

# The Clean Thesis Style

---

Ricardo Langner

*August 26, 2015*  
Version: My First Draft



Clean Thesis Style University

# Clean**Thesis**

Department of Clean Thesis Style

Institut for Clean Thesis Dev

Clean Thesis Group (CTG)

Documentation

## The Clean Thesis Style

Ricardo Langner

- |                    |  |
|--------------------|--|
| <i>1. Reviewer</i> | <b>Jane Doe</b><br>Department of Clean Thesis Style<br>Clean Thesis Style University |
| <i>2. Reviewer</i> | <b>John Doe</b><br>Department of Clean Thesis Style<br>Clean Thesis Style University |
| <i>Supervisors</i> | Jane Doe and John Smith  |

August 26, 2015

**Ricardo Langner**

*The Clean Thesis Style*

Documentation, August 26, 2015

Reviewers: Jane Doe and John Doe

Supervisors: Jane Doe and John Smith

**Clean Thesis Style University**

*Clean Thesis Group (CTG)*

Institut for Clean Thesis Dev

Department of Clean Thesis Style

Street address

Postal Code and City

# Contents

<b>1 Multiple time series forecasting using ensembles of quasi-randomized functional link neural networks</b>	<b>1</b>
1.1 Introduction . . . . .	1
1.2 Description of the base models . . . . .	2
1.3 Ensembles of RVFL . . . . .	5
1.3.1 Bagging . . . . .	5
1.3.2 Stacking . . . . .	7
1.4 Numerical examples . . . . .	9
1.4.1 Descriptive statistics . . . . .	9
1.4.2 Numerical example for bagging . . . . .	12
1.4.3 Numerical example for stacking . . . . .	12
<b>Bibliography</b>	<b>15</b>



# Multiple time series forecasting using ensembles of quasi-randomized functional link neural networks

## 1.1 Introduction

The goal of ensemble learning is to combine two or more individual statistical/machine learning models - the base models or base learners - into one, in order to obtain an ensemble model, with an improved out-of-sample error over the base models.

Ensemble learning is widely used in the winning solutions of machine learning competitions. It allows to achieve very good performances indeed, at the expense of being relatively less interpretable than the base learners.

As a consequence, choosing to use ensemble models or a base model for solving a given problem, could sometimes be seen as finding a trade-off between the desire for interpretability and the desire for an highly increased performance. That said, some techniques such as variable importance for ensemble of trees, can give a sense of which predictor contributes the most to the performance of the model.

In this paper, we apply two popular ensemble learning methods to multiple time series forecasting: Bootstrap aggregating (cite Efron), known as bagging and stacked generalization(cite Wolpert), known as stacking. The base learners that we use, are the quasi-random vector functional link neural networks introduced in Moudiki et al. (2017).

In the next section, we give an overview of the base learners. Then, we describe how we use these models as the basic components of our bagged/stacked ensembles. To finish, we present some numerical examples of use of these new models for forecasting multiple time series.

## 1.2 Description of the base models

The model described in Moudiki et al. (2017) is a single layer feed forward neural networks (SLFN). We have an output variable  $y \in \mathbb{R}^n$ , which has to be explained by a set of observed predictors  $X^{(j)} \in \mathbb{R}^n, j \in \{1, \dots, p\}$ . The RVFL networks that we use to explain  $y$  is described for  $i \in \{1, \dots, n\}$  as:

$$y_i = \beta_0 + \sum_{j=1}^p \beta_j X_i^{(j)} + \sum_{l=1}^L \gamma_l g \left( \sum_{j=1}^p W^{(j,l)} X_i^{(j)} \right) + \epsilon_i$$

Where  $g$  is the *activation function*,  $L$  the number of nodes in the hidden layer,  $W^{(j,l)}$  are elements of the hidden layer, and the parameters  $\beta_j$  and  $\gamma_l$  are to be learned from the observed data  $X^{(j)}, j \in \{1, \dots, p\}$ . The  $\epsilon_i$ 's are the residual differences between the output variable values and the RVFL model.

This type of model can be seen as a one explaining  $y_i$ , by finding a compromise between linear and potentially non-linear effects of the original predictors  $X^{(j)}$  and transformed predictors

$$\Phi(\mathbf{X})^{(l)} = g \left( \sum_{j=1}^p W^{(j,l)} X_i^{(j)} \right)$$

$\{1, \dots, L\}$  on the response. In this paper, we use the Rectified Linear Units activation function, known as ReLU

$$g : x \mapsto \max(x, 0)$$

But other choices of activation functions, such as the sigmoid function or the hyperbolic tangent (or others) can be envisaged.

