ProfARIMA: a profit-driven order identification algorithm for ARIMA models in sales forecasting

Tine Van Calster[a,*], Bart Baesens[a], Wilfried Lemahieu[a]

[a]*Faculty of Economics and Business, KU Leuven, Naamsestraat 69, 3000 Leuven, Belgium*

Abstract

In forecasting, evolutionary algorithms are often linked to existing forecasting methods to optimize their input parameters. Traditionally, the fitness function of these search heuristics is based on an accuracy measure. In this paper, however, we combine forecasting accuracy with business expertise by defining a flexible and easily interpretable profit function for sales forecasting, which is based on the profit margin of a given product, the volume of its sales and the accuracy of the forecast. ProfARIMA is a new procedure that selects the lags of a Seasonal ARIMA model according to the profit of a model's forecasts by taking advantage of search heuristics. This procedure is tested on both publicly available datasets and a real-life application with datasets of The Coca-Cola Company in order to assess its performance, both in profit and accuracy. Three different evolutionary algorithms were implemented during this testing process, i.e. Genetic Algorithms, Particle Swarm Optimization and Simulated Annealing. The results indicate that ProfARIMA always performs at least equally to the Box-Jenkins methodology and often outperforms this traditional procedure. For The Coca-Cola Company, our new algorithm in combination with Genetic Algorithms even leads to a significantly larger profit for out-of-sample forecasts.

Keywords: Sales forecasting, Seasonal ARIMA, Order identification, Genetic algorithms, Particle Swarm Optimization, Simulated Annealing

*Corresponding author

Email address: tine.vancalster@kuleuven.be (Tine Van Calster)

1. Introduction

In forecasting, many biologically inspired algorithms have been used to improve the accuracy of both linear and non-linear forecasting methods for different applications, including sales forecasting [16, 22]. Especially neural networks have become extremely popular in many forecasting applications, such as load forecasting [5, 19, 23, 38, 54], as they perform well on non-linear and high-frequency time series. Many applications combine these evolutionary algorithms with traditional forecasting techniques in order to capture both linear and non-linear time series information [2, 6, 13, 61], or use the same algorithms to optimize the input parameters of existing forecasting methods. In this introduction, we will focus on this last group, as parameter optimization is exactly the problem setting that this paper deals with. The real-life use case in this paper consists of a sales forecast for a range of products of The Coca-Cola Company. In sales forecasting, classical time series modeling has been popular in business applications for many decades. Especially exponential smoothing models, such as the Holt-Winters methodology, and Autoregressive Integrated Moving Average models (ARIMA) are regarded as go-to techniques for any application and are therefore often used as benchmarks for new methodologies as well [50]. We turn to traditional Seasonal ARIMA models for the application in this paper, because of their interpretability, their simplicity and their popularity within business settings. These are three key aspects in any business application, as these enhance the trust in the methodology and increase the maintainability of the final model. However, an important issue with ARIMA modeling, as with many forecasting techniques, is the expert knowledge that is necessary to select the input parameters, as this usually, at least partly, involves a trial-and-error process. Automated order identification with evolutionary search heuristics seems to be an answer to this problem.

Parameter optimization significantly influences the performance of all forecasting techniques, from ARIMA models [49] to neural networks [62], and from Support Vector Regression [14] to GARCH models [30]. The forecast accuracy of these models has been improved by optimizing their input with evolutionary search heuristics, such as Particle Swarm Optimization [4, 32, 60, 62, 63], Genetic Algorithms [22, 30, 39, 42, 43, 49, 54], Simulated Annealing [23, 40], Artificial Bee Colony Algorithm [5, 24, 47], Differential Evolution [25, 57] and Fruit Fly Optimization [38, 41]. These hybrid methodologies have been applied to many different fields in forecasting, including tourism flow forecasting [14], electricity demand forecasting [63], rainfall prediction [60], price forecasting [47] and many others. The literature on evolutionary algorithms for AR(I)MA modeling mainly focuses on Genetic Algorithms [1, 21, 29, 45, 49], with a few exceptions, such as [26]. Most of these papers deal with the correctness of the identification by turning to simulated or real-life datasets of which the number of lags is known [1, 45, 49]. This paper, however, aims to assess the performance of the proposed methodology on several real-life datasets, which greatly adds to the strength of the conclusions.

