the first part (lines 3-7) we draw R samples $\hat{y}_{t+h}^{(r)}$ by sampling the conditional distribution $p(y_{t+h}|q)$ for each horizon $h = 1, \ldots, H$. This is made possible thanks to a Direct estimator $\hat{p}(y_{t+h}|q)$ which works here as the generator of the proposal distribution and which is learned from historical data with a conventional learner (e.g. linear model or nearest-neighbour). The samples $\hat{y}_{t+h}^{(r)}$ of the conditional distributions are obtained by sampling with replacement (lines 5-6) the original dataset and retraining the Direct estimator.

In the second part (lines 13-29), we loop over an increasing horizon $h = 2, \ldots, H$. For each horizon h each sample $\hat{y}_{t+h}^{(r)}$ is weighted by a term measuring how much this value is compliant with the constraint (3) by computing (line 25) the quantity

$$w_{t+h}^{(r)} = \sum_{j=1}^{J} p(\hat{y}_{t+h}^{(r)}|\hat{y}_{t+h-1}^{(j)}, \ldots, \hat{y}_{t+h-p}^{(j)}) = \sum_{j=1}^{J} p(\hat{y}_{t+h}^{(r)}|q^{(j)}) \tag{7}$$

Such computation requires the sampling of J embedding vectors $q^{(j)} = [\hat{y}_{t+h-1}^{(j)}, \ldots, \hat{y}_{t+h-p}^{(j)}]$ composed by observed values if $h \leq p$ and by estimated values otherwise (lines 17-22). Note that the sampling at the horizon h is done proportionally to the weight $w_{t+h}^{(r)}$ (line 22).

In accordance with (5), the last phase (lines 30-32) assembles the Direct samples and the importance weights by returning for each horizon h the forecast

$$\hat{y}_{t+h} = \frac{\sum_{r=1}^{R} w_{t+h}^{(r)}\hat{y}_{t+h}^{(r)}}{\sum_{r=1}^{R} w_{t+h}^{(r)}}, \quad h = 1, \ldots, H$$

The SMC algorithm relies on the availability of an estimator of the conditional probability $p(t_{t+h}|q)$, which is implemented by the function DIR. This estimator is the one required to perform the Direct strategy and can be implemented by a conventional linear or nonlinear regression technique. In the following section we will use a local learning regression technique based on [14].

Algorithm 1 SMC

Require: Observed time series $Y = \{y_1, \ldots, y_t\}$, order p, horizon H, number J of embedding vectors

1: $q \leftarrow [y_N, \ldots, y_{N-p+1}]$
2: Put the series in input/output form $X_{(N,p)}$, $O_{(N,H)}$
3: **for** $h = 1$ to H **do**
4: **for** $r = 1$ to R **do**
5: $I^{(r)} \leftarrow$ sample(1:N)
6: $\hat{y}^{(r)}_{t+h} \leftarrow$ DIR($X[I^{(r)},], O[I^{(r)}, h], q$)
7: **end for**
8: **end for**
9: **for** $r = 1$ to R **do**
10: $w^{(r)}_{t+1} \leftarrow 1/R$
11: **for** $h = 2$ to H **do**
12: $w^{(r)}_{t+h} \leftarrow 1$
13: **end for**
14: **end for**
15: **for** $h = 2$ to H **do**
16: **for** $j = 1$ to J **do**
17: $q^{(j)} \leftarrow []$
18: **for** $k = t + h - p$ to $t + h - 1$ **do**
19: **if** $k \leq t$ **then**
20: $q^{(j)} \leftarrow [y_k, q^{(j)}]$
21: **else**
22: $s \leftarrow$ sample$(1 : R, w_k)$
23: $q^{(j)} \leftarrow [\hat{y}^{(s)}_k, q^{(j)}]$
24: **end if**
25: **for** $r = 1$ to R **do**
26: $p(y^{(r)}_{t+h}|q^{(j)}) \leftarrow$ DIR$(X, O[, 1], q^{(j)}))$
27: $w^{(r)}_{t+h} \leftarrow w^{(r)}_{t+h} + p(y^{(r)}_{t+h}|q^{(j)})$
28: **end for**
29: **end for**
30: **end for**
31: **end for**
32: **for** $h = 1$ to H **do**
33: $\hat{y}_{t+h} = \frac{\sum_{r=1}^{R} w^{(r)}_{t+h} \hat{y}^{(r)}_{t+h}}{\sum_{r=1}^{R} w^{(r)}_{t+h}}$
34: **end for**

Algorithm 2 DIR

Require: Training set (input matrix X, output vector O), query point q
Ensure: posterior density $p(o|q)$, $E[o|q]$

IV. Experiments

The performance of the SMC method was assessed on three benchmarks and compared to six alternative strategies.