The elements  $W^{(j,l)}$  of the hidden layer are taken from a quasi-random (deterministic) *sobol* sequence on  $[0, 1]$ , which is shifted in such a way that they belong to  $[-1, 1]$ . Solving for the optimal  $\beta_j$ 's and  $\gamma_l$ 's is done by searching these parameters, in restricted regions where we have:

$$\sum_{j=1}^p \beta_j^2 \leq u$$

and

$$\sum_{l=1}^L \gamma_l^2 \leq v$$

for  $u, v \in \mathbb{R}^*$ . That is, by applying a regularization to these unknown parameters.



Now, if we consider  $p \in \mathbb{N}^*$  time series  $(X_t^{(j)})_{t \geq 0}, j = 1, \dots, p$ , observed at  $n \in \mathbb{N}^*$  discrete dates. We are interested in obtaining simultaneous forecasts of the  $p$  time series at time  $n + h, h \in \mathbb{N}^*$ , by allowing each of the  $p$  variables to be influenced by the others. We use  $k < n$  lags of each of the observed  $p$  time series. So that, the output variables to be explained are:

$$Y^{(j)} = \left( X_n^{(j)}, \dots, X_{n-k+1}^{(j)} \right)^T \quad (1.1)$$

for  $j \in \{1, \dots, p\}$ . Where  $X_n^{(j)}$  is the most contemporaneous observed value of the  $j^{th}$  time series, and  $X_{n-k+1}^{(j)}$  was observed  $k$  dates earlier in time for  $(X_t^{(j)})_{t \geq 0}$ . These output variables are stored in a matrix:

$$\mathbf{Y} \in \mathbb{R}^{(n-k) \times p}$$

and the predictors are stored in a matrix:

$$\mathbf{X} \in \mathbb{R}^{(n-k) \times (k \times p)}$$

where  $\mathbf{X}$  consists in  $p$  blocks of  $k$  lags, for each one of the observed  $p$  time series.

An additional layer of transformed predictors is added to  $\mathbf{X}$ , in order to capture the potentially non-linear interactions between the predictors and the output variable. Adding the transformed predictors to the original ones, leads to a new matrix of predictors with dimensions  $(n - k) \times (k \times p + L)$ , where  $L$  is the number of nodes in the hidden layer.

For example, we have  $p = 2$  time series  $(X_{t_1}^{(1)}, \dots, X_{t_5}^{(1)})$  and  $(X_{t_1}^{(2)}, \dots, X_{t_5}^{(2)})$  observed at  $n = 5$  dates  $t_1 < \dots < t_5$ , with  $k = 2$  lags, and  $L = 3$  nodes in the hidden layer. In this case, the response variables are stored in:

$$\mathbf{Y} = \begin{pmatrix} X_{t_5}^{(1)} & X_{t_5}^{(2)} \\ X_{t_4}^{(1)} & X_{t_4}^{(2)} \\ X_{t_3}^{(1)} & X_{t_3}^{(2)} \end{pmatrix}$$

The predictors are stored in:

$$\mathbf{X} = \begin{pmatrix} X_{t_4}^{(1)} & X_{t_3}^{(1)} & X_{t_4}^{(2)} & X_{t_3}^{(2)} \\ X_{t_3}^{(1)} & X_{t_2}^{(1)} & X_{t_3}^{(2)} & X_{t_2}^{(2)} \\ X_{t_2}^{(1)} & X_{t_1}^{(1)} & X_{t_2}^{(2)} & X_{t_1}^{(2)} \end{pmatrix}$$

And the coefficients in the hidden layer are:

$$\mathbf{W} = \begin{pmatrix} W^{(1,1)} & W^{(1,2)} & W^{(1,3)} \\ W^{(2,1)} & W^{(2,2)} & W^{(2,3)} \\ W^{(3,1)} & W^{(3,2)} & W^{(3,3)} \\ W^{(4,1)} & W^{(4,2)} & W^{(4,3)} \end{pmatrix}$$

We let  $y$  be the  $j_0^{th}$  column (out of  $p$ ) of the response matrix  $\mathbf{Y}$ , and  $\Phi(\mathbf{X})$  be the matrix of transformed predictors obtained from  $\mathbf{X}$  by the hidden layer. We also denote the set of regression parameters associated with this  $j_0^{th}$  time series, as:

$$\beta_m^{(j_0)} =: \beta_m$$

and

$$\gamma_l^{(j_0)} =: \gamma_l$$

for  $m \in \{1, \dots, k\}; l \in \{1, \dots, L\}$ . Solving for the regression parameters for the  $j_0^{th}$  time series, under the constraints

$$\sum_{m=1}^{k \times p} \beta_m^2 \leq u$$

and

$$\sum_{l=1}^L \gamma_l^2 \leq v$$

for  $u, v \in \mathbb{R}^*$ , leads to minimizing a penalized residual sum of squares:

$$\mathcal{L}(\mathbf{X}; \beta, \gamma) = (y - \mathbf{X}\beta - \Phi(\mathbf{X})\gamma)^T (y - \mathbf{X}\beta - \Phi(\mathbf{X})\gamma) + \lambda_1 \beta^T \beta + \lambda_2 \gamma^T \gamma$$

where  $\lambda_1$  and  $\lambda_2$  are Lagrange multipliers. Now, by denoting:

$$A = \begin{pmatrix} \mathbf{X}^T \mathbf{X} + \lambda_1 I_{k \times p} & \mathbf{X}^T \Phi(\mathbf{X}) \\ \Phi(\mathbf{X})^T \mathbf{X} & \Phi(\mathbf{X})^T \Phi(\mathbf{X}) + \lambda_2 I_L \end{pmatrix} =: \begin{pmatrix} B & C^T \\ C & D \end{pmatrix}$$

and  $S = D - CB^+C^T$ . And:

$$A^+ = \begin{pmatrix} B^+ + B^+C^T S^+ C B^+ & -B^+C^T S^+ \\ -S^+ C B^+ & S^+ \end{pmatrix} =: \begin{pmatrix} A_1^+ & A_2^+ \\ A_3^+ & A_4^+ \end{pmatrix}$$

where  $S^+$  and  $B^+$  are the Moore-Penrose pseudo-inverse of matrices  $S$  and  $B$ . The whole set of parameters, for all the  $p$  observed time series is given by:

$$\begin{pmatrix} \hat{\beta} \\ \hat{\gamma} \end{pmatrix} := \begin{pmatrix} A_1^+ & A_2^+ \\ A_3^+ & A_4^+ \end{pmatrix} \begin{pmatrix} \mathbf{X}^T \mathbf{Y} \\ \Phi(\mathbf{X})^T \mathbf{Y} \end{pmatrix}$$

## 1.3 Ensembles of RVFL

As mentioned in the introduction, we will use bagging and stacking to construct ensembles of RVFL from the previous section 1.2. Both techniques will be described in the next sections 1.3.1 and 1.3.2. Firstly in a general context, and secondly as how they are applied to our RVFL models.

### 1.3.1 Bagging

The Bootstrap (cite Efron) uses multiple random replications of a given data set, to obtain standard errors of model parameters, for example. In the context of ensemble learning, these replications of the original data set are used to produce various predictions of the base models, which are then aggregated - in the case of regression problems, they are averaged, and in the case of classification problems, majority vote could be used - to obtain a single prediction with less out-of-sample variance. This procedure is called bootstrap aggregating or *bagging*.

In order to illustrate the benefits of the bagging, we consider  $n$  different base learners, with out-of-sample prediction errors equal to  $\epsilon_1, \dots, \epsilon_n$ , and assume that the distribution of these errors are centered around 0:

$$\mathbb{E}[\epsilon_i] = 0$$

for  $i = 1, \dots, n$ . In addition, we have:

$$\mathbb{E}[\epsilon_i \epsilon_j] = \gamma$$

for  $i \neq j$ ; that is, the covariance between the errors is equal to  $\gamma$ . And:

$$Var(\epsilon_i) = \sigma^2$$

for  $i = 1, \dots, n$ . The out-of-sample mean squared error (MSE) of the aggregated (averaged) model including these  $n$  base models is equal to:

$$\begin{aligned}
MSE &= \mathbb{E} \left[ \left( \frac{1}{n} \sum_{i=1}^n \epsilon_i \right)^2 \right] = \frac{1}{n^2} \mathbb{E} \left[ \left( \sum_{i=1}^n \epsilon_i \right)^2 \right] \\
&= \frac{1}{n^2} \mathbb{E} \left[ \sum_{i=1}^n \epsilon_i^2 + 2 \sum_{i < j} \epsilon_i \epsilon_j \right] \\
&= \frac{1}{n^2} \left( \sum_{i=1}^n \mathbb{E} [\epsilon_i^2] + 2 \sum_{i < j} \mathbb{E} [\epsilon_i \epsilon_j] \right) \\
&= \frac{1}{n^2} \left( n\sigma^2 + 2 \frac{n(n-1)}{2} \gamma \right) \\
&= \frac{\sigma^2}{n} + \frac{(n-1)}{n} \gamma
\end{aligned}$$

From this expression of the  $MSE$ , we can observe that if  $n$  is high ( $n \rightarrow \infty$ ), that is, if there are several base learners in the ensemble, and the out-of-sample expected error is low, the out-of-sample MSE of the ensemble prediction will decrease, and can eventually be reduced to  $\gamma$ . If, in addition, the model predictions are perfectly uncorrelated, i.e  $\gamma = 0$ , then the out-of-sample MSE is further decreased.