However, the true novelty of this paper consists in the use of an entirely different fitness function when optimizing the input parameters. All evolutionary algorithms require a fixed fitness function that needs to be maximized or minimized in order to select the optimal solution to a given problem. In parameter optimization for forecasting models, several typical fitness functions exist that are either based on the goodness of fit of the model or the accuracy of the forecast on a validation set. The first group is mainly popular in AR(I)MA settings, where the Akaike's Information Criterion (AIC) [9, 45] and the Bayesian Information Criterion (BIC) [21, 49] play an important role in order identification. Especially the AIC value is often used when manually selecting the parameters of an AR(I)MA model [19, 65]. However, the most popular criterion for optimization in forecasting is accuracy, which can take many forms, such as the Mean Squared Error (MSE) [1, 4, 5, 30, 34, 54, 62], the Mean Absolute Percentage Error (MAPE) [14, 23, 38, 39, 42, 47, 57, 63] or the Root Mean Squared Error (RMSE) [22, 25, 27, 38]. In this paper, however, we turn to a profit measure for sales forecasting to optimize the order identification of Seasonal ARIMA models. This measure combines the aspect of forecast accuracy with expert knowledge on the profit margins of a set of products. By constructing an easily interpretable and flexible profit measure that incorporates business expertise, we increase the usability and credibility of the model in business contexts. Optimization according to accuracy and goodness-of-fit measures does not necessarily lead to a higher profit, even though both evaluation criteria do play a role in any profit calculation. This profit measure is therefore based on two components that are fundamentally important to the business side of sales predictions. Firstly, profit is always driven by the amount of sales, but also by the profit margin of a given product. Companies usually sell an array of products that each have their own value, so the profit function should be able to capture this variation. This is where expert knowledge comes in to adapt the profit function to a specific product by setting the size of this profit margin. Secondly, a forecast should still be as accurate as possible, so it will not result in profit loss by, e.g., unforeseen production costs, storage costs or a loss of sales. The proposed profit function therefore also aims to achieve an accurate forecast. By penalizing inaccuracy, we incorporate the impact of the forecasting error on the profit of a product.

The use of profit measures in model evaluation has already arisen in other predictive applications, such as [55, 56], and revenue forecasting remains a hot topic in many businesses [37, 58]. Profit therefore seems to be an ideal fitness function to drive model selection, as it combines the business perspective with the traditional research goal of an accurate forecast. This paper aims to test this by looking into the performance of search heuristics for ARIMA order identification in a real-life business setting. Concretely, we perform a sales forecast for two publicly available data sets, as well as a real life data set from the Coca-Cola Company, which consist of 50 different test cases in total. Furthermore, we implement this profit fitness function in three different search heuristics, i.e. Genetic Algorithms, Particle Swarm Optimization and Simulated Annealing. Genetic Algorithms were chosen because of its history in ARIMA modeling, while Particle Swarm Optimization and Simulated Annealing were selected because of their popularity and ease of implementation. By combining a profit-driven search heuristic for model identification with Maximum Likelihood Estimation for parameter estimation, we define the hybrid ProfARIMA procedure. The performance of this model is then compared to a Seasonal ARIMA model that was estimated according to the traditional Box-Jenkins procedure.

The paper will start with an overview of relevant theoretical background concerning the techniques that were used during the testing process. Section 3 then explains the exact methodology of the ProfARIMA procedure with its unique fitness function. Finally, we apply the technique to the two publicly available and the real-life use case of The Coca-Cola Company in order to assess its performance.

2. Theoretical background

2.1. ARIMA modeling

Autoregressive Integrated Moving Average (ARIMA) models constitute one of the most popular time series techniques to tackle univariate forecasting problems. Because of its interpretability, speed and accuracy, the technique remains one of the go-to methods in many business applications. ARIMA models are generalizations of ARMA models, which were first described by Whitle [59], but were more widely distributed by Box and Jenkins in the 1970's [12]. They consist of three components, which define the $ARIMA(p, d, q)$ model with p autoregressive terms, q moving average terms and d differencing terms. In this paper, we will solely deal with the seasonal variant of ARIMA models, the $SARIMA(p, d, q)(P, D, Q)_s$ model, which includes three additional seasonal variants of all aforementioned terms [12]. The formula for a $SARIMA(1, 1, 1)(1, 1, 1)_{12}$ model, for example, is given in equation 1. Y_t and ε_t respectively stand for the forecast at time t and the forecast error at time t, while ϕ_i represents the weight of the given parameter, of which the estimation is explained in the next paragraph.