The three benchmarks are: the NN5 dataset, the NN3 dataset and a set of 12 NAR time series (Table I).

The NN5 dataset contains the 111 time series of the NN5 Competition (complete dataset) [12]. These time series are all the same length and contain the daily retirement amounts from independent cash machines at different, randomly selected locations across England. They show different patterns of single or multiple overlying seasonality, including day of the week effects, week or month in the year effects, and calendar effects. In our forecasting experiments, we adopt five prediction windows with the horizons $H = 50, 70, 90, 100, 200$.

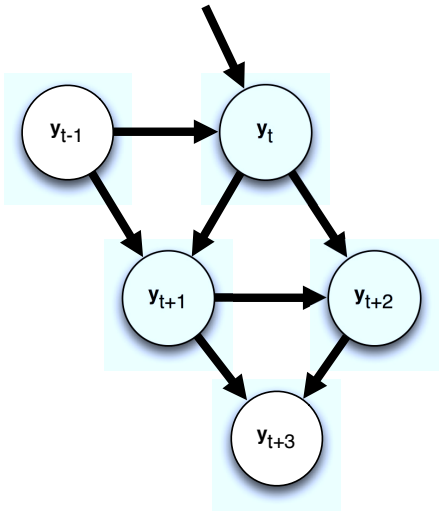

Fig. 1. Graphical model of the dependencies in a NAR(2) stochastic process

The NN3 Dataset [13] is made of 111 monthly time series starting at January, with a variable number of points (from 50 to 126). All series are drawn from homogeneous population of empirical business time series. For each time series, the competition required to forecast the values of the next $H = 18$ months based on the given historical data points. Here we consider also the horizon $H = 10$.

The third benchmark consists of a set of 1080 series obtained by simulating 90 times (different random seeds and increasing noise variances) the 12 series in Table I.

We compared the SMC algorithm to an Iterated (IT) algorithm, a Direct (DIR) algorithm, an Averaged (AVG) version of the Direct Algorithm, a LL-MIMO algorithm [2], a linear AR and a Random Walk estimator. All the forecasters uses the same embedding order $p \leq 12$ which is calculated for each series by considering the highest (and smaller than 12) significant delay in the partial correlation function. The SMC, IT, DIR and AVG algorithms rely on a locally constant estimator with an adaptive number of neighbors ranging between 3 and 15 and selected on the basis of the PRESS leave-one-statistic [14]. Though the estimation could have been performed also with other algorithms (e.g. neural networks) we adopted a local learning estimator since in litterature this approach is known to be extremely effective for short and long term forecasting [15], [16]. The MIMO algorithm implements the multiple output algorithm proposed in [2] with a number of neighbors ranging between 3 and 15. The AVG algorithm returns the average of $R = 100$ DIR estimations obtained by resampling each time two thirds of the training set. The same resampling strategy and the same number R of repetitions (line 5 of Algorithm 1) is used by the SMC algorithm in the Direct phase. The use of the same resampling strategy allows a paired assessment of the benefit deriving from the adoption of the structured prediction procedure with respect to a simple averaging approach. The

AR forecast is implemented by the R `forecast` package[17].

The forecasting accuracy results are reported in Table II in terms of average Symmetric Mean Absolute Percentage of Error (SMAPE) defined as

$$\texttt{SMAPE} = \frac{1}{H}\sum_{h=1}^{H}\frac{|\hat{y} - y_h|}{(\hat{y}_h + y_h)/2} \times 100$$

where y_h is the target output and $\hat{y}$ is the prediction. The reported values are obtained by averaging over 5 different starting points for the NN5 and the NAR series and over 3 starting points for the NN3 series. The bold notation is used to denote an average SMAPE significantly different from the SMAPE of SMC according to a paired t-test (pv < 0.05). Table III reports a Win/Losses count of the number of times that a specific technique returns a SMAPE superior (SMC wins) or inferior (SMC loses).

Some considerations can be made on the basis of the results:

- the SMC strategy is competitive with state-of-the-art approaches,
- the effectiveness of the method is not simply due to its averaging nature as shown by the comparison with the AVG approach. It appears indeed that SMC has all the times (and 7 times significantly) better average SMAPE than AVG,
- the improvement of SMC with respect the iterated strategy becomes more evident as the forecasting horizon increases,
- SMC is also consistently significantly better than the Direct method and the MIMO approach.
- In the NN3 competition benchmark the linear AR outperforms, yet not significantly, SMC. On this matter, we should however remark that neither model nor feature selection is performed in our experiments and that it is known that conventional forecasting approaches performed very well in NN3.