Having a low value for  $\gamma$ , along with a low out-of-sample expected error, helps in achieving a lower out-of-sample prediction error for the ensemble model. Decorrelation in ensemble learning is hence important, and consists in increasing the diversity in the base learners, in order to obtain a low value for  $\gamma$ .

This decorrelation among the base learners is achieved for example in Random Forest models (cite Breiman, L. (2001)), by growing each tree in the forest, with only a subset of the predictors available. Also, an example of use of decorrelation, specifically for ensembles of neural networks is demonstrated in (cite Rosen (1996)). The author presents three approaches for achieving *disagreement* between the networks: one in which they are trained independently and aggregated with the hope that their predictions are somewhat different; a second one, in which different activation functions or architecture (typically, more or less hidden layers or nodes in the hidden layers etc.) for each base learner. A third approach consists in training the individual networks on different subsamples of the original training set. The case of decorrelation learning for RVFL is treated in (cite Alamdoosh et Wang (2014))

In both papers, (cite Rosen (1996)) and (cite Alamdoosh et Wang (2014)), the procedure implemented in order to obtain a decorrelation of the base learners is

denoted as Negative Correlation Learning (NCL). The general idea is to minimize the penalized Root Mean Squared Error:

$$\sum_{i=1}^n \left[ (y_i - f_k(\mathbf{x}_i))^2 + \lambda p_k(\mathbf{x}_i) \right]$$

where  $\mathbf{x}_i \in \mathbb{R}^p$  is the  $i^{th}$  observation of the training data set, with  $p$  features.  $f_k$  is a base learner with  $k \in \{1 \dots B\}$ , and  $B$  is the number of bootstrap resamples.  $p_k$  is a penalty term decorrelating the current network's error with the errors of the networks previously trained.  $\lambda$  is a Lagrange multiplier, a regularization parameter, preventing the correlation between the successive base learners in-sample errors from being high. For example,  $p_k$  could be defined as:

$$p_k(\mathbf{x}_i) = (y_i - f_k(\mathbf{x}_i)) \sum_{j < k} (y_i - f_j(\mathbf{x}_i))$$

In this chapter, we use the third approach described in (cite Rosen (1996)) (although many other approaches can be envisaged). Multiple training data sets are constructed by randomly picking a fraction of the covariates, and a subset of the lines. Taking a subset of the lines is done by respecting the serial dependence of the series and taking consecutive observations (that is, without skipping any observation) for each bootstrap resample. The predictions associated to each training set resample are then averaged, in order to obtain a single prediction, with a confidence interval around it. Some numerical examples on the application of bagging to RVFL base learners can be found in section 1.4

### 1.3.2 Stacking

Stacked generalization or stacking, was introduced (cite Wolpert (1992)). It is also presented in (Breiman (2001)) for regression models. The idea behind this procedure is to construct new predictors for the training data set, by using diverse predictions of multiple models, which are combined by a meta-learner model. A simple example of 2-fold stacking applied to regression is described hereafter. It is possible to imagine examples with more stacked layers:

- Divide the original training data set  $\mathbf{X} \in \mathbb{R}^{n \times p}$ , into two parts: part1 and part2, each with  $n/2$  rows
- Train a base learner model on part1 and obtain  $n/2$  predictions on part2.
- Train the same base learner model on part2 and obtain  $n/2$  predictions on part1.

- Train a meta-learner model on the new data set `part3`, consisting in the the original predictors, plus the predictors constructed by applying the base learner model on `part1` and `part2`.

Typically, in this procedure, various types of base learners are used, to increase the diversity of new predictors. Since we are using time series data here, which inherently exhibit a serial dependence between the observations, we have to adapt the procedure described previously. In the case of a 2-fold stacking again, we divide the original training data set  $\mathbf{X} \in \mathbb{R}^{n \times p}$  (with  $p$  time series observed at  $n$  dates) into two parts: `part1` and `part2`; each consisting of  $n/2$  rows. `part1` contains the most ancient observations and `part2`, the most recent observations of the time series contained in  $\mathbf{X}$ . Also, we use multiple resamples of the base learner, as follows:

- Train  $B$  resampled RVFL models on `part1`, using a random subset of `part1` in each resampling iteration. Obtain  $B$  sets of predictions ( $B \times p$  new predictors) with these models, over the whole horizon of `part2`.
- Create a new enriched data set `part3`, containing the observed time series from `part2`, and the  $B \times p$  additional series predicted from `part1` on `part2`.
- Use a meta-learner to obtain predictions on the new, enriched data set `part3`.