$$(Y_t - \phi_1 Y_{t-1})(Y_t - \phi_2 Y_{t-12})(Y_t - Y_{t-1})(Y_t - Y_{t-12}) = (\varepsilon_t + \phi_3 \varepsilon_{t-1})(\varepsilon_t + \phi_4 \varepsilon_{t-12}) \quad (1)$$

For each component, the number of terms needs to be identified and the weights of the AR and MA terms need to be estimated. Firstly, the two differencing terms need to be considered, as the correct identification of the other terms can only be achieved for a stationary time series. The addition of both trend and seasonal differencing is typically decided on by performing unit root tests, such as the Augmented Dickey-Fuller test [15] and the Osborn-Chui-Smith-Birchenhall test [51] respectively. Further model identification can then be done by utilizing expert knowledge or by comparing models using several evaluation criteria. The latter can, for example, consist of Akaike's Information Criterion, which both checks for goodness of fit and model simplicity [3], or choosing the model that had the lowest forecasting error in a pre-defined testing period. Lastly, the weights of every AR and MA term, both non-seasonal and seasonal, need to be estimated as well. Typically, this optimization is achieved by minimizing the sum of the squared errors (SSE). This error is calculated by taking the difference between the actual value at time t and the forecasted value at time t when only $t-1$ was known. We therefore look to an in-sample error for this estimation:

$$SSE = \sum_{t=1}^{T} (Y_t - \hat{Y}_{t|t-1})^2 = \sum_{t=1}^{T} e_t^2 \quad (2)$$

The processes of model identification and weight estimation are repeated until the model satisfies the selected evaluation criterion/criteria. Finally, the model is ready to forecast a chosen amount of time periods in the future.

The parameter optimization of AR(I)MA models therefore consists of both order identification and parameter estimation. Both problems can be tackled by search heuristics, where authors can either treat them separately [29, 49] or deal with them at the same time [1]. While some dismiss the parameter estimation by genetic algorithms as too complex compared to traditional estimation methods, such as Maximum Likelihood or Least Squared Error [8], others actually prefer it for solving the same problem [29]. Most authors emphasize one of the two topics, while some authors explicitly try to combine them into one framework [1, 45]. In this paper, we opted to leave parameter estimation out of our methodology, as traditional techniques provide a fast and accurate way of assessing the correct weight of each parameter. Furthermore, by combining both problems, the scalability of the search space quickly becomes an issue, especially in real-life applications. Therefore, we preferred slightly enlarging the search space by taking a larger number of possible lags into account, to limiting this amount because of the added complexity of the parameter estimation.

2.2. Iterated F-racing

Before we turn to the explanation of the search heuristics that were considered in this paper, it is important to note that all of these methods have input parameters themselves. Therefore, the parameters of each evolutionary algorithm had to be optimized (or tuned) themselves before they were implemented as parameter optimization tools. There are several methodologies for selecting the best set-up of an evolutionary algorithm, of which [18] and [46] give a nice overview.

In this paper, we turn to the concept of Iterated F-racing for parameter tuning, which was first mentioned in [7] and was further extended in [10]. This methodology is a racing method, which essentially compares different combinations of parameters to one another as if they were in a race [46]. During this process, all possible solutions are ranked and the worst performing ones are eliminated once it can be statistically proven that the other solutions are significantly better [18]. In F-racing, this ranking is performed by the Friedman test in order to determine if there is a significant difference between the solutions, followed by post-hoc analyses that select the combinations that can be eliminated [10]. Iterative F-racing further improves the process by first defining a probability measure over the parameter space at each iteration that is acquired by selecting the best combinations from the previous iteration. The algorithm then generates new possible solutions based on the best solutions from the previous round and applies F-race on these new parameter combinations [7]. The best parameter combinations over all test cases are given as an output.

2.3. Genetic algorithms

Genetic algorithms (GA) are a form of evolutionary computation that was first described in 1975 by Holland [28]. The idea behind these algorithms is to mimic nature in its evolution from generation to generation in order to find optimal solutions to given problems. Essentially, an original population of possible solutions is allowed to 'reproduce' and create new solutions. The survival of these solutions depends on a fitness criterion that is decided beforehand, so the best solution will ultimately be chosen by the algorithm after a number of iterations [8, 64]. This general procedure of GA is formalized in Figure 1.

