PPGEE
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

INSTITUTO FEDERAL
Paraíba
Campus João Pessoa

INTELIGÊNCIA COMPUTACIONAL

Lista de Exercicio 1A

Thyago Borges de Carvalho

João Pessoa - PB
Março de 2021

To my Gramma Adélia.

ABSTRACT

This work seeks to optimize demand contracts for consumer units (CUs) in the Brazilian electrical group A. For this purpose, an algorithm was created to calculate the best demand for a given period of time called function DC12. In addition, in the first moment, a time series forecasting model called ARIMA was used to compare its performance with DC12-P (the result of the DC12 function prediction). The model analyzed the demand consumed in the last 12 months of each unit and ARIMA was the best method for 47.61 % of UCs. In a second step, 9 more mathematical forecasting models were used in addition to ARIMA and the result was a better performance of this method with the use of those prediction models in 81.63 % of the UCs.

Keywords: Demand Contracts; ARIMA; DC12; Power Management,Energy Efficiency.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMMARY

1 INTRODUCTION

Large companies, which have a large number of consumer units in their organization, have the need to manage all their energy requirements.

As well illustrates Albuquerque (2015):

> The annual cost of the electricity bill is an important part for most industries and needs to be analyzed carefully, as wrong or poorly reasoned choices regarding the tariff framework can have very high cost consequences. To have access to electricity, the industry needs to enter into a supply contract with the concessionaire, with a minimum term of one year, and indicate its contracted demand, which is the amount of active power needed to supply its energy demand ".

Endorsed by this vision, it is extremely important that the consumer should be able to define, in an optimized way, which is the best contracted demand for the future contract period (minimum term of one year), which minimizes the annual cost with contracted demand and, consequently, , increase efficiency in annual electricity expenses.

In addition, the increasingly accurate calculation of Contracted Demand provides everyone involved, generators, distributors and consumers greater security and accuracy on they services. Because, the more accurate the forecast of the country's energy use with its consumption and loads, the lower the risk of blackouts, promoting loss reduction and energy efficiency.

In the literature, it is possible to find several papers that apply the time series methodology, among other methodologies, to analyze the behavior of energy consumption in order to provide the best energy management and increase efficiency in the use of electricity (PRADO, 2011).

The work of Dalmas et al. (2003 textbf apud PRADO, 2011, pg.14) applied the method of Box and Jenkins to identify Integrated Auto-Regressive Forecasting Models for Seasonal Moving Averages (SARIMA), in the analysis of the behavior of the electric energy consumption of the classes of commercial and industrial consumers in the State of Paraná, verifying trend and seasonality in the observed data. Equations were obtained for the commercial and industrial consumer classes, which promoted reflexes in the containment of electric energy by these consumers. The work of Giarola and Rocha (2008 textbf apud PRADO, 2011, pg.14), in turn, shows that the SARIMA model with interventions was the one that best represented, among 15 analysis that the models for time series, the series of electricity consumption in southeastern Brazil.

Talking about it, this work describes the use of new data science technologies, which form the basis of the so-called industry 4.0, in the generation of an algorithm capable of providing the best choice of contracted demand for consumer units in group A (defined in chapter 2 ; In resume it is a Brazilian classification of energy voltage use, the group A has to set a contract demand mandatorily), considering the anticipated behavior in the next 12 months of consumption.

To this end, prediction models generated with the support of Data Science and Machine Learning tools are applied, which are able to predict with good precision the best value of contracted demand for the next current period of primary consumers. , based on your past consumption history. For the application of this study, demand information from consumer units of Paraíba waters and sewage company (CAGEPA), the company responsible for water supply and sanitation for the entire state of Paraíba, was used with the mission of meeting the environmental sanitation needs of the population, contributing to the improvement of life and health. Paraiba public health.

To exercise its obligation, CAGEPA has more than 800 Consumer Units (CU), of which 191 are Group A CUs, which had sufficient demand history for analysis, according to information from the Energy Management Management - GEGE (CAGEPA , 2020), which is responsible for the good energy practices of all consumer units that are part of CAGEPA.

1.1 Objectives

The general objective of the work is to obtain the optimized value of demand to be contracted for the next cycle of electricity contract, for consumer units of group A, through the study and application of time forecasting tools. To achieve this objective, specific objectives were defined:

- Propose a methodology capable of defining the best value for the demand contract of each of the consumer units evaluated in the study.
- Create an algorithm capable of defining which is the best demand in a predefined period of time.
- Apply and evaluate several time series forecasting methods to predict the value for the demand contract to be contracted.
- Define the best forecasting model for each of the consumer units evaluated in the study.

In order to fulfill these proposed objectives, at first it was necessary to obtain and analyze the historical demand consumed by the CUs under study, to enable the definition of the best contracted demand for the future period. This choice cannot simply be the average of the historical values, as fines (penalties) may occur due to the excess of demand (overtaking demand) usually this happens because of the seasonality of consumption in each CU considered.

An algorithm, named DC12, was then developed to evaluate the demand calculation to be contracted based on the last 12 months of registered demand for a consumer unit, in order to find the value that would best fit this scenario.

The DC12 algorithm uses the last 12 months for its calculation base for two reasons: the main reason is because the demand contract lasts for 12 months and for that it is necessary to have a complete sense of how the consumer unit behaves in these 12 months. The secondary reason is that as the consumer units of group A, of CAGEPA, are exclusively for pumping water, the annual seasonality of this service must be taken into account in order to get a sense of the reality of the use of CU loads.

In sequence, a study of time series forecasting methods was performed, since the DC12 algorithm, despite being efficient in finding a reasonable value for the contracted demand, needs to be compared to other methods to evaluate its effectiveness.

Observing similar works in the literature, a study of the applicability of time series tools was carried out. For this purpose, the autoARIMA function of the forecast library of the R language was initially used, which defines the best parameters to be used in ARIMA given a time series. Then, comparing the ARIMA method with 9 other forecasting methods, with the objective of further improving the results obtained, mainly regarding adherence to the consumption profile of each UC.

The methodology used to choose the best forecasting model for each consumer unit comprises, for the last 12 months of each UC, carrying out a test with each of the forecasting methods. The method with the lowest error in a month will be considered the best method for that month. Thus, the method that has the smallest error in the sum of the last 12 months will be chosen as the best method for that UC.

1.2 Work Structure

In the development of the work, Chapter 2 presents a brief theoretical review of the basic concepts involved in the characterization and study of the aforementioned problem. Then, in Chapter 3, entitled implementation and methods, it presents the main elements that were addressed in the demand optimization study, while Chapter 4 analyzes and discusses the results obtained for each of the case studies. The work is concluded in Chapter 5, with the forwarding of the conclusions and proposal of refinements for similar researches later.

2 Theoretical Foundation

Energy management and the search for energy efficiency involves knowledge of broad concepts covered in energy management. For a better understanding of the subjects related to this study, a brief presentation of the topics listed below is necessary.

2.1 Energy Management

2.1.1 Consumer Unit - CU

Each consumer, be it a simple residence or a large complex industry, is referred by the concessionaire that supplies the electric energy as a consumer unit - CU. Consumer units are classified according to electrical voltage, energy consumption and physical supply characteristics.

2.1.2 Energy Consumption

Every electrical appliance requires a certain amount of energy to function. This consumption is defined by the amount of energy required in a time interval and can be determined by the product of demand versus time, at each instant (in Wh or multiples). For example, a lamp whose power is 30W means that every hour it will consume 30Wh, in 100 hours that lamp will have consumed 3 kWh (3,000Wh).

A CU, however, will not have just a single device, or lamp characterizing its consumption. In the case of a sanitation company, object of this study, its lifting stations will count, in addition to motor pumps, general and specific loads for example. The sum of all these powers demanded in the same instant of time brings to the concept of Energy Demand.

2.1.3 Energy Demand

The demand corresponds to the power of all loads at a given time. In fact, demand is the sum of power required by all equipment connected in the same time.

For billing purposes, demand is recorded at the UC every 15 minutes. The concessionaire uses the highest demand recorded in the billing cycle (usually 30 days), to charge the consumer.

2.1.4 Contracted and Overcome/Overtaking Demand

In order to have a control of how much energy will be required to be produced in the country and the maximum energy flow that will be demanded in the entire system, each consumer (Group A) contracts a demand that they deem necessary to supply the Consumer Unit. This value is called contracted demand, which will be charged to the consumer, regardless if he does not reach this level during the entire consumption cycle.

If demand values are recorded above the contracted demand, the maximum difference between the registered and contracted value will be charged as an exceeded demand.

For example, if the customer hires 100kW for his company and at the end of the month his biggest instant load has reached 110kW he will pay a penalty for what has been exceeded. As mentioned in the National Agency of Electrical Energy (ANEEL) Resolution 456, the overtaking rate applicable to the consumer unit billed in the conventional tariff structure must correspond to 3 (three) times the value of the normal supply tariff. Since when analyzing the difference between the price of the exceeded demand and the usual demand, a factor that varies from 2.27 to 2.5 times the value of the normal supply tariff is found in practice. For this reason, this approach was used in the calculation of the DC12 algorithm developed.

2.1.5 Calculated Demand

In the scope of this work, the Calculated Demand is the value calculated based on the demand histories, of a determined time interval, in order to obtain the lowest possible expenditure on the billing of a CU.

The algorithm that calculates the best demand for a given period of time and its functionalities are described in detail in the next chapter.

2.2 Programming Environment

For the Pyhton language Libraries (packages) such as Pandas and Numpy were used, which are scripts developed based on Python to facilitate data manipulation.

The R language was chosen to generate the forecasts of all the time series forecasting models mentioned in this work.

2.2.1 Python

Python is a high-level, interpreted, scripting, imperative, object-oriented programming language. Launched by Guido van Rossum in 1991 (PYTHON, 2020).