For the latter point, we consider a few meta-learner models that have been used in the past in the literature (cite Breiman's Stacked Regressions). In general, the idea is that a relatively simple meta-learner will achieve a good performance. We use the following models for this purpose:

- **Linear least squares regression** model
- Linear least squares regression model **with positive coefficients** (cite Lawson CL, Hanson RJ (1974)) and Lawson CL, Hanson RJ (1995), as used in (cite Breiman (2001))
- **Ridge regression** with the use of singular values of . . . (cite Hoerl et Kennard 1970)
- **Elastic net regression** with the use of coordinate descent (cite Friedman, J., Hastie, T. and Tibshirani, R. (2008))

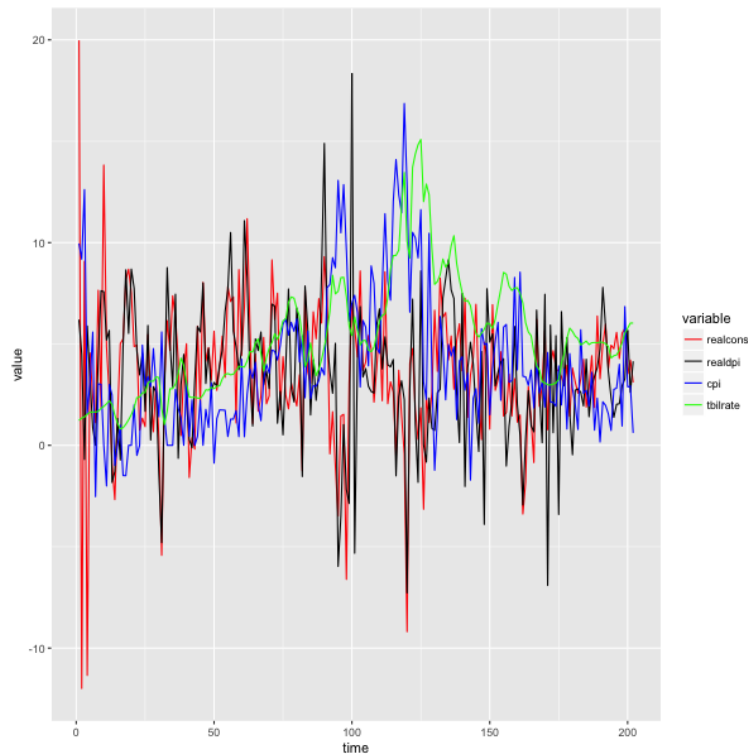
These selected meta-learners are trained on `part3`, with the most contemporaneous observations of the series as responses, and their respective lags as predictors. Some numerical examples on the application of stacking to RVFL base learners can be found in section 1.4

## 1.4 Numerical examples

For the numerical examples, we obtain forecast of the Treasury Bill rates among other macroeconomic variables, in a *data-rich* environment. We use data from (cite Greene (20something)). Four of the time series are considered: the Treasury Bill rates, the real consumption, the consumer price index, and the real expenditure. They are all observed on a quaterly basis. We annualize the last three time series, but keep the Treasury Bill rates unchanged. So that, the four time series are nearly on the same scale.

### 1.4.1 Descriptive statistics

Figure 1.1 presents the resulting time series' data obtained after the few transformations described in the previous paragraph. Table 1.1 contains a summary of these data, where we can see that the four time series are now nearly expressed in the same scale.



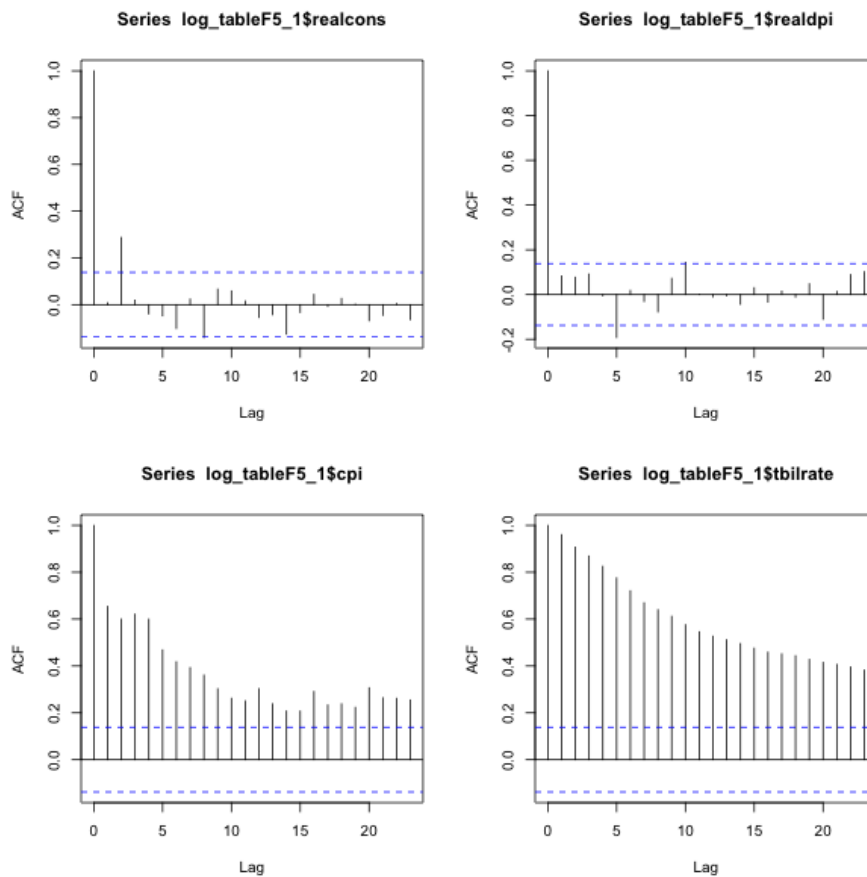
**Fig. 1.1:** Treasury Bills rates and transformed Real consumption, Real disposable income, and Inflation data from (cite Greene et. al (2007)) in a *data-rich* environment