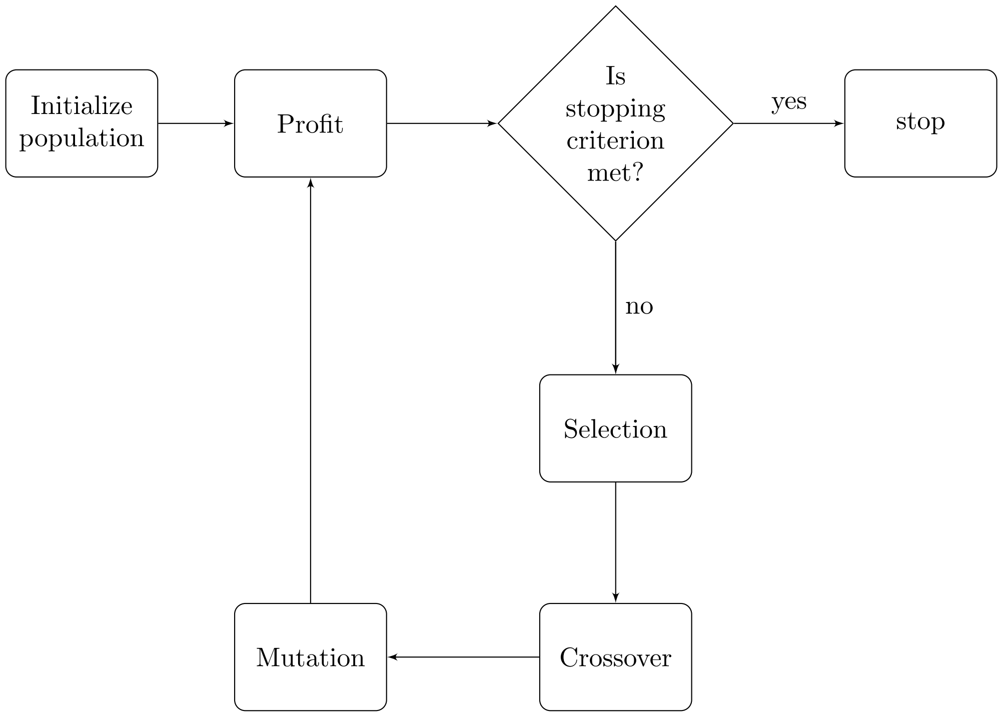

Figure 1: The Genetic Algorithm process

Before we can initialize the population, the solutions and their features need to be encoded in the chromosome, which is usually formalized in binary or decimal string values [64]. Once the possible solutions are encoded, we can choose an initial population of solutions, which often consists of a random selection process that can be guided by pre-set rules. These initial solutions are immediately evaluated by calculating their fitness, which is done by the predetermined fitness function. The algorithm then decides which chromosomes to select as the 'parents' of the next generation. According to the principle of the 'survival of the fittest', the fittest chromosomes should be more likely to reproduce than the less fitter ones. Therefore, a selection procedure should assign a larger probability of being selected to the stronger chromosomes. Once these probabilities are calculated, the crossover step can commence, where pairs of chromosomes are selected to produce children that form combinations of their genes. The probability of crossover has to be determined beforehand and usually lies between 0.5 and 1. Additionally, there are three common methods for crossovers to choose from: one-point crossover, multiple-point crossover and uniform crossover [1]. Next, the chromosomes can possibly undergo the process of mutation, according to a predetermined probability as well, albeit a small probability. Mutation occurs when, for example, a binary 0 turns into a 1 or vice versa, but examples of this type of evolution are rare in nature and therefore also rare in the algorithm [8]. After this last genetic operator, the chromosomes are evaluated once more and the best ones are selected to maintain a population. This process iterates until a stopping criterion is met.

The steps in Figure 1 require the selection of a number of factors that ultimately define the GA and that need to be tuned before applying the search heuristic. Concretely, we need to determine the chromosome encoding, the initial population size, the initial population selection, the fitness function, the distribution of the selection probabilities, the probabilities for crossover and mutation, the crossover method, the elitist strategy and the stopping criterion in order to execute the technique [64].