With its friendly and easy to use structure, as it is a high-level language. In this project, libraries (packages) such as Pandas and Numpy were used, which are scripts developed based on Python to facilitate data manipulation.

2.2.1.1 Pandas

Pandas is a library written in Python for data manipulation and analysis with several functions created with the aim of editing dataframes (PANDAS, 2020).

2.2.1.2 Numpy

Numpy is a library written in Python to manipulate multidimensional arrays and arrays in addition to having a large number of mathematical functions that helped in the execution of the project.

2.2.2 R

The main advantages of using the R language was the use of the forecast library where all the forecast functions used in this work were found.

2.3 Forecasting Techniques

Estimating future values or simply forecasting, consists of trying to predict future values using existing information and knowledge of future events that may influence the results. Forecasting techniques are widely used by organizations to base decision making, and these forecasts can be Short, Medium or Long term.

The forecast is based primarily on historical values (from the past). The reliability of the forecast depends on the existence of concrete values, so that it will be very small if strongly based without historical data.

In general, historical data are known as Time Series, which can be defined as an ordered sequence of values according to a certain time interval (DEB et al, 2017).

A very relevant feature of time series is that of Auto-correlation, which consists of a measure of the correlation between the values of a time series separated by a value in time units. Time series need to know the order of events. The changes in ambient temperature between the previous month and the next can influence water consumption and, consequently, the energy consumption of elevating substations, for example, and this is very important to make a more accurate demand forecast.

The forecasting methods applied at work and some specific functions applied to these methods are listed below.

2.3.1 ARIMA

ARIMA *(Autoregressive integrated moving average)* is a forecasting model that uses auto-regression integrated with the mean time series movement. In this work, ARIMA is used to forecast 12 months of demand and then perform the demand calculation.

Using an R language library, the function *autoarima* chooses the best parameters for a given time series.

In his dissertation Jair Prado (2011) says: "The class of ARIMA models are capable of satisfactorily describing stationary and non-stationary series, but they cannot exhibit explosive behavior"(PRADO, 2011), which in fact will be full influence on the present work and will influence the simulation results because ARIMA is unable to predict possible total disconnections and sudden changes in load of a consumer unit.

Selecting the best ARIMA model for a given time series is not simple, and there is hardly a perfect model, according to EMILIANO et al. (2010).

2.3.2 Naive

While ARIMA works with stationary and non-stationary models, Naive is able to more efficiently predict random models and is widely used in the capital markets. It is the simplest method as it uses the last value of a time series to make the forecast.

2.3.3 Arithmetic Average

The arithmetic mean function was used in the *forecast* library as well as all the forecast functions used in this work. The *meanf ()* function used in the R language of the *forecast* library is a simple summation of all input values divided by the number of input values.

2.3.4 Drift

The textit Drift follows the trend of the time series. If there is then an upward trend, textit Drift will indicate a rise in the predicted values. It occurs, similarly, if the time series presents a downward or stability trend. The algorithms used the functions (HYNDMAN, 2020):

- *rwf* - returns predictions and forecast intervals for a random time series with the deviation model applied to the y axis. This is equivalent to an ARIMA model (0.1,0) with an optional deviation coefficient.
- *Naive* - it is simply an envelope for *rwf* () in a simplified way.

2.3.5 Holt

Holt is a more sophisticated forecasting function than previously mentioned. From the use of textit Holt the intervals will have different weights. The most recent intervals will have a greater weight in the forecast. This is based on a capital market principle where the last values in which the stock is found are the most relevant values to be used for a forecast.

2.3.6 Holt Winter Additive, Multiplicative and Cushioned Multiplicative

The *Holt-Winters* model is an Exponential Smoothing model, based on the *Holt* model, which takes into account seasonality and / or the time series trend (SMARTEN, 2018). Holt Winter, like its predecessor *Holt*, also uses the weights of the last values in a time series as most relevant.

According to Feroni Andreão (2017) and Silva et al (2016) it can be used with the Additive and Multiplicative methods, considering that the additive model assumes a constant variation of seasonality over time and in the multiplicative model it is considered that the extent of seasonality varies over time, and that this variation is very likely to occur in an increasing way.

Another variation of the method is called *Holt Winter* Multiplied Damped, which also uses the multiplicative method but which dampens exponentially rather than linearly a trend existing in the series (high or low, for example).

2.3.7 Linear Regression Method

Linear models are those that have a relationship between variables that is linear in the parameters. In the linear regression method for time series, this linearity implies that the variation of each parameter mathematically is independent of the other parameters of the model. In the implemented algorithm, the function used to build linear regression models is the lm function (formula, data, weights, subset, na.action), whose main arguments are:

- 'formula' - statistical formula that indicates the model to be adjusted;
- 'data'- dataset (data.frame);
- 'weights' –weights for weighted regression;
- 'subset' – vector with the conditions that define a subset of the data;
- 'na.acation' – function that specifies what to do in case of missed observations (NA). The default value is 'na.omit' which eliminates the lines (observations) that have missing observations in the variables defined in the 'formula'.

2.3.8 Artificial Neural Network for Time Series

The function *nnetar* was used from the forecast package of language R. It adjusts a neural network model to a time series with lagged values using the time series as inputs. Therefore, it is a non-linear autogressive model and it is not possible to derive analytically prediction intervals.

Artificial Neural Networks (ANN) are a tool of great importance today, due to their ability to "learn" patterns through training, which makes their use important in the development of Artificial Intelligence. The main advantage of a neural network is its ability to approximate functional relationships, particularly when the relationships are not well defined and / or are non-linear, which makes it difficult to use conventional methods in an attempt to predict future variations in these relations.(Kovàcs, 1997)

It can be concluded from this study that the linear regression model is more efficient than the neural network model, since its application involves a reduced number of variables and a lower computational effort, when compared with the necessary parameters for the model based on in artificial neural networks. (MACHADO, 2010)

2.4 Error Metrics

2.4.1 Mean Error - ME

The metric ME *Mean Error*, is nothing more than the sum of the subtraction of the predicted values minus the tested (or realized) values, divided by the number of values (or iterations), as defined by Equation (1).

$$ME = \sum_{t=1}^{n} \frac{p_i - t_i}{n} \tag{1}$$

where: *p* and *t* are the predicted and tested values in the iteration, respectively; *n* is the number of values.

2.4.2 Mean Absolute Percentage Error - MAPE

The metric MAPE *Mean Absolute Percentage Error*, is equivalent to the error for absolute and percentage values, as defined by Equation (2).

$$MAPE = \sum_{t=1}^{n} \frac{\frac{|p_i - t_i|}{|t_i|}}{n} \tag{2}$$

In the paper, because it is a percentage value, MAPE is used as a comparative indicator between different UCs.

3 Implementation and methods

The historical values of demand consumed were taken from the database of the ENERGIA program, which is a private software under the exclusive right of CAGEPA used to monitor and manage all the company's UCs. In view of the large data collection, Pandas was used to perform data analysis and organization.

3.1 Function DC12 - Calculated Demand 12 months

To find the value of the contracted demand that will result in the lowest cost for the company, the DC12 algorithm developed in the Python programming language is used, whose mathematical model is presented in the equations (3) e (4).

$$[P(i,j)] = \begin{Bmatrix} 3 & se & Dh_j - Dh_i > 0 \\ -1 & se & Dh_j - Dh_j < 0 \end{Bmatrix} \tag{3}$$

$$DC12[i] = \sum_{j=Dh_{min}}^{Dh_{max}} \sum_{i=Dh_1}^{Dh_{len}} [Dh_j - Dh_i] \times [P(i,j)] \tag{4}$$

Dh = History of Consumed Demand
Dhmáx = Highest Demand Consumed from a Historical Data
Dhmín = Less Demand Consumed from a Historical Data
Dhlen = the lenghth of the demand history
P(i,j) = Conditional function that returns 3 or -1 depending on the value of $Dh_j - Dh_i$ be greater or less than 0

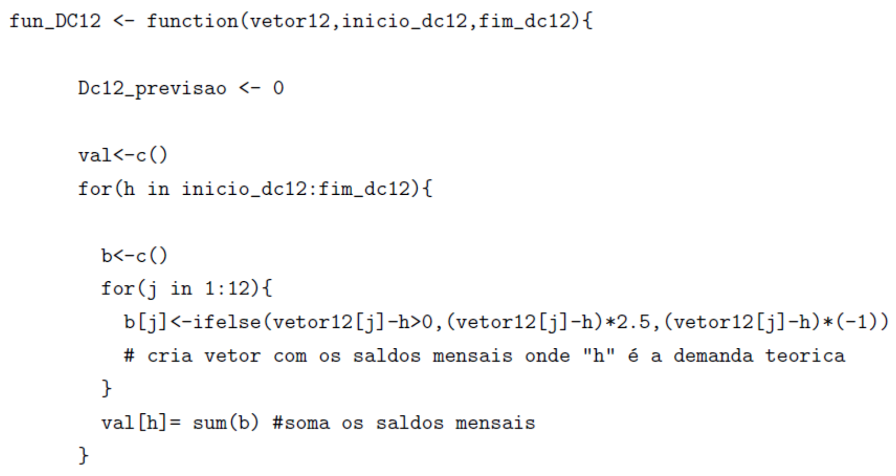
The DC12 Vector [i] will store monetary values in a historical data of demand consumed, of a vector that varies from Dhmin to Dhmáx. The lowest value of the DC12 vector will bring the index of the best demand value that should be the contracted demand for the last 12 months, that is, the value that would result in the best contracted demand value (paying the lowest amount to the concessionaire). For example, considering a vector of historical demands Dh = [80, 79, 85, 85, 80, 85, 82, 83, 81, 80, 82, 80], it is observed that the minimum value is Dhmin = 80, the maximum value is Dhmáx = 85 and the size of the vector Dhlen = 12. So, for this example the first summation of j varies from 80 to 85, while the summation of i will follow the vector Dh in the order Dh [1] to Dh [12], having as its first element 80. Therefore, in the first iteration, the first result will be 80-80 (eighty minus eighty) which will return 0. In the second iteration it will also be 80-79 returning the value -1, as $Dh_j - Dh_i$ <0, the conditional function will return -2.5. Therefore, the result that will be stored in the DC12 vector of this interaction will be -2.5 * (- 1) = 2.5. In the third iteration it will be 80-85 which will result in

-5, but as the conditional function imposes that values less than 0 the result is multiplied by -1, the result will be 5. And so consecutively, at the end of the 12 elements the sum of this value is stored in the first index of the vector DC12 [i]. The DC12 vector in turn will have the monetary sum referring to the demands of 80 to 85 (Dhmin to Dhmáx) the lowest value of the vector has the index of the optimal demand.