**Tab. 1.1:** Summary statistics for the four transformed time series: Treasury Bills and transformed Real consumption, Real disposable income, and Inflation data from (cite Greene et. al (2007))

Series	Min	1st Qrt	Median	Mean	3rd Qrt	Max	Std. Dev.
Real consumption	-11.990	1.814	3.637	<b>3.513</b>	5.414	19.978	<b>3.546</b>
Real disposable income	-7.275	1.613	3.390	<b>3.423</b>	5.518	18.358	<b>3.486</b>
Inflation	-2.530	1.755	3.135	<b>3.936</b>	5.593	16.864	<b>3.407</b>
Treasury Bills rates	0.810	3.138	5.050	<b>5.270</b>	6.715	15.090	<b>2.829</b>

**Tab. 1.2:** Box-Pierce test for the independence in the time series observations

Series	X-squared	p-value
Real consumption	0.0133	<b>0.9082</b>
Real disposable income	1.3912	<b>0.2382</b>
Inflation	86.428	2.2e-16
Treasury Bills rates	186.35	2.2e-16



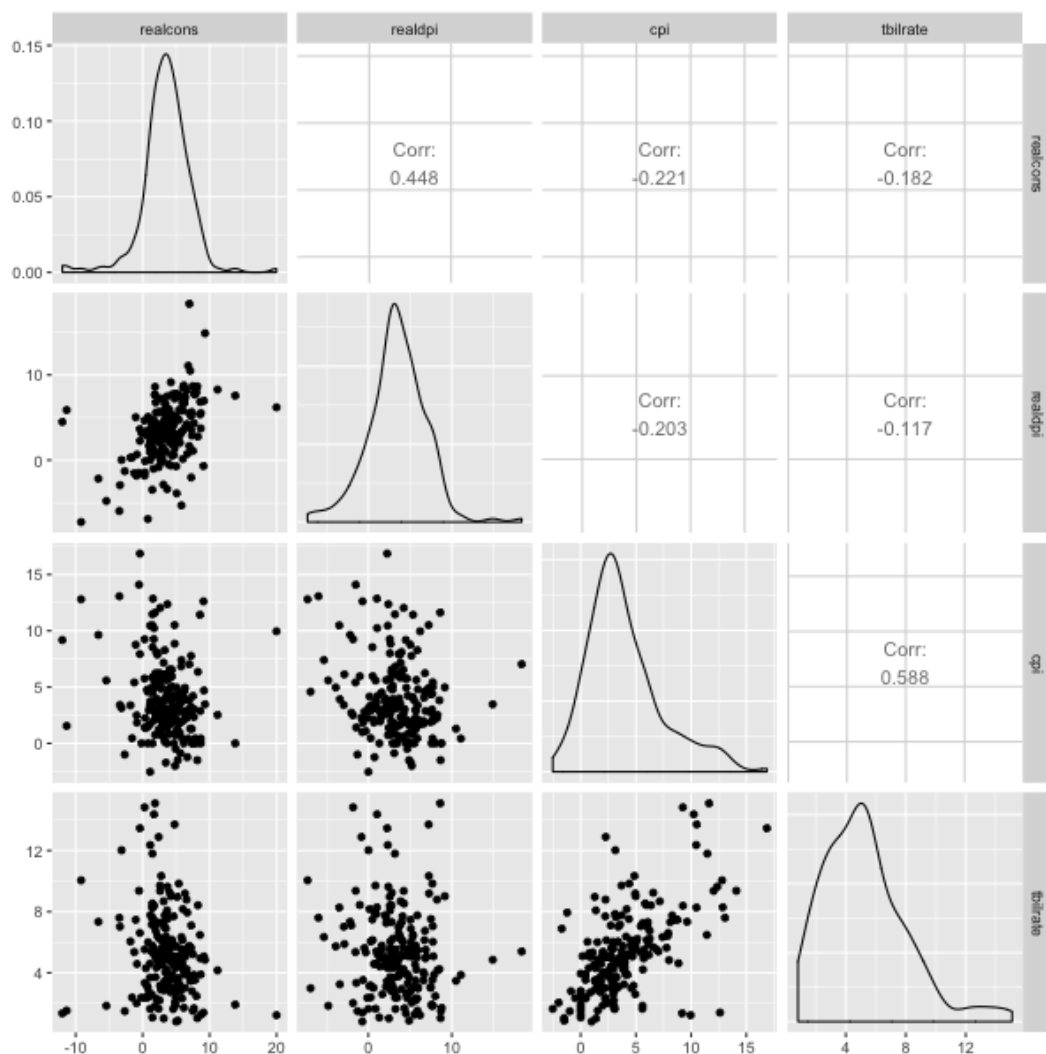
**Fig. 1.2:** Autocorrelation function for the four times series: Treasury Bills rates, Real consumption, Real disposable income, Inflation