2.4. Particle Swarm optimization

In 1995, Kennedy and Eberhart introduced a new search heuristic called Particle Swarm Optimization (PSO) [35]. In this paper, we turn to Standard PSO (SPSO), which was created by [52]. The algorithm is inspired by the behaviour of a flock of birds that can fly together, scatter suddenly and regroup again. Each bird or particle not only keeps track of its own behaviour, but of the behaviour of others as well [64]. Each particle knows four elements: its current location, its current velocity, its best location so far and the global best location so far, which takes the entire swarm into consideration. This memory aspect is the main difference with GA, next to the ability of particles to move in completely different directions than other solutions. As chromosomes in GA always share some information, the group of solutions gradually moves towards an optimum together [17]. In PSO, the particles are aware of the best solution so far, together with their own best solution, which both influence the next position of the particle. The particles therefore typically converge faster than GA, but might be trapped in a local optimum more easily as well [11].

In order to find the optimal solution, PSO uses a similar procedure as GA [64]. The initial swarm is randomly selected, but the swarm size still needs to be determined beforehand. Each particle consists of a location and velocity, which are both randomly initialized as a first step. A fitness value is calculated at each iteration, which is based on the profit function. Once the fitness values of all particles are calculated, the particle best value and the global best value are updated. These values are needed to update the location and the velocity of each particle. These updates at time *t + 1* are done for each particle i by Equation 3 and Equation 4 for location x and velocity v respectively [52, 64].

$$x^i_{t+1} = x^i_t + v^i_{t+1} \tag{3}$$

$$v^i_{t+1} = \omega v^i_t + \varphi_1 rand_1(p^i_t - x^i_t) + \varphi_2 rand_2(g_t - x^i_t) \tag{4}$$

where $0 < \omega < 1$ is the inertia coefficient, $\varphi_1 > 0$ and $\varphi_2 > 0$ are the acceleration coefficients, $rand_1$ and $rand_2 \sim U(0,1)$, p^i_t is the personal best location of particle i at time t and g_t is the global best location of the entire swarm at time t. The velocity update equation consists of three parts: the momentum component, the cognitive component and the social component [11]. The three coefficients ω, φ_1 and φ_2 influence these three sides of the behaviour of the particles. It is therefore crucial that they are tuned properly. The procedure of PSO keeps iterating over these steps until a certain stopping criterion is achieved.

2.5. Simulated Annealing

The final search heuristic that we consider in this paper, is Simulated Annealing (SA), which was created by Kirkpatrick, Gelatt and Vecchi [36].

The authors were inspired by the Metropolis procedure [44] and realised this algorithm could be used for general optimization purposes instead of being limited to thermodynamic systems alone. During SA, the temperature is slowly decreased, which translates to a lower probability of accepting bad solutions in the optimization algorithm [64]. Essentially, temperature is a control parameter that allows for exploration of the solution space when it is high, but limits this freedom when it is low. The entire procedure of SA again starts with a random solution, which is immediately evaluated according to a fitness function. The algorithm then jumps to a neighbouring state, which is also generated randomly, and assesses the probability of changing to this state or remaining in the current one by comparing the fitness values of the two candidates. This acceptance probability is heavily influenced by the temperature. If the new solution has a better fitness value than the current one, the new solution is accepted. If, however, the new solution is worse, it still has a chance of getting accepted. This acceptance depends on the temperature, see equation 5, and a randomly generated number.

$$P_t(accept(n)) = e^{\frac{f(o)-f(n)}{t}} \tag{5}$$

With o as the original solution, n as the new solution, f as the function to minimize and t as the current temperature. This equation only holds if the new solution is worse than the original one. If the acquired probability is larger than the randomly generated number, the worse solution is accepted as the new state. This enables the algorithm to make large jumps in the search space. However, the acceptance rate of bad solutions decreases as the temperature decreases as well. Temperature is therefore an essential parameter to tune correctly. The procedure of SA iterates until a predefined stopping criterion has been reached.

3. Methodology

3.1. Datasets

In this paper, we consider two publicly available datasets and one company-specific dataset from The Coca-Cola Company. As the profit function focuses on the difference in profit between product categories, all datasets consist of multiple types of products. This section provides a short overview of the characteristics of each dataset.

The first dataset consists of the US monthly sales of petroleum and related products from January 1971 until December 1991. This dataset is publicly available in the Time Series Data Library on DataMarket[1]. Four different categories are defined: chemicals, coal, petrol and vehicles. Each of these time series has different properties concerning volume, season and trend. As the necessary profit information was not available, see section 3.3.1, the β weights for this dataset were randomly set at $1, 3, 2$ and 5 for the product categories in the order that they were mentioned above.