There are two constants in the equation **??** refers to the price of the fine which is three times higher than the normal demand price which in the equation **??** is exposed as -1 so that the resulting multiplication with $Dh_j - Dh_i$ <0 results in a positive value. It is also important to point out that the value 3 was not used in the simulations, as an average of 2.5 times the fine was calculated in the CAGEPA accounts, taking into account that only registered demand values that exceed 5 % of the contracted demand are charged .

For a better understanding of this function, it is suggested to the reader the detailed interpretation of the code presented in Appendix B. This specific function is represented in R language in the part of the code illustrated in Figure 1.

Figura 1 – DC12 function coding



```
fun_DC12 <- function(vetor12,inicio_dc12,fim_dc12){

   Dc12_previsao <- 0

   val<-c()
   for(h in inicio_dc12:fim_dc12){

     b<-c()
     for(j in 1:12){
       b[j]<-ifelse(vetor12[j]-h>0,(vetor12[j]-h)*2.5,(vetor12[j]-h)*(-1))
       # cria vetor com os saldos mensais onde "h" é a demanda teorica
     }
     val[h]= sum(b) #soma os saldos mensais
   }
```

FONTE: Autoria Própria

3.2 Forecast Analysis

3.2.1 Demand History

147 Consumer Units were used for this simulation with demand values varying from 2003 to 2019. An example of the demand information for one of the Consumer Units used in this work is found in Table 1 where Dh is the Demand (Registered Demand).

3.2.2 Cascading DC12 function calculation

As explained in section 3.1, the calculation of DC12 will result in the best value for the demand contract for the last 12 months. It is worth emphasizing that when you can analyze the result of DC12 in two different ways. The first is when registered or historical demand is

Tabela 1 – Example of Consumer Unit and its demand values

UC	MATRICULA	ANO	MES	Dh
1	####	2003	1	65,19
	####	2003	2	80,77
	####	2003	3	64,37
	####	2003	4	64,78
	####	2003	5	65,19
	####	2003	6	130,38
		...	...	...
	####	2019	1	112,34
	####	2019	2	114,8
	####	2019	3	106,6
	####	2019	4	105,78
	####	2019	5	110,7
	####	2019	6	113,16

used to define the demand contract. When this occurs, the DC12 function result will be called DC12-P because it will be used as a forecast (prediction). A second way to analyze the use of the DC12 function result is when this result is used to test a forecast, in this case the DC12 function result will be called DC12-T, that is, DC12 test.

Using the Pandas library of the *Python* language it was possible to perform the cascade demand calculation. That is, every 12 months an optimal demand value is calculated and stored (month 1 to month 12), then the vector is "displaced"eliminating the demand value for the first month (month 2 to month 13). This process is carried out until the last month of the CU's history is reached.

An example of this process is illustrated in Table 2, in which the first value found at 125 kW refers to the best demand value from January 2003 to December 2003. Its successor, 128 kW, refers to the best value for contracted demand from February 2003 to January 2004 and so on.

3.2.3 Analysis of the use of the DC12 function for each CU

Following the presented algorithm, DC12 will obtain the best demand value based on the last 12 months. However, it is not guaranteed that this value will be the best value to be assigned to a future demand contract.

Considering that the first calculation of DC12 for the first 12 months will represent the optimal demand for this period of time and that the person responsible for the CU then decides to use this value for the demand contract for the next 12 months (DC12-P), after plus 12 months (24 months in total) the DC12 (DC12-T) can be recalculated to compare it with the defined contract value and calculate the metric errors with such an estimate: ME and MAPE.

For example, for CU 1, having its history of demand with 196 values, it becomes possible to compare the forecast, which is the previous DC12 calculation (DC12-P) with its

Tabela 2 – Results of routine DC12 obtained from the demand history

UC	MATRICULA	ANO	MES	Dh	DC12
1	####	2003	1	65,19	
	####	2003	2	80,77	
	####	2003	3	64,37	
	####	2003	4	64,78	
	####	2003	5	65,19	
	####	2003	6	130,38	
	####	2003	7	128,74	
	####	2003	8	127,92	
	####	2003	9	125,46	
	####	2003	10	120,54	
	####	2003	11	120,54	
	####	2003	12	118,08	
	####	2004	1	113,8	125
	####	2004	2	139,58	128
	####	2004	3	121,36	128
	...	...	...	...	...
	####	2019	1	112,34	120
	####	2019	2	114,8	116
	####	2019	3	106,6	115
	####	2019	4	105,78	115
	####	2019	5	110,7	113
	####	2019	6	113,16	113

actual historical values, performing the DC12 calculation after 12 months (DC12-T) that were estimated in the demand contract. This comparison can be made 196-12 = 184 times for CU 1.

3.2.4 Analysis of the use of the ARIMA function in each CU

If it is possible to calculate the difference between the forecast and the test (DC12-P x DC12-T), it is possible to use a temporal forecast model to predict how demand consumption will behave in the next 12 months and perform the calculation of DC12. For this purpose, the ARIMA model with integrated regression of moving averages was chosen.

The analysis method used takes into account the cascade structuring for the analysis of a CU, as seen in the previous session, which, for example, has 196 demand values for the year 2003 to 2019.

The first 25 historical values (corresponding to the minimum value accepted by the ARIMA function) are stored for the CU to be studied, the forecast for the 12 future months is carried out and the function DC12 is calculated. This process is repeated, moving one month in the initial vector until the entire demand history is tested.

The 3 table shows the DC12-P and ARIMA forecasts as well as the values used to test the DC12-T. The ME and MAPE error metrics are highlighted in the relationship between

DC12-P x DC12-T and ARIMA x DC12-T.

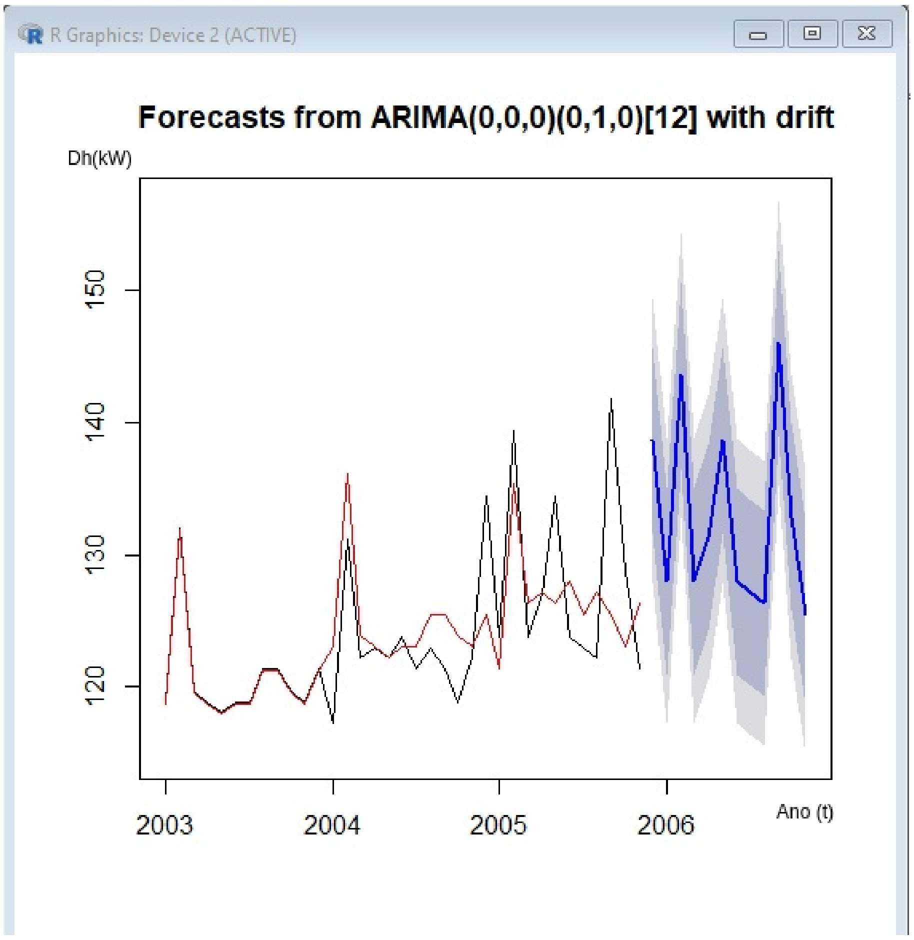
When the DC12 function result is used to make a forecast, this result is called DC12-P and when DC12 is used as the test, the objective to be achieved, this result is called DC12-T.