The Box-Pierce tests results presented in table 1.2, and the autocorrelation functions presented in 1.2, show that the two time series for Real consumption and Real disposable income could be considered as stationary after the transformations. Whereas for the Treasury Bills rates and Inflation, there is still a non-negligible autocorrelation within the series.

Another interesting information is given by figure 1.3. We observe that the distribution of the data for inflation and Treasury Bills rates is skewed, when compared to the data for Real consumption and Real disposable income.

Also, based on the correlations displayed in figure 1.3, the Real consumption is globally decreasing when the Treasury Bills rates and the inflation are increasing. But when the Real disposable income increases, the Real consumption increases as well.



**Fig. 1.3:** Correlation plot for the four times series: Treasury Bills rates, Real consumption, Real disposable income, Inflation

In sections 1.4.2 and 1.4.3, we apply the bagging and stacking procedures described in sections 1.3.1 and 1.3.2 to the dataset we have just described. The dataset is divided into two parts:

- A **training/testing set**: used for applying a rolling forecasting cross-validation method (see [Ber+15]). At the beginning of the procedure, the training set contains 12 points (4 years), and the test set contains 4 points (1 year). The training set is then advanced of one point forward (rolled), and the procedure is repeated until no more data for the training set are found.
- A **validation set**: the validation set is used to verify that the parameters produced by the cross-validation from the previous point lead to *good* forecast on unseen data. It contains four points (1-year ahead forecast) for each time series.

We use the Root Mean Squared Error (RMSE) as a measure of the out-of-sample error.

## 1.4.2 Numerical example for bagging

## 1.4.3 Numerical example for stacking





# Bibliography

- [Ber+15] Christoph Bergmeir, Rob J Hyndman, Bonsoo Koo, et al. „A Note on the Validity of Cross-Validation for Evaluating Time Series Prediction“. In: *Monash University, Department of Econometrics and Business Statistics, Tech. Rep.* (2015) (cit. on p. 12).
- [Bon+15] F Bonnin, F Combes, F Planchet, and M Tammar. „Un modèle de projection pour des contrats de retraite dans le cadre de l'ORSA“. In: *Bulletin Français d'Actuariat* 14 (2015), pp. 107–129.
- [BT97] Phelim P Boyle and Ken Seng Tan. „Quasi-Monte Carlo methods“. In: *International AFIR Colloquium Proceedings, Australia*. Vol. 1. 1997, pp. 1–24.
- [Car98] Rich Caruana. „Multitask learning“. In: *Learning to learn*. Springer, 1998, pp. 95–133.
- [Cha+92] Kanad Chakraborty, Kishan Mehrotra, Chilukuri K Mohan, and Sanjay Ranka. „Forecasting the behavior of multivariate time series using neural networks“. In: *Neural networks* 5.6 (1992), pp. 961–970.
- [Cor09] Thomas H Cormen. *Introduction to algorithms*. MIT press, 2009.
- [DC10] Satchidananda Dehuri and Sung-Bae Cho. „A comprehensive survey on functional link neural networks and an adaptive PSO–BP learning for CFLNN“. In: *Neural Computing and Applications* 19.2 (2010), pp. 187–205.
- [DL06] Francis X Diebold and Canlin Li. „Forecasting the term structure of government bond yields“. In: *Journal of econometrics* 130.2 (2006), pp. 337–364.
- [DR13] Francis X Diebold and Glenn D Rudebusch. *Yield Curve Modeling and Forecasting: The Dynamic Nelson-Siegel Approach*. Princeton University Press, 2013.
- [DS15] Christophe Dutang and Petr Savicky. *randtoolbox: Generating and Testing Random Numbers*. R package version 1.17. 2015.
- [Ext+16] Peter Exterkate, Patrick JF Groenen, Christiaan Heij, and Dick van Dijk. „Nonlinear forecasting with many predictors using kernel ridge regression“. In: *International Journal of Forecasting* 32.3 (2016), pp. 736–753.
- [HK08] Robin Hyndman and Yeasmin Khandakar. „Automatic time series forecasting: The forecast Package for R“. In: *Journal of Statistical Software* 27.3 (2008), pp. 1–22.
- [HK70] Arthur E Hoerl and Robert W Kennard. „Ridge regression: Biased estimation for nonorthogonal problems“. In: *Technometrics* 12.1 (1970), pp. 55–67.