V. Conclusion

The comparative analysis of Iterated and Direct strategies is a hot topic in computational intelligence as well as in related domains [18], [19]. This paper advocates that these two methods can be properly integrated for the sake of accuracy and robustness. If on one side Direct strategies are able to provide good approximation of the marginal distribution of future values of the time series, on the other side Iterated methods are more effective in modeling the conditional dependency between forecasts. The Monte Carlo strategy we presented aims to preserve the best of both techniques by having recourse to an importance sampling paradigm. At the same time by introducing a resampling aspect, it is able to deal with noisy configurations. Future work will focus on related domains where the introduction of structural dependency can represent an effective inductive bias in the learning process, like spatio-temporal and vector autoregressive forecasting.

Acknowledgment

This work was supported by grants from the Communauté Française de Belgique - Actions de Recherche Concertées (ARC) and INNOVIRIS (Brussels Region).

References

[1] A. Sorjamaa, J. Hao, N. Reyhani, Y. Ji, and A. Lendasse, "Methodology for long-term prediction of time series," *Neurocompuing*, vol. 70, no. 16-18, pp. 2861–2869, 2007.

[2] G. Bontempi and S. Ben Taieb, "Conditionally dependent strategies for multiple-step-ahead prediction in local learning," *International Journal of Forecasting*, 2011.

[3] S. Ben Taieb, G. Bontempi, A. Atiya, and A. Sorjamaa, "A review and comparison of strategies for multi-step ahead time series forecasting based on the nn5 forecasting competition," *Expert Systems with Applications*, 2012.

[4] R. Caruana, "Multitask learning," *Mach. Learn.*, vol. 28, pp. 41–75, Jul. 1997.

[5] G. H. Bakir, T. Hofmann, B. Schölkopf, A. J. Smola, B. Taskar, and S. V. N. Vishwanathan, Eds., *Predicting Structured Data*, ser. Neural Information Processing. The MIT Press, 2007.

[6] D. M. Kline, *Methods for Multi-Step Time Series Forecasting Neural Networks*. IGI Global, 2004.

[7] D. Koller and N. Friedman, *Probabilistic graphical models*. The MIT Press, 2009.

[8] A. Doucet and N. D. Freitas, *Sequential Monte Carlo Methods in Practice*. Springer, 2001.

[9] J. Weston, B. Schölkopf, O. Bousquet, T. Mann, and W. S. Noble, *Joint Kernel Maps*. The MIT Press, 2007.

[10] S. Ben Taieb, A. Sorjamaa, and G. Bontempi, "Multiple-output modeling for multi-step-ahead time series forecasting," *Neurocomputing*, vol. 73, no. 10, pp. 1950–1957, 2010.

[11] C. P. Robert and G. Casella, *Monte Carlo Statistical Methods*. Springer-Verlag, Aug. 1999.

[12] R. R. Andrawis, A. F. Atiya, and H. El-Shishiny, "Forecast combinations of computational intelligence and linear models for the NN5 time series forecasting competition," *International Journal of Forecasting*, Jan. 2011.

[13] S. F. Crone, M. Hibon, and K. Nikolopoulos, "Advances in forecasting with neural networks? empirical evidence from the nn3 competition on time series prediction," *International Journal of Forecasting*, vol. 27, 2011.

[14] G. Bontempi, M. Birattari, and H. Bersini, "A model selection approach for local learning," *Artificial Intelligence Communications*, vol. 121, no. 1, 2000.

[15] T. Sauer, "Time series prediction by using delay coordinate embedding," in *Time Series Prediction: forecasting the future and understanding the past*, A. S. Weigend and N. A. Gershenfeld, Eds. Harlow, UK: Addison Wesley, 1994, pp. 175–193.

[16] G. Bontempi, M. Birattari, and H. Bersini, "Local learning for iterated time-series prediction," in *Machine Learning: Proceedings of the Sixteenth International Conference*, I. Bratko and S. Dzeroski, Eds. San Francisco, CA: Morgan Kaufmann Publishers, 1999, pp. 32–38.

[17] R. Hyndman, *forecast: Forecasting functions for time series and linear models*, 2013, R package version 4.8. [Online]. Available: http://CRAN.R-project.org/package=forecast

[18] M. H. Pesaran, A. Pick, and A. Timmerman, "Variable selection, estimation and inference for multi-period forecasting problems," *Journal of Econometrics*, vol. 164, no. 1, pp. 173–187, 2011.