Tabela 3 – DC12-P and ARIMA forecasts and their error metrics with respect to DC12-T

	PREVISÕES		**TESTE**	**MÉTRICAS DE ERRO**			
Amostra	**DC12-P (Previsão)**	**ARIMA**	**DC12-T (Teste)**	**ME DC12-P**	**MAPE DC12-P**	**ME ARIMA**	**MAPE ARIMA**
1	125	121	127	-2	1.6	-6	4.95
2	124	126	128	-3	2.4	-2	1.58
3	125	122	129	-4	3.2	-7	5.73
4	125	128	130	-5	4	-2	1.56
5	125	126	131	-6	4.8	-5	3.96
6	125	118	131	-6	4.8	-13	11.02
7	125	125	131	-6	4.8	-6	4.8
8	125	125	131	-6	4.8	-6	4.8
9	125	127	132	-7	5.6	-5	3.94
10	125	124	132	-6	4.761	-8	6.45
11	125	136	132	-5	3.93	4	2.941
12	126	132	133	-6	4.72	-1	0.76
13	127	140	133	-6	4.72	7	5
14	127	137	133	-5	3.90	4	2.92
15	127	141	133	-4	3.1	8	5.67
16	128	138	133	-3	2.30	5	3.62
17	129	138	133	-2	1.53	5	3.62
18	130	142	133	-2	1.53	9	6.34
19	131	132	133	-2	1.53	-1	0.76
20	131	131	133	-2	1.53	-2	1.53
21	131	131	133	-1	0.76	-2	1.53
22	131	135	133	-1	0.76	2	1.48
23	132	135	133	-1	0.76	2	1.48
24	132	30	132	1	0.76	-102	340
25	132	94	132	1	0.76	-38	40.42
26	133	118	131	2	1.50	-13	11.02
27	133	123	130	3	2.26	-7	5.69
28	133	127	130	3	2.26	-3	2.36
29	133	129	129	4	3	0	0

The ARIMA function is found in the *forecast* library of the R programming language and was used in its standard format. The figure **??** shows the forecast made for ARIMA when the consumer unit had 35 elements in its demand history. The 12 values that were generated (predicted) in the image are shown in blue.

Figura 2 – Forecast made by ARIMA



FONTE: Autoria Própria

4 Results

4.1 DC12-P x ARIMA - Error metric

Now equipped with two ways to calculate the most efficient demand to be contracted for the coming year; one using the history of demand consumed and performing the calculation of maximum savings through the function DC12 (resulting in the DC12-P), and another using the history of demand consumed and projecting through ARIMA the next 12 months to then perform the demand calculation ; it is necessary to find out which of the two techniques is the most efficient for each Consumer Unit.

The study was carried out by comparing the values in the table 3. For comparison, the Pandas library of the Python programming language was used. The 4 table shows results of the comparison of the number of times that the ARIMA and DC12-P methods perform better for each consumer unit. In the ARIMA and DC12-P column, the values indicated are how many times it was more efficient to use each method, that is, the method whose MAPE between the forecast and the test obtained the lowest percentage index.

Tabela 4 – Performance comparison between ARIMA and DC12-P methods

UC	Arima	DC12-P	empate
1	68	58	33
2	50	65	44
3	61	90	8
4	63	86	10
5	20	75	64
6	85	51	23
7	65	83	11
8	36	39	41
9	38	54	6
10	20	45	20
11	30	22	0
12	13	7	0
13	46	85	9
14	33	58	16
15	42	55	10
—	—	—	—
134	49	82	28
135	25	30	104
136	38	35	66
137	41	105	13
138	25	129	5
139	79	60	20
140	42	105	12
141	71	84	4
142	47	100	11
143	83	74	2
144	48	105	6
145	53	90	16
146	58	62	4
147	67	38	53

The 5 table shows only the consumer units where the ARIMA method was the best option, taking into account the possible values for each CU. The tie column represents when the results of the ARIMA and DC12-P values were the same. of the 147 CUs analyzed, 38 CUs proved to be more efficient in general with ARIMA, which represents 25.85% of the total.

Tabela 5 – Comparison ARIMA x DC12-P - ARIMA winning

UC	Arima	DC12-P	empate
1	68	58	33
6	85	51	23
11	30	22	0
12	13	7	0
23	57	41	4
27	28	14	35
32	29	19	32
33	35	13	18
34	28	22	6
35	39	25	0
36	30	27	7
37	30	6	22
50	56	49	54
58	61	61	37
68	72	71	16
69	68	68	23
70	56	50	53
73	58	30	71
75	70	68	21
78	22	19	118
79	37	27	94
80	66	55	38
85	57	53	17
86	51	47	29
95	97	32	30
105	68	68	23
113	66	31	62
114	61	59	39
116	54	54	13
118	69	69	20
123	6	0	128
124	69	63	13
128	42	40	10
129	83	63	13
136	38	35	66
139	79	60	20
143	83	74	2
147	67	38	53

As ARIMA is a technique that needs the largest possible number of values to make a good forecast and as the main objective of this work is to find the optimized demand to be used in the next demand contracts, the table 6 presents the comparison between ARIMA x DC12-P for the last 12 months.

Tabela 6 – Comparison ARIMA x DC12-P - Last 12 months

UC	Arima	DC12-P	Empate
1	8	2	2
2	7	5	0
3	4	3	5
4	10	1	1
5	1	7	4
6	11	0	1
7	9	1	2
8	3	2	7
9	5	6	1
10	2	7	3
11	6	6	0
12	0	0	0
13	0	12	0
14	5	6	1
15	5	7	0
—	—	—	—
133	0	12	0
134	2	7	3
135	3	0	9
136	4	3	5
137	3	9	0
138	0	12	0
139	6	3	3
140	0	12	0
141	3	9	0
142	0	12	0
143	8	4	0
144	0	12	0
145	2	4	6
146	5	7	0
147	4	1	7

The 7 table shows the comparison between ARIMA and DC12-P for the last 12 months only when the ARIMA technique is the best to be used in the demand contract. There were 66 consumer units, representing 44.89% of the CUs that were analyzed. The symbol (=) was used to represent the tie.

Tabela 7 – Comparação ARIMA x DC12-P - Último 12 meses - ARIMA ganhando

UC	**Arima**	**DC12-P**	**=**	**UC**	**Arima**	**DC12-P**	**=**	**UC**	**Arima**	**DC12-P**	**=**
1	8	2	2	*63*	11	1	0	*108*	4	4	4
2	7	5	0	*64*	7	5	0	*110*	7	5	0
3	4	3	5	*66*	8	2	2	*113*	5	2	5
4	10	1	1	*67*	6	4	2	*114*	8	1	3
6	11	0	1	*68*	4	2	6	*115*	0	0	12
7	9	1	2	*72*	10	0	2	*116*	11	1	0
8	3	2	7	*74*	9	3	0	*118*	10	2	0
11	6	6	0	*75*	10	2	0	*120*	6	4	2
12	0	0	0	*76*	7	5	0	*121*	2	0	10
19	7	4	1	*78*	6	2	4	*123*	0	0	12
20	3	2	7	*79*	2	0	10	*126*	12	0	0
23	10	2	0	*82*	8	4	0	*127*	12	0	0
25	12	0	0	*86*	3	3	6	*129*	8	4	0
27	9	0	3	*88*	7	5	0	*131*	7	4	1
33	7	3	2	*90*	8	3	1	*132*	7	4	1
36	9	2	1	*91*	7	5	0	*135*	3	0	9
37	3	2	7	*93*	11	1	0	*136*	4	3	5
50	6	1	5	*95*	6	2	4	*139*	6	3	3
51	7	0	5	*96*	7	5	0	*143*	8	4	0
56	5	5	2	*102*	5	4	3	*147*	4	1	7
57	8	0	4	*103*	9	0	3				
60	8	3	1	*105*	11	1	0				

The table 8 shows the results of the forecasts and their percentage errors in Consumer Units where ARIMA was considered the best forecast method using the error metric.

Tabela 8 – Forecast results in CUs where ARIMA performed best

UC	Arima	MAPE (%)		UC	Arima	MAPE (%)
1	123	0.81		*75*	110	1.81
2	198	1.51		*76*	92	2.17
3	482	2.07		*78*	56	1.78
4	355	0.0		*79*	41	2.43
6	68	1.47		*82*	30	26.66
7	71	7.04		*86*	82	1.21
8	31	35.48		*88*	53	30.18
11	141	30.49		*90*	847	2.01
12	67	0.0		*91*	49	22.44
19	167	2.39		*93*	46	0.0
20	30	120.0		*95*	95	1.052
23	142	28.16		*96*	67	28.35
25	177	38.98		*102*	336	12.5
27	57	3.50		*103*	138	0.0
33	54	1.85		*105*	74	1.35
36	140	3.57		*108*	45	60.0
37	43	2.32		*110*	50	6.0
39	59	13.55		*113*	111	56.75
41	436	2.293		*114*	40	5.0
42	48	2.08		*115*	30	0.0
43	564	2.83		*116*	75	6.66
44	507	2.56		*118*	315	2.22
45	103	7.76		*120*	57	10.52
50	53	3.77		*121*	48	2.08
51	94	50.0		*123*	30	170.0
56	61	0.0		*126*	69	2.89
57	69	1.44		*127*	51	19.60
60	83	3.61		*129*	115	22.60
63	216	28.70		*131*	41	9.75
64	146	2.05		*132*	167	0.59
66	99	18.18		*135*	40	0.0
67	131	8.39		*136*	31	0.0
68	75	8.0		*139*	143	4.89
72	88	6.81		*143*	220	0.0
74	274	15.32				

4.1.1 DC12-P x ARIMA - Demand Fine Weight

The method previously used used the error metric to define the best technique to define the contracted demand of a consumer unit. In this session the weight of the fine was used, that is, when the forecast value exceeds the optimal value of demand, the average weight of the fine will be attributed to consumer units of the group in the state of Paraíba. The table 9 shows the comparison of the number of times that the ARIMA and DC12-P methods perform better in the last 12 months taking into account the weight of the fine only when ARIMA wins from DC12-P that occurred in 70 consumer units or 47.61%.