The second dataset contains the monthly sales of Australian wine from January 1980 until July 1995. It consists of six product categories: dry white, fortified, red, rose, sparkling and sweet wine. This dataset is also available in the Time Series Data Library on DataMarket[2]. The product categories again differ in terms of volume, season and trend. In this case, the β factors needed to be determined randomly as well. For the six categories in the same order as above, the weights are $2, 6, 4, 1, 2$ and 1 respectively.

The real-life use case of this paper consists of monthly sales datasets from The Coca-Cola Company, which consist of five countries with completely different product ranges and market characteristics. While some countries are examples of relatively stable and therefore more easily predictable markets, the others represent more irregular markets, where the time series fluctuate strongly from year to year. Each country then consists of eight separate product categories, which differ a great deal from one another in both absolute sales volume and profitability. As a result, a total of 40 unique instances constitute the testing process. Each time series includes sales from January 2005 until September 2015. The β factors for all of these datasets were estimated by business experts of The Coca-Cola Company.

3.2. Meta-heuristic parameter tuning

While all the search heuristics mentioned above will be used for parameter optimization, their own input parameters need to be estimated as well, according to the given dataset. In this section, the results of the Iterated F-racing methodology for parameter tuning are presented for the three chosen search heuristics. This optimization was performed for each of the three datasets: US petroleum, Australian wine and The Coca-Cola Company. The

[1]https://datamarket.com/data/list/?q=provider:tsdl
[2]https://datamarket.com/data/list/?q=provider:tsdl

results of the parameter tuning for each of the search heuristics in the next paragraph were obtained by taking a random sample out of the given dataset and then optimizing on a validation set. Therefore, the input parameters of the search algorithms were also optimized according to the profit fitness function, which will be thoroughly explained in the next sections.

3.2.1. Genetic Algorithms

As an ARIMA model is most easily encoded as a binary string, this representation was selected beforehand [8], as well as the unique fitness function that consists of a profit function. As the stopping criteria, we chose a maximum number of iterations, i.e. 150 generations, and a cut-off of 25 generations if there is no improvement in the best solutions. The elitist strategy keeps the best ten percent of the solutions after each iteration. All other input parameters were tuned according to the Iterated F-racing procedure, of which the result can be found in Table 1.

	Petroleum	**Wine**	**Coca-Cola**
Population size	20	20	30
Initial population	Random	Random	Random
Selection probabilities	Linear-rank	Roulette wheel	Roulette wheel
Crossover probability	0.8	0.7	0.7
Crossover method	Single point	Single point	Single point
Mutation probability	0.15	0.15	0.1

Table 1: Genetic Algorithm parameter tuning

3.2.2. Particle Swarm Optimization

Binary encoding is not needed for PSO, but it is important to note the value of the fitness function is multiplied by -1, as PSO searches for a global minimum. This fitness function again consists of the profit function that is explained in the next section. Furthermore, the stopping criteria for this algorithm also consist of a maximum number of iterations, i.e. 1000, and no improvement in the best solution for 100 generations. This number is higher than the GA configuration as PSO is meant for real-valued optimization and the search space of ARIMA order identification is a discrete problem. The final solution was therefore mapped to the discrete search space after optimization by simply rounding the numbers of the best solution. Thus, because the continuous search space was much larger, we opted for a higher number of iterations as a stopping criterion. The Iterated F-racing procedure tuned the swarm size, together with the acceleration coefficients and the inertia coefficient. The results thereof can be found in Table 2.

	Petroleum	**Wine**	**Coca-Cola**
Swarm size	20	30	30
Inertia coefficient	0.7988	0.2719	0.5363
Acceleration coefficient 1	1.9318	1.8761	1.8565
Acceleration coefficient 2	1.5382	1.4757	1.7829

Table 2: Particle Swarm Optimization parameter tuning

3.2.3. Simulated Annealing

For this paper, we turn to an extended version of SA, called Generalized Simulated Annealing [53], which combines classical and fast SA. This combination leads to a faster convergence than the classical format and has been tested on many applications [53]. We tuned most input parameters according to the extensive analyses in the aforementioned paper. However, the initial temperature of the algorithm was estimated by the Iterated F-racing procedure. All final parameter settings can be found in Table 3. The stopping criteria of the SA algorithm are a maximum number of iterations, i.e. 1000, and no improvement after 100 iterations, as SA also implements real-valued optimization. Because SA also searches for a global minimum, the outcome of the fitness function was multiplied by -1 as well.