- [HS97] Sepp Hochreiter and Jürgen Schmidhuber. „Long short-term memory“. In: *Neural computation* 9.8 (1997), pp. 1735–1780.
- [JK08] S Joe and F Kuo. *Notes on generating Sobol sequences*. <http://web.maths.unsw.edu.au/~fkuo/sobol/joe-kuo-notes.pdf>. 2008.
- [Lüt05] Helmut Lütkepohl. *New introduction to multiple time series analysis*. Springer Science & Business Media, 2005.
- [Nie92] Harald Niederreiter. *Random number generation and quasi-Monte Carlo methods*. SIAM, 1992.
- [NS87] Charles R Nelson and Andrew F Siegel. „Parsimonious modeling of yield curves“. In: *Journal of business* (1987), pp. 473–489.
- [Pan12] Alan Pankratz. *Forecasting with dynamic regression models*. Vol. 935. John Wiley & Sons, 2012.
- [Pao+94] Yoh-Han Pao, Gwang-Hoon Park, and Dejan J Sobajic. „Learning and generalization characteristics of the random vector functional-link net“. In: *Neurocomputing* 6.2 (1994), pp. 163–180.
- [Pen55] Roger Penrose. „A generalized inverse for matrices“. In: *Mathematical proceedings of the Cambridge philosophical society*. Vol. 51. 03. Cambridge Univ Press. 1955, pp. 406–413.
- [Pfa+08] Bernhard Pfaff et al. „VAR, SVAR and SVEC models: Implementation within R package vars“. In: *Journal of Statistical Software* 27.4 (2008), pp. 1–32.
- [Ren+16] Ye Ren, PN Suganthan, N Srikanth, and Gehan Amaratunga. „Random vector functional link network for short-term electricity load demand forecasting“. In: *Information Sciences* 367 (2016), pp. 1078–1093.
- [Rum+88] DE Rumelhart, GE Hinton, and RJ Williams. „Learning internal representations by error propagation“. In: *Neurocomputing: foundations of research*. MIT Press. 1988, pp. 673–695.
- [Sch+92] Wouter F Schmidt, Martin A Kraaijveld, and Robert PW Duin. „Feedforward neural networks with random weights“. In: *Pattern Recognition, 1992. Vol. II. Conference B: Pattern Recognition Methodology and Systems, Proceedings., 11th IAPR International Conference on*. IEEE. 1992, pp. 1–4.
- [Wic16] Hadley Wickham. *ggplot2: elegant graphics for data analysis*. Springer, 2016.
- [ZS16] Le Zhang and PN Suganthan. „A comprehensive evaluation of random vector functional link networks“. In: *Information Sciences* 367 (2016), pp. 1094–1105.

## List of Figures

1.1	Treasury Bills rates and transformed Real consumption, Real disposable income, and Inflation data from (cite Greene et. al (2007)) in a <i>data-rich</i> environment . . . . .	9
1.2	Autocorrelation function for the four times series: Treasury Bills rates, Real consumption, Real disposable income, Inflation . . . . .	10
1.3	Correlation plot for the four times series: Treasury Bills rates, Real consumption, Real disposable income, Inflation . . . . .	11





## List of Tables

1.1	Summary statistics for the four transformed time series: Treasury Bills and transformed Real consumption, Real disposable income, and Inflation data from (cite Greene et. al (2007)) . . . . .	10
1.2	<b>Box-Pierce test for the independence</b> in the time series observations	10



## Colophon

This thesis was typeset with  $\text{\LaTeX}$  2<sub>ε</sub>. It uses the *Clean Thesis* style developed by Ricardo Langner. The design of the *Clean Thesis* style is inspired by user guide documents from Apple Inc.

Download the *Clean Thesis* style at <http://cleanthesis.der-ric.de/>.



# Declaration

You can put your declaration here, to declare that you have completed your work solely and only with the help of the references you mentioned.

*City, August 26, 2015*

---

Ricardo Langner