Tabela 9 – Comparison ARIMA x DC12-P - Last 12 months - Weight of Fine - ARIMA winning

UC	**Arima**	**DC12-P**	**=**	**UC**	**Arima**	**DC12-P**	**=**	**UC**	**Arima**	**DC12-P**	**=**
1	8	2	2	*50*	6	1	5	*95*	6	2	4
3	4	3	5	*51*	7	0	5	*96*	7	5	0
4	10	1	1	*56*	5	5	2	*102*	5	4	3
6	11	0	1	*57*	8	0	4	*103*	9	0	3
7	7	3	2	*60*	8	3	1	*105*	10	2	0
8	4	1	7	*63*	11	1	0	*108*	4	4	4
11	6	6	0	*64*	6	6	0	*110*	8	4	0
12	0	0	0	*66*	8	2	2	*113*	5	2	5
19	7	4	1	*67*	6	4	2	*114*	8	1	3
20	3	2	7	*68*	4	2	6	*115*	0	0	12
23	10	2	0	*72*	8	2	2	*116*	11	1	0
25	12	0	0	*74*	9	3	0	*118*	8	4	0
27	9	0	3	*75*	10	2	0	*121*	2	0	10
33	7	3	2	*76*	7	5	0	*123*	0	0	12
36	8	3	1	*78*	6	2	4	*126*	10	2	0
39	0	0	0	*79*	2	0	10	*127*	12	0	0
41	0	0	0	*82*	8	4	0	*129*	8	4	0
42	0	0	0	*86*	3	3	6	*131*	7	4	1
43	0	0	0	*88*	7	5	0	*135*	3	0	9
44	0	0	0	*90*	9	2	1	*136*	4	3	5
45	0	0	0	*91*	7	5	0	*139*	6	3	3
47	4	3	5	*93*	11	1	0	*143*	8	4	0
								147	4	1	7

The table 10 shows the result of the forecasts and their percentage errors in Consumer Units where ARIMA was considered the best forecast method using the weight of the fines.

Tabela 10 – Forecast results in CUs where ARIMA performed better

UC	Arima	Erro (%)		UC	Arima	Erro (%)
1	123	0.81		*75*	110	1.81
3	482	2.07		*76*	92	2.17
4	355	0.0		*78*	56	1.78
6	68	1.47		*79*	41	2.43
7	71	7.04		*82*	30	26.66
8	31	35.48		*86*	82	1.21
11	141	30.49		*88*	53	30.18
12	67	0.0		*90*	847	2.01
19	167	2.39		*91*	49	22.44
20	30	120.0		*93*	46	0.0
23	142	28.16		*95*	95	1.05
25	177	38.98		*96*	67	28.35
27	57	3.50		*102*	336	12.5
33	54	1.85		*103*	138	0.0
36	140	3.57		*105*	74	1.35
39	59	13.55		*108*	45	60.0
41	436	2.29		*110*	50	6.0
42	48	2.08		*113*	111	56.75
43	564	2.83		*114*	40	5.0
44	507	2.56		*115*	30	0.0
45	103	7.76		*116*	75	6.66
47	65	7.69		*118*	315	2.22
50	53	3.77		*121*	48	2.08
51	94	50.0		*123*	30	170.0
56	61	0.0		*126*	69	2.89
57	69	1.44		*127*	51	19.60
60	83	3.61		*129*	115	22.60
63	216	28.70		*131*	41	9.75
64	146	2.05		*135*	40	0.0
66	99	18.18		*136*	31	0.0
67	131	8.39		*139*	143	4.89
68	75	8.0		*143*	220	0.0
72	88	6.81		*147*	30	0.0
74	274	15.32				

4.2 10 forecasting methods x DC12-P

To finish the study, instead of using only ARIMA, 10 forecasting methods were used to define the best contracted demand. The 11 table shows an overview of the performance of the 10 methods and the DC12-P while the 12 table summarizes the result.

Tabela 11 – Performance of the 10 forecasting methods and DC12-P - Overview

UC	naive	meanf	rwf	holt	hw	hw2	hw3	liner	neural	DC12-P	ARIMA
1	5	1	0	0	0	4	0	0	0	3	10
2	0	0	4	5	3	1	1	0	4	0	0
3	6	0	0	0	1	1	0	0	0	2	3
4	0	4	1	0	0	0	0	6	1	2	2
5	6	0	0	0	0	1	1	0	2	11	1
6	3	0	0	3	4	4	3	8	1	2	3
7	1	3	2	1	2	2	3	2	1	5	2
8	1	10	1	1	0	0	0	0	0	0	0
9	1	0	1	1	1	0	4	3	1	2	2
10	1	0	1	2	0	2	3	5	3	1	2
11	1	0	3	2	3	0	0	1	0	2	0
12	3	4	0	3	2	0	0	1	1	2	2
13	4	5	3	1	2	2	0	0	0	2	0
14	2	1	0	1	2	4	0	0	0	5	2
15	0	0	2	1	1	1	1	0	2	6	0
—	—	—	—	—	—	—	—	—	—	—	—
133	4	0	5	5	1	1	1	0	0	3	1
134	0	0	0	0	1	2	0	6	1	6	0
135	7	0	8	8	9	6	2	1	4	8	8
136	6	0	6	9	7	5	8	6	11	9	7
137	1	0	4	0	2	0	0	0	0	6	0
138	3	0	0	0	0	0	0	0	6	0	4
139	5	0	2	0	0	1	1	0	1	2	3
140	2	0	1	1	1	1	1	0	0	5	2
141	2	0	1	0	0	0	0	0	6	4	1
142	1	0	0	0	0	0	1	4	1	5	0
143	0	0	2	3	1	2	0	0	3	1	1
144	4	0	4	7	0	0	2	0	1	2	0
145	1	5	1	3	1	2	2	0	4	5	4
146	1	0	0	0	0	1	3	0	1	7	5
147	3	0	3	3	2	1	3	9	1	3	3

the table 13 shows a simulation where each consumer unit was assigned the value of its most efficient method as shown in the selection of the table 11. The error per unit of demand (EUD) considers the weight of the fine here set at 2.5 to compare how much of the unit of demand is being paid in each of the forecasting methods (Forecast, in table 11, refers to the best forecasting method found for each unit (its error expressed in the table as EUD-P). the sum of the last column "EUD difference EUD forecast DC12-P"resulted in -376 demand units.

Tabela 12 – Performance of 10 forecasting methods and DC12-P

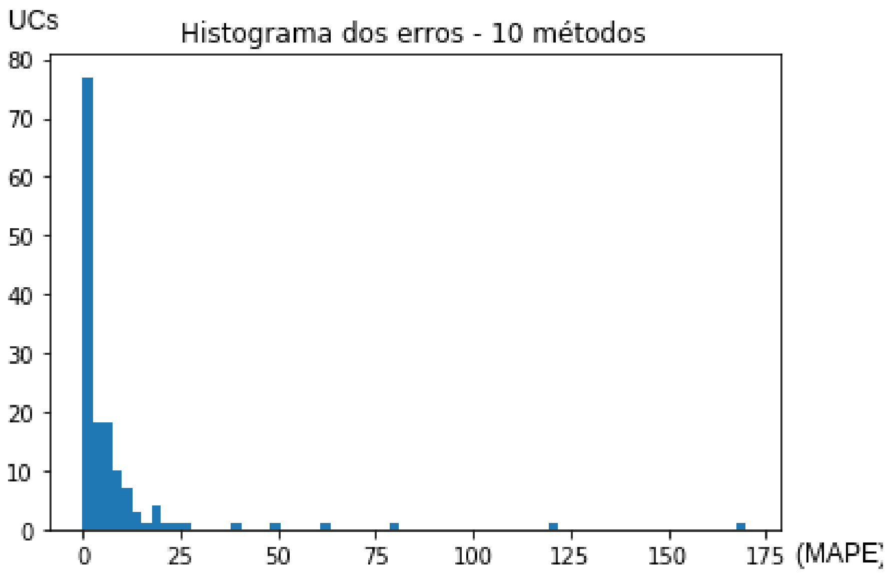
Método	**Quantidade de UC**
Naive	14
Meanf	25
Drift	9
Holt	8
Holt winter aditivo	6
Holt winter multiplicativo	8
Holt winter multiplicativo amortecido	2
Tslm	27
Nnetar	12
DC12-P	27
ARIMA	9

Tabela 13 – Simulation - comparison between 10 forecasting models vs. historical demand

UC	Previsão	Erro (%)	EUD/P	EUD/DC12-P	EUD/P - EUD/DC12-P)
1	123.0	0.81	1.0	7.0	-6.0
2	203.0	0.98	2.0	3.0	-1.0
3	483.0	2.27	11.0	11.0	0.0
4	377.0	5.83	22.0	4.0	18.0
5	88.0	0.0	0.0	0.0	0.0
6	69.0	2.89	2.0	1.0	1.0
7	91.0	7.31	6.0	6.0	0.0
8	37.0	13.51	5.0	11.0	-6.0
9	70.0	4.28	3.0	5.0	-2.0
10	55.0	9.09	5.0	4.0	1.0
11	101.0	2.97	3.0	73.0	-70.0
12	73.0	8.21	6.0	18.0	-12.0
13	62.0	25.80	16.0	1.0	15.0
14	37.0	6.52	3.0	3.0	0.0
15	44.0	7.69	6.0	6.0	0.0
16	55.0	1.81	1.0	1.0	0.0
17	127.0	6.29	8.0	28.0	-20.0
18	79.0	1.26	1.0	0.0	1.0
19	217.0	24.88	54.0	0.0	54.0
20	30.0	120.0	36.0	36.0	0.0
21	204.0	1.96	4.0	28.0	-24.0
22	69.0	7.24	5.0	1.0	4.0
23	104.0	1.92	2.0	93.0	-91.0
24	63.0	1.58	1.0	2.0	-1.0
25	176.0	38.63	68.0	87.0	-19.0
26	85.0	11.76	10.0	9.0	1.0
27	56.0	1.78	1.0	3.0	-2.0
28	103.0	2.912	3.0	2.0	1.0
29	35.0	0.0	0.0	0.0	0.0
30	51.0	3.92	2.0	2.0	0.0

The figure 3 shows the histogram of the errors of the 10 forecasting methods for the 147 consumer units. This result shows the error of the MAPE metric (percentage). The y-axis shows us the quantity of CU while the x-axis shows the percentage of the MAPE error.