	Petroleum	**Wine**	**Coca-Cola**
Initial temperature	4000	4000	3000
Visiting temperature	2.7	2.7	2.7
Acceptance temperature	-5	-5	-5

Table 3: Simulated Annealing parameter tuning

3.3. ProfARIMA

The proposed new algorithm, ProfARIMA, focuses on the model identification of an ARIMA model and leaves the weight estimation to the traditional method of Maximum Likelihood Estimation. However, instead of implementing a variation on a given accuracy measure, we aim to incorporate the profit of a forecast as the fitness function of the search heuristic. It is, therefore, a new hybrid technique that sets the number of lags for Seasonal ARIMA models by maximizing the expected profit. In this section, we will firstly discuss the profit measure that we have defined, before fully explaining the general procedure of the algorithm.

3.3.1. Profit measure

The fitness function of the genetic algorithm consists of an estimation of the expected profit in a sales forecast. This indication of profit is an easily interpretable formula that is based on both the volume of a certain product's sales and the accuracy of the forecast. As the literature has widely discussed in the past, both over- and under-forecasting lead to a list of potential costs, ranging from a loss in sales or overstock costs to shipment or storage costs [31, 33]. It is difficult to put an exact number on the profit loss that is contributed to each of these cost categories for any company. Additionally, not every company has the same costs that are linked to their product sales nor the same business strategy with regards to a preference for either an under- or an over-forecast. Therefore, our profit-driven genetic algorithm aims to find the most profitable model that stays within the realistic boundaries of sales by using a general formula that is penalized by the forecast error. In order to adjust the formula to a specific business, some input is required from business experts in the form of one weight per product, which indicates its profit margin, and an accuracy margin for penalization, which influence the profit according to the cost of forecasting errors. The specifics of this profit measure will be thoroughly described in the next paragraphs.

Firstly, the accuracy measure that will be used in the profit function is the Percentage Error, which is defined in equation 6. This evaluation measure takes into account the error over the entire forecast period m and therefore allows the effects of monthly under- and over-forecasting to cancel each other out over this period of time. This error metric was chosen because certain costs are not calculated at the same rate as the forecast frequency, but over a longer period of time, such as a quarter. For example, in the practical application of this paper, the sales volumes per quarter are considered to be more useful than monthly sales, as the growth of sales is publicly communicated for each quarter by The Coca-Cola Company. This motivated us to select a less strict accuracy metric that is more closely linked to the business goal at hand. However, when performing the same experiments with a profit that is based on MAPE, the exact same models were selected. Naturally, models that forecast results with a low MAPE automatically produce forecasts with a low PE as well. In terms of the profit defined, however, the PE was more interesting for the business context of this paper and was therefore ultimately selected as our primary evaluation metric, next to the profit itself.

$$PE = \frac{\sum_{i=1}^{m} Forecast_i - \sum_{i=1}^{m} Actuals_i}{\sum_{i=1}^{m} Actuals_i} * 100 \tag{6}$$

The complete calculation of the profit measure can be found in the following stack formula 7. As we can immediately notice, the profit is only penalized when the forecast error exceeds the 1% mark. These boundaries can naturally also be adjusted, but we have determined, in agreement with the business experts for this project, that any error larger than 1% in both directions, starts having a significant effect on the profit for our use case. This margin can be set accordingly for any sales forecast, where the option of simply setting it to zero makes sure that any forecasting error influences the profit. Furthermore, if over- or under-forecasting is worse for a specific case, this margin can also be set unequally in order to integrate this difference into the profit calculation. Therefore, these boundaries also incorporate expert knowledge into the profit function, which makes it more flexible.

$$Profit = \begin{cases} ((1 - (\alpha * |PE|)) * (\beta_{cat} * Volume_{cat} & PE > 1 \text{ or } PE < -1 \\ \beta_{cat} * Volume_{cat} & otherwise \end{cases} \tag{7}$$

As we mentioned above, the β factor should be set by business experts, while we chose to determine α by performing a sensitivity analysis on a sample of all three datasets. β refers to the weight that indicates the expected profit per volume unit of the product in question. Naturally, every product of a company has a different profit margin associated with it. This profit margin can be expressed relatively by comparing different products to one another, but can also take on an absolute form in a currency of the company's choosing. The α weight of the equation, however, is more difficult to determine by business experts, as it requires them to precisely estimate the loss of profit due to inaccuracy. Therefore, in order to establish the value for this factor that leads to the best results, we turned to a sensitivity analysis on a sample of datasets. By setting this weight with this methodology, we ensure that the model with the lowest PE for the validation set is found.