[19] M. Marcellino, J. H. Stock, and M. W. Watson, "A comparison of direct and iterated multistep ar methods for forecasting macroeconomic time series," *Journal of Econometrics*, vol. 135, no. 1??2, pp. 499 – 526, 2006.

$$y_{t+1} = -0.4\frac{(3-y_t^2)}{(1+y_t^2)} + 0.6\frac{3-(y_{t-1}-0.5)^3}{1+(y_{t-1}-0.5)^4} + w_{t+1}$$

$$y_{t+1} = (0.4 - 2\exp(-50y_{t-5}^2))y_{t-5} + (0.5 - 0.5\exp(-50y_{t-9}^2))y_{t-9} + w_{t+1}$$

$$y_{t+1} = (0.4 - 2\cos(40y_{t-5})\exp(-30y_{t-5}^2))y_{t-5} + (0.5 - 0.5\exp(-50y_{t-9}^2))y_{t-9} + w_{t+1}$$

$$y_{t+1} = 2\exp(-0.1y_t^2)y_t - \exp(-0.1y_{t-1}^2)y_{t-1} + w_{t+1}$$

$$y_{t+1} = -2y_t I(y_t < 0) + 0.4y_t I(y_t < 0) + w_{t+1}$$

$$y_{t+1} = 0.8\log(1+3y_t^2) - 0.6\log(1+3y_{t-2}^2) + w_{t+1}$$

$$y_{t+1} = 1.5\sin(\pi/2y_{t-1}) - \sin(\pi/2y_{t-2}) + w_{t+1}$$

$$y_{t+1} = (0.5 - 1.1\exp(-50y_t^2))y_t + (0.3 - 0.5\exp(-50y_{t-2}^2))y_{t-2} + w_{t+1}$$

$$y_{t+1} = 0.3y_t + 0.6y_{t-1} + \frac{(0.1 - 0.9y_t + 0.8y_{t-1})}{(1+\exp(-10y_t))} + w_{t+1}$$

$$y_{t+1} = sign(y_t) + w_{t+1}$$

$$y_{t+1} = 0.8y_t - \frac{0.8y_t}{(1+\exp(-10y_t))} + w_{t+1}$$

$$y_{t+1} = 0.3y_t + 0.6y_{t-1} + \frac{(0.1 - 0.9y_t + 0.8y_{t-1})}{(1+\exp(-10y_t))} + w_{t+1}$$

TABLE I. NAR TIMES SERIES

Dataset	No. series	SMC	IT	DIR	AVG	MIMO	AR	RW
NN5 ($H = 50$)	111	4.24	4.27	**4.38**	**4.29**	**4.45**	**5.98**	**10.98**
NN5 ($H = 70$)	111	4.52	4.67	**4.60**	**4.55**	**4.67**	**6.38**	**8.67**
NN5 ($H = 90$)	111	6.04	**7.02**	6.07	**6.09**	**6.19**	**11.83**	**13.22**
NN5 ($H = 100$)	111	5.91	**6.1**	**5.96**	**5.93**	**6.08**	**8.53**	**12.83**
NN5 ($H = 200$)	111	13.31	**14.25**	**13.6**	**13.4**	**13.6**	**19.22**	**20.68**
NN3 ($H = 10$)	111	3.18	3.28	**3.25**	3.20	**3.32**	3.00	**4.39**
NN3 ($H = 18$)	111	8.70	8.87	**8.89**	**8.72**	**8.94**	8.23	**11.46**
NAR ($H = 10$)	1080	1.46	**1.54**	**1.49**	**1.46**	**1.48**	**1.65**	**2.08**

TABLE II. AVERAGE SMAPE: THE BOLD NOTATION IS USED TO DENOTE THAT A TECHNIQUE HAS AN AVERAGE SMAPE SIGNIFICANTLY DIFFERENT (PAIRED T-TEST P-VALUE < 0.05) FROM THE ONE OF SMC.

Dataset	IT	DIR	AVG	MIMO	AR	RW
NN5 ($H = 50$)	59/52	98/13	95/16	92/19	107/4	111/0
NN5 ($H = 70$)	54/57	78/33	91/20	84/27	105/6	108/3
NN5 ($H = 90$)	79/32	64/47	82/29	91/20	109/2	111/0
NN5 ($H = 100$)	67/44	79/32	83/28	90/21	107/4	108/3
NN5 ($H = 200$)	67/44	99/12	90/21	85/26	107/4	107/4
NN3 ($H = 10$)	62/49	72/39	61/50	54/57	43/68	65/46
NN3 ($H = 18$)	65/46	72/39	65/46	67/44	52/59	75/36
NAR ($H = 10$)	658/422	680/400	646/434	583/497	583/497	731/349

TABLE III. NUMBER OF SMC WIN-LOSSES: THE NOTATION W/L MEAN THAT SMC HAS A BETTER SMAPE THAN THE CONSIDERED TECHNIQUE W OUT OF (W+L) TIMES.