Figura 3 – Histogram Errors 10 methods in percentage



FONTE: Own Authorship

5 Conclusion

The study sought to define the best value for the contracted demand, knowing that this value remains unchanged for 12 months. The first approach was to define an algorithm capable of identifying which would be the best value given a vector of 12 elements containing the demand history. DC12, demand calculated for 12 months, was the technique developed for this purpose. If it were possible to predict what would be the next demand values consumed for the next 12 months, the DC12 function would undoubtedly be the correct way to define the demand contract. However, the demand consumed is somewhat variable. It depends on how the loads from a Consumer Unit were triggered during the month. The initial idea to define the demand contract was to use the last 12 months and to assume that this same value would be projected in the future.

Although the DC12- P represented a good approximation to the ideal value of the demand contract. It proved to be inefficient because the Consumer Unit and its consumed demand is a living and variable organism. Decided to make a prediction of CU behavior, the most robust mathematical model of today (ARIMA) was used to try to predict what the values of the next 12 months would be and with these values perform the calculation of the DC12 function. The results were satisfactory, for the last 12 months of each CU, ARIMA was more efficient than DC12-P in 44.89% of the total of CU. For this first analysis, MAPE *Mean absolute percentage error* was used to compare ARIMA with DC12-P. Despite finding the smallest absolute error between the two techniques and being the best way to find the closest value to the ideal, MAPE did not take into account that when the value of demand consumed exceeded that of the contractor, there is a fine, which in Paraíba is on average 2.4 times the normal value.

To perform the comparison between ARIMA and DC12-P using the weight of the fines, the ME *Mean Error* was used. Whenever the forecast or historical demand exceeded the ideal demand value, the mean error had a negative value, so it was enough to multiply it by the -2.4 fine to be able to compare the values, properly assigning the weight of the fines. When comparing the results of the last 12 months with the weight of the fines, the ARIMA forecast became the best in 47.61% of the CUs.

Finally, 10 different forecasting methods were used, taking into account the weight of the fines. This time in 81.63% of the UC the predictive methods were the best to be used. Ideally, 100% of the units would use some of the forecasting methods since the DC12-P uses the last 12 months of historical demand to perform the calculation and any forecast, even if it is a reflection of the last 12 months, tends to perform better than DC12-P.

In the simulation performed in this study represented by the table 13, the sum of the last column "EUD-P - EUD-DC12-P", which is the difference between the ME of the forecast and the ME of the DC12-P with the weight of the fine already assigned, resulted in -376 demand units, that is, the use of the 10 forecast models in that month resulted in savings of 376 demand units (or R$/kW). This demand unit value varies from dealership to dealership as it also takes into account the tariff modality in which the consumer unit is located. In Paraíba, for the year 2019-2020, this value for group A CU ranges from R$ 11.11 to R$ 52.77. In order to preserve the data of the company that provides the data for this work, this simulation hid the current value of contracted demand as well as all consumer units with more than 1000kW of contracted demand, otherwise the value saved with this technique would be even greater.

The figure 3 shows the MAPE of the simulation that was done to test the 10 methods for all the tested CU. The vast majority of CUs have an error of less than 2.5% while almost all of them are below 25% which is a good performance for the tested model, it is worth remembering that the exceeded demand is only charged when the registered demand exceeds 5% of contracted demand which makes the result shown in the MAPE histogram excellent. The 6 Units with error values above 25% require a more detailed analysis by the energy manager. Probably in these 6 Units there was a sudden change in the use of their loads, either due to disconnection or a new quantity of equipment purchased. It is then up to the manager to investigate directly what happened and to inform himself about sudden changes in general for his consumer units in group A because neither DC12-P neither mathematical forecasting models were able to predict this reaction of demand consumed.

The results of the study were satisfactory. For the future, more mathematical forecasting models should be sought to further reduce the influence of DC12-P on consumer units. In addition, this entire study and its methodology can be used for consumption history in order to adapt not only demand contracts but also the tariff modality of group A consumer units.

Referências

Albuquerque, Felipe Oliveira. **Otimização robusta aplicada à contratação de energia elétrica considerando incerteza na demanda futura** / Felipe Oliveira Albuquerque. São José dos Campos, 2015.

IBGE. **Anuário Estatístico do Brasil** . Disponível em : <https://biblioteca.ibge .gov.br/visualizacao/periodicos/20/aeb_2014.pdf> Acesso em: Outrubro 2019.

Prado, Jair Rocha do. **MODELOS PARA DEMANDA E CONSUMO DE ENERGIA ELÉTRICA UTILIZANDO SÉRIES TEMPORAIS NA UNIVERSIDADE FEDERAL DE LAVRAS**. 2011.

Hyndman, J Rob. **naive: Naive and Random Walk Forecasts**. Disponível em : <https://rdrr.io/cran/forecast/man/naive.html> Acesso em : Julho de 2020.

Machado, Telmo Nuno.Teixeira, João Paulo Fernandes, Paula O.**Modelação da procura turística em Portugal: regressão linear versus redes neuronais artificiais**.2010. Universidade de Aveiro.

Carvalho, Daniel José de **Métodos de previsão de consumo de energia elétrica residencial em grande volume de dados** Daniel José de Carvalho. - 2019

FERONI, R. C.; ANDREÃO, W. L. **Análise do modelo de holt-winters aplicado a uma série histórica de dados com tendência e sazonalidade.** In: Blucher Physics Proceedings. Editora Blucher, 2017. p. 228–231. Disponível em: <http: //www.proceedings.blucher .com.br/article-details/27773>.

SILVA, D. A. d.; SANTOS, M. E. d.; COSTA, D. F. A **utilização do modelo holt-winters na elaboração de um orçamento de resultado de uma cooperativa de crédito rural.** Revista de Contabilidade do Mestrado em Ciências Contábeis da UERJ, v. 21, n. 1, 2016. Using R. **Modelos Lineares** Disponível em : <http://ecologia.ib.usp.br/bie5782 /doku.php?id=bie5782:03 apostila:06-modelos> Acesso em : Agosto de 2020.

Kovàcs, Z. L.; **Redes Neurais Artificiais: Fundamentos e Aplicações**, 2. ed., São Paulo:Collegium Cognitio, 1997.

Tanyavutti,Tanyavutti. Tanlamai, Utha. **ARIMAX versus Holt Winter Methods: The Case of Blood Demand Prediction in Thailand**. 2018. INTERNATIONAL JOURNAL OF ENVIRONMENTAL SCIENCE EDUCATION.2018, Vol. 13.

DALMAS, J. C. et al. **Determinação de um modelo de previsão do consumo de energia elétrica no Estado do Paraná.** In: REUNIÃO ANUAL DA REGIÃO BRASILEIRA DA SOCIEDADE INTERNACIONAL DE BIOMETRIA, 48.; SIMPÓSIO DE ESTATÍSTICA APLICADA À EXPERIMENTAÇÃO AGRONÔMICA, 10., 2003, Lavras. Anais... Lavras: UFLA, 2003. p. 702-706.

GIAROLA, L. T. P.; ROCHA, R. C. **Modelagem do consumo de energia elétrica**

no Sudeste Brasileiro. In: REUNIÃO ANUAL DA REGIÃO BRASILEIRA DA SOCIEDADE INTERNACIONAL DE BIOMETRIA, 53., 2008, Lavras. Anais... Lavras: UFLA, 2008. 1 CD-ROM.

CAMPOS, R. J.; JESUS, T. A.; MENDES, E. M. A. M. **Uma abordagem Fuzzy para a previsão de curto-prazo do consumo de energia elétrica.** In: CONGRESSO NACIONAL DE MATEMÁTICA APLICADA E COMPUTACIONAL, 30., 2007, Florianópolis. Anais... Florianópolis: UFSC, 2007. Disponível em: <http://www.sbmac.org.br/eventos/cnmac/xxx_cnmac/30cnmac.php>. Acesso em: 10 out. 2010.

CAMPOS, R. J. **Previsão de séries temporais com aplicações a séries de consumo de energia elétrica.** 2008. 110 p. Dissertação (Mestrado em Engenharia Elétrica) - Universidade Federal de Minas Gerais, Belo Horizonte, 2008.

MARTARELLI FILHO, A. **Estimação de tipologia para dados funcionais agrupados.** 2006. 86 p. Dissertação (Mestrado em Estatística) - Universidade de Campinas, Campinas, 2006.

PANDAS. Disponível em : https://pandas.pydata.org/ . Acesso em : 18 de agosto.

PYTHON . Disponível em : https://www.python.org/about/ . Acesso em : 18 de agosto.
AGÊNCIA NACIONAL DE ENERGIA ELÉTRICA -ANEEL. RESOLUÇÃO N.º456, DE 29 DE NOVEMBRO DE 2000.

Apêndices

APÊNDICE A – Python programming codes

The work used several scripts to complete. To assist the reader in understanding the results found, the following code shows in general the organization and structuring of data in Python but that were generated in R. The author chose Python for structuring the data for reasons of greater intimacy with the language .

```
##### 10 metodos com peso da multa ######
##### Criado por Thyago Carvalho #######

import pandas as pd
import numpy as np
df = pd.read_excel("C15_DTR30_v2.xlsx")

# A ordem das UC foi alterado porq algumas nao estavam com o minimo de necesario
#esse codigo reorganiza as UC utilizadas

array= df["MATRICULA"].unique()
for y in range(len(df["Num"].unique()-1)): # existe um nan
    for x in range(len(df)):
        if (array[y] == df.loc[x,"MATRICULA"]):
            df.loc[x,("Num")] = y+1
          #  df



df1=pd.read_csv("Dataframe_1.csv")
df2=pd.read_csv("Dataframe_2.csv")
df3=pd.read_csv("Dataframe_3.csv")
df4=pd.read_csv("Dataframe_4.csv")
df5=pd.read_csv("Dataframe_5.csv")
df6=pd.read_csv("Dataframe_6.csv")
df7=pd.read_csv("Dataframe_7.csv")
df8=pd.read_csv("Dataframe_8.csv")
df9=pd.read_csv("Dataframe_9.csv")
df1_1 = pd.read_csv("Dataframe_10.csv")
df2_2 = pd.read_csv("Dataframe_11.csv")

array_tudo = np.zeros((147,3))
```

```
array_tudo2 = np.zeros((147,1))
df10 = pd.DataFrame()
df20 = pd.DataFrame()
df30 = pd.DataFrame()
df40 = pd.DataFrame()
df50 = pd.DataFrame()
df60 = pd.DataFrame()
df70 = pd.DataFrame()
df80 = pd.DataFrame()
df90 = pd.DataFrame()
df100 = pd.DataFrame()
df110 = pd.DataFrame()
df120 = pd.DataFrame()

for x in range(147):



df120= pd.concat([df[df.loc[:,"Num"] == x+1]], ignore_index=True).astype(float)
df10= pd.concat([df1[df1.loc[:,"X6"] == x+1]], ignore_index=True).astype(float)
df20= pd.concat([df2[df2.loc[:,"X6"] == x+1]], ignore_index=True).astype(float)
df30= pd.concat([df3[df3.loc[:,"X6"] == x+1]], ignore_index=True).astype(float)
df40= pd.concat([df4[df4.loc[:,"X6"] == x+1]], ignore_index=True).astype(float)
df50= pd.concat([df5[df5.loc[:,"X6"] == x+1]], ignore_index=True).astype(float)
df60= pd.concat([df6[df6.loc[:,"X6"] == x+1]], ignore_index=True).astype(float)
df70= pd.concat([df7[df7.loc[:,"X6"] == x+1]], ignore_index=True).astype(float)
df80= pd.concat([df8[df8.loc[:,"X6"] == x+1]], ignore_index=True).astype(float)
df90= pd.concat([df9[df9.loc[:,"X6"] == x+1]], ignore_index=True).astype(float)
df100= pd.concat([df1_1[df1_1.loc[:,"X6"] == x+1]], ignore_index=True)
.astype(float)
df110= pd.concat([df2_2[df2_2.loc[:,"X6"] == x+1]], ignore_index=True)
.astype(float)

#df111.iloc[len(df111)-12,7]

    if(array[x] == 0):
        array_tudo[x,0] = df10.iloc[len(df10)-12,7]
        array_tudo[x,1] = df10.iloc[len(df10)-12,4]

        if (df10.iloc[len(df10)-12,1]) < 0 :
            array_tudo[x,2] = df10.iloc[len(df10)-12,1]* (-2.4)

        else:
            array_tudo[x,2] = df10.iloc[len(df10)-12,1]

        if (df100.iloc[len(df100)-12,1] < 0):
            array_tudo2[x,0] = df100.iloc[len(df100)-12,1] * (-2.4)
        else:
             array_tudo2[x,0] = df100.iloc[len(df100)-12,1]
    elif(array[x] == 1):
        array_tudo[x,0] = df20.iloc[len(df20)-12,7]
        array_tudo[x,1] = df20.iloc[len(df20)-12,4]

        if (df20.iloc[len(df20)-12,1]) < 0 :
            array_tudo[x,2] = df20.iloc[len(df20)-12,1]* (-2.4)

        else:
            array_tudo[x,2] = df20.iloc[len(df20)-12,1]

        if (df100.iloc[len(df100)-12,1] < 0):
            array_tudo2[x,0] = df100.iloc[len(df100)-12,1] * (-2.4)
        else:
             array_tudo2[x,0] = df100.iloc[len(df100)-12,1]

    elif(array[x] == 2):
        array_tudo[x,0] = df30.iloc[len(df30)-12,7]
        array_tudo[x,1] = df30.iloc[len(df30)-12,4]

        if (df30.iloc[len(df30)-12,1]) < 0 :
            array_tudo[x,2] = df30.iloc[len(df30)-12,1]* (-2.4)

        else:
            array_tudo[x,2] = df30.iloc[len(df30)-12,1]
        if (df100.iloc[len(df100)-12,1] < 0):
            array_tudo2[x,0] = df100.iloc[len(df100)-12,1] * (-2.4)
        else:
             array_tudo2[x,0] = df100.iloc[len(df100)-12,1]

    elif(array[x] == 3):
        array_tudo[x,0] = df40.iloc[len(df40)-12,7]
        array_tudo[x,1] = df40.iloc[len(df40)-12,4]

        if (df40.iloc[len(df40)-12,1]) < 0 :
            array_tudo[x,2] = df40.iloc[len(df40)-12,1]* (-2.4)

        else:
            array_tudo[x,2] = df40.iloc[len(df40)-12,1]

        if (df100.iloc[len(df100)-12,1] < 0):
            array_tudo2[x,0] = df100.iloc[len(df100)-12,1] * (-2.4)
        else:
             array_tudo2[x,0] = df100.iloc[len(df100)-12,1]
    elif(array[x] == 4):
        array_tudo[x,0] = df50.iloc[len(df50)-12,7]
        array_tudo[x,1] = df50.iloc[len(df50)-12,4]

        if (df50.iloc[len(df50)-12,1]) < 0 :
            array_tudo[x,2] = df50.iloc[len(df50)-12,1]* (-2.4)

        else:
            array_tudo[x,2] = df50.iloc[len(df50)-12,1]
        if (df100.iloc[len(df100)-12,1] < 0):
            array_tudo2[x,0] = df100.iloc[len(df100)-12,1] * (-2.4)
        else:
             array_tudo2[x,0] = df100.iloc[len(df100)-12,1]

    elif(array[x] == 5):
        array_tudo[x,0] = df60.iloc[len(df60)-12,7]
        array_tudo[x,1] = df60.iloc[len(df60)-12,4]

        if (df60.iloc[len(df60)-12,1]) < 0 :
            array_tudo[x,2] = df60.iloc[len(df60)-12,1]* (-2.4)

        else:
            array_tudo[x,2] = df60.iloc[len(df60)-12,1]
        if (df100.iloc[len(df100)-12,1] < 0):
            array_tudo2[x,0] = df100.iloc[len(df100)-12,1] * (-2.4)
        else:
             array_tudo2[x,0] = df100.iloc[len(df100)-12,1]

    elif(array[x] == 6):
        array_tudo[x,0] = df70.iloc[len(df70)-12,7]
        array_tudo[x,1] = df70.iloc[len(df70)-12,4]

        if (df70.iloc[len(df70)-12,1]) < 0 :
            array_tudo[x,2] = df70.iloc[len(df70)-12,1]* (-2.4)

        else:
            array_tudo[x,2] = df70.iloc[len(df70)-12,1]

        if (df100.iloc[len(df100)-12,1] < 0):
            array_tudo2[x,0] = df100.iloc[len(df100)-12,1] * (-2.4)
        else:
             array_tudo2[x,0] = df100.iloc[len(df100)-12,1]

    elif(array[x] == 7):
        array_tudo[x,0] = df80.iloc[len(df80)-12,7]
        array_tudo[x,1] = df80.iloc[len(df80)-12,4]
        if (df80.iloc[len(df80)-12,1]) < 0 :
            array_tudo[x,2] = df80.iloc[len(df80)-12,1]* (-2.4)

        else:
            array_tudo[x,2] = df80.iloc[len(df80)-12,1]

        if (df100.iloc[len(df100)-12,1] < 0):
            array_tudo2[x,0] = df100.iloc[len(df100)-12,1] * (-2.4)
        else:
             array_tudo2[x,0] = df100.iloc[len(df100)-12,1]
    elif(array[x] == 8):
        array_tudo[x,0] = df90.iloc[len(df90)-12,7]
        array_tudo[x,1] = df90.iloc[len(df90)-12,4]

        if (df90.iloc[len(df90)-12,1]) < 0 :
            array_tudo[x,2] = df90.iloc[len(df90)-12,1]* (-2.4)

        else:
            array_tudo[x,2] = df90.iloc[len(df90)-12,1]
        if (df100.iloc[len(df100)-12,1] < 0):
            array_tudo2[x,0] = df100.iloc[len(df100)-12,1] * (-2.4)
        else:
             array_tudo2[x,0] = df100.iloc[len(df100)-12,1]
    elif(array[x] == 9):
        array_tudo[x,0] = df120.iloc[len(df120)-51,6]
        array_tudo[x,1] = df100.iloc[len(df100)-12,4]

        if (df100.iloc[len(df100)-12,1]) < 0 :
            array_tudo[x,2] = df100.iloc[len(df100)-12,1]* (-2.4)

        else:
            array_tudo[x,2] = df100.iloc[len(df100)-12,1]
        if (df100.iloc[len(df100)-12,1] < 0):
            array_tudo2[x,0] = df100.iloc[len(df100)-12,1] * (-2.4)
        else:
             array_tudo2[x,0] = df100.iloc[len(df100)-12,1]

    elif(array[x] == 10):
        array_tudo[x,0] = df110.iloc[len(df110)-12,7]
        array_tudo[x,1] = df110.iloc[len(df110)-12,4]

        if (df110.iloc[len(df110)-12,1]) < 0 :
            array_tudo[x,2] = df110.iloc[len(df110)-12,1]* (-2.4)

        else:
            array_tudo[x,2] = df110.iloc[len(df110)-12,1]

        if (df100.iloc[len(df100)-12,1] < 0):
            array_tudo2[x,0] = df100.iloc[len(df100)-12,1] * (-2.4)
        else:
             array_tudo2[x,0] = df100.iloc[len(df100)-12,1]

 tudo = [p0,p1,p2,p3,p4,p5,p6,p7,p8,p9,p10]
tudo =np.array(tudo)
comp = [0 for row in range(11)]
c = [0 for row in range(len(p0))]
```

```
for y in range(len(p0)):
    for x in range(11):
        comp[x] = tudo[x][y]
    comp=np.array(comp)
    c[y] = np.where(comp==comp.max())

a0=array.count(0)
a1=array.count(1)
a2=array.count(2)
a3=array.count(3)
a4=array.count(4)
a5=array.count(5)
a6=array.count(6)
a7=array.count(7)
a8=array.count(8)
a9=array.count(9)
a10=array.count(10)

df_f["Todos_10_Metodos"] = [a0,a1,a2,a3,a4,a5,a6,a7,a8,a9,a10]

df_f.to_csv("Resultados_10_métodos.csv")
```

APÊNDICE B – Programming codes in R

The main advantages of using the R language was the use of the *forecast* library where all the forecast functions used in this work were found. This code therefore simulates the forecast of each consumer unit for an initial period of 24 months of training up to the last possible consumed demand value of each UC. This code can take days to complete because the auto-ary function required a lot of processing to define the best ARIMA parameters for each time series.

```
#### Previsão dos 10 metodos para 150 UCs #####
#### Criado por Thyago Carvalho #####
library(forecast)
library(ggplot2)
library(xts)
library(readxl)
dados<-read_excel(file.choose(), sheet = 1)
#max(dados$DC12, na.rm = TRUE)
UC<-unique(dados$MATRICULA)
tUC<-length(UC)

df1 <- data.frame(matrix(ncol = 8, nrow = 0))
df2 <- data.frame(matrix(ncol = 8, nrow = 0))
df3 <- data.frame(matrix(ncol = 8, nrow = 0))
df4 <- data.frame(matrix(ncol = 8, nrow = 0))
df5 <- data.frame(matrix(ncol = 8, nrow = 0))
df6<- data.frame(matrix(ncol = 8, nrow = 0))
df7 <- data.frame(matrix(ncol = 8, nrow = 0))
df8 <- data.frame(matrix(ncol = 8, nrow = 0))
df9 <- data.frame(matrix(ncol = 8, nrow = 0))




a=c(0,0,0,0,0)

for(i in 1:tUC){
  x<-matrix(0,length(dados$DTR[dados$MATRICULA==UC[i]]),4)
  x[,1]<-as.numeric(dados$DTR[dados$MATRICULA==UC[i]])
  x[,2]<-as.numeric(dados$ANO[dados$MATRICULA==UC[i]])
```

```
x[,3]<-as.numeric(dados$MES[dados$MATRICULA==UC[i]])
x[,4]<-as.numeric(dados$DC12[dados$MATRICULA==UC[i]])



#tudo.ts= ts(x[1:25,1], start=c((x[1,2]),(x[1,3])),frequency=12)
tam_x<-length(x[,1])

for (t in 1:(tam_x-39)){ #reserva dados de 3 anos para começar a fazer previsões





  treino= ts(x[1:(25+t),1], start=c((x[1,2]),(x[1,3])),frequency=12)
  tudo_DC12.ts= ts(x[1:(38+t),4], start=c((x[1,2]),(x[1,3])),frequency=12)
  # DFP.ts tem todos os valores de treino
  #DTR.ts=window(tudo.ts, start=c((x[1,2]),(x[1,3])),end=c((x[tam_x-13,2]),
  (x[tam_x-13,3])),frequency=12) #declara que é uma serie temporal mensal f=12,
  anual=1
  # ultimos 12 valores da serie temporal
  #  teste_DTR.ts= window(tudo.ts, start=c((x[tam_x-12,2]),(x[tam_x-12,3]))
  ,end=c
  ((x[tam_x-1,2]),
  (x[tam_x-1,3])),
  frequency=12)



  #   tryCatch( {(arima =auto.arima(tudo.ts))  }
  #            , warning = function (w){arima =auto.arima(window(tudo.ts)
  ,lambda=0)}
  #           , error = function (e){arima = c(0,0,0,0,0,0,0,0,0,0,0,0)})

  valr=12
  #naive
  Mnaive = naive(treino, h=valr)
  #média
  Mmeanf = meanf(treino, h=valr)
  #drift
  Mrwf = rwf(treino, h=valr, drift = T)
  #holt - pesos
```

```
    Mholt = holt(treino, h=valr)
    #holt winter aditivo
    Mhw = hw(treino,seasonal = "additive", h=valr)
    #hold winter multiplicativo
    Mhw2 = hw(treino,seasonal = "multiplicative", h=valr)
    #holt winter multiplicativo amortecido
    Mhw3 = hw(treino,seasonal = "multiplicative", damped = T, phi = 0.9,h=valr)



    #linear
    Mtslm = tslm(treino ~ trend, data=treino)
    Mtslm = forecast(Mtslm, h=valr)

    #rede neural
    Mnnetar = nnetar(treino)
    Mnnetar = forecast(Mnnetar, h=valr)









fun_DC12 <- function(vetor12,inicio_dc12,fim_dc12){

      Dc12_previsao <- 0

      val<-c()
      for(h in inicio_dc12:fim_dc12){

        b<-c()
        for(j in 1:12){
          b[j]<-ifelse(vetor12[j]-h>0,(vetor12[j]-h)*2.5,(vetor12[j]-h)*(-1))
          # cria vetor com os saldos mensais onde "h" é a demanda teorica
        }
        val[h]= sum(b) #soma os saldos mensais
      }

      Dc12_previsao <- match(c( min(val,na.rm = TRUE)), val)
      print(Dc12_previsao)
      return(Dc12_previsao)
    }

for1 = fun_DC12(Mnaive$mean,floor(min(Mnaive$mean,na.rm = TRUE))
-1,ceiling(max(Mnaive$mean,na.rm = TRUE))+1  )
for2 = fun_DC12(Mmeanf$mean,floor(min(Mmeanf$mean,na.rm = TRUE))
-1,ceiling(max(Mmeanf$mean,na.rm = TRUE))+1  )
for3 = fun_DC12(Mrwf$mean,floor(min(Mrwf$mean,na.rm = TRUE))
-1,ceiling(max(Mrwf$mean,na.rm = TRUE))+1  )
for4 = fun_DC12(Mholt$mean,floor(min(
  ,na.rm = TRUE))
  -1,ceiling(max(Mholt$mean,na.rm = TRUE))+1  )
for5 = fun_DC12(Mhw$mean,floor(min(Mhw$mean,na.rm = TRUE))
-1,ceiling(max(Mhw$mean,na.rm = TRUE))+1  )
for6 = fun_DC12(Mhw2$mean,floor(min(Mhw2$mean,na.rm = TRUE))
-1,ceiling(max(Mhw$mean,na.rm = TRUE))+1  )
for7 = fun_DC12(Mhw3$mean,floor(min(Mhw3$mean,na.rm = TRUE))
-1,ceiling(max(Mhw3$mean,na.rm = TRUE))+1  )
for8 = fun_DC12(Mtslm$mean,floor(min(Mtslm$mean,na.rm = TRUE))
-1,ceiling(max(Mtslm$mean,na.rm = TRUE))+1  )
for9 = fun_DC12(Mnnetar$mean,floor(min(Mnnetar$mean,na.rm = TRUE))
-1,ceiling(max(Mnnetar$mean,na.rm = TRUE))+1  )


  erro_1= accuracy(tudo_DC12.ts[38+t],for1)
  erro_2= accuracy(tudo_DC12.ts[38+t],for2)
  erro_3= accuracy(tudo_DC12.ts[38+t],for3)
  erro_4= accuracy(tudo_DC12.ts[38+t],for4)
  erro_5= accuracy(tudo_DC12.ts[38+t],for5)
  erro_6= accuracy(tudo_DC12.ts[38+t],for6)
  erro_7= accuracy(tudo_DC12.ts[38+t],for7)
  erro_8= accuracy(tudo_DC12.ts[38+t],for8)
  erro_9= accuracy(tudo_DC12.ts[38+t],for9)

    #Salvado valores dos erros em dois dataframes
  df1[nrow(df1) + 1,]<-rbind(append(append(append(erro_1,i),t),for1))
  df2[nrow(df2) + 1,]<-rbind(append(append(append(erro_2,i),t),for2))
  df3[nrow(df3) + 1,]<-rbind(append(append(append(erro_3,i),t),for3))
  df4[nrow(df4) + 1,]<-rbind(append(append(append(erro_4,i),t),for4))
  df5[nrow(df5) + 1,]<-rbind(append(append(append(erro_5,i),t),for5))
  df6[nrow(df6) + 1,]<-rbind(append(append(append(erro_6,i),t),for6))
  df7[nrow(df7) + 1,]<-rbind(append(append(append(erro_7,i),t),for7))
  df8[nrow(df8) + 1,]<-rbind(append(append(append(erro_8,i),t),for8))
  df9[nrow(df9) + 1,]<-rbind(append(append(append(erro_9,i),t),for9))

    #### erro_DC12 e erro_DTR estavam vindo com Nan solucao rapida aplicada
    mas n mais eficiente
    cat("----- UC: ",i," -----\n")
       #plot
# plot(arima)
    #  data = data.frame(arima, arima2)
    #  qplot(time2, data, color=colors, scale_colour_identity(guide="legend"),
    #      geom=c("line", "line"))
    # lines(DC12.ts - arima2\$resid,col="red")

    if (t == 1200) {

      break

    }

  }

  if (t == 1200) {

    break

  }
}
df7
```