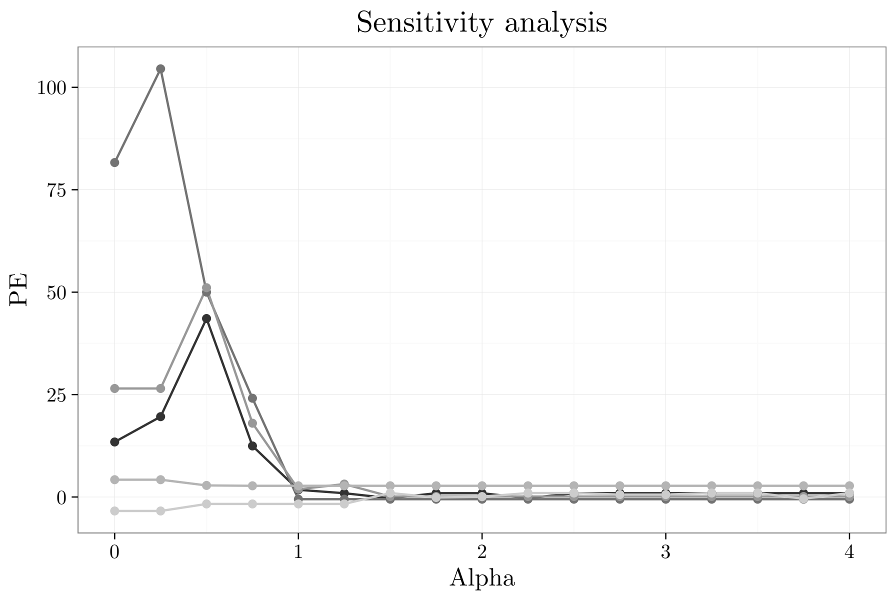

Figure 2: Sensitivity analysis of weight α

We performed this sensitivity analysis on our datasets by calculating the PE of the chosen model as a response variable to the value of α. We let α range from 0% to 4% with 0.25% intervals and took a random sample of five datasets to test the profit measure on, in which all three data sources are represented. It is important to note that the response variable is the PE of the validation sets and not the PE of the test sets. We therefore expect this response variable to be higher for the smallest values of α, as over-forecasted models will be preferred because of their higher sales volume, which is linearly linked to the final profit. We are therefore interested in discovering the point that penalizes highly over-forecasted models enough, but still gives an accurate representation of the expected profit. Figure 2 shows the results of the sensitivity analysis by means of a scatterplot. As we can clearly observe, any penalization that is smaller than 1% of error, leads to highly over-forecasted models. However, once this mark has been passed, only very accurate models seem to be chosen by the GA. In order to ensure that our technique selects the optimal model in as many cases as possible, we have decided to fix the α weight to 1.5% for this project. In Figure 2, the results of the sensitivity analysis show that the results become stable at this point.

A final remark on this profit measure should be made with regards to the choice of setting equal penalizations for under- and over-forecasting. As the higher sales volumes for an over-forecast will result in a higher profit even with these equal weights, the algorithm still shows a slight preference for over-forecasts. We have decided not to correct this imbalance, because a slight over-forecast was preferred to an under-forecast in our particular project. However, by ensuring that the penalization on the accuracy is large enough, the model will still stay clear of high over-forecasts. Additionally, the lowest possible profit is zero for all cases, as large forecast percentage errors for very small volume categories led to disproportionate profit losses in our case. This motivated us to set the lower boundary of the profit at zero, as we are mainly interested in whether or not a forecast is profitable, and in the extent of this profit.

3.3.2. The ProfARIMA procedure

In order to correctly apply the ProfARIMA procedure, we limit the search space to a maximum of five AR terms, five MA terms, two SAR terms and two SMA terms. For GA, the solutions are binary encoded chromosomes that consist of 14 binary bits. PSO and SA simply take a vector of four real numbers as their input. Subset ARIMA models are not considered in this instance, so when a larger AR term is selected and a smaller one is not, the final model will contain all AR terms until the largest one. Additionally, both the trend and seasonal differencing parameters are taken care of by performing unit root tests beforehand [49], i.e. the Augmented Dickey-Fuller test [15] and the Osborn-Chui-Smith-Birchenhall test [51]. The following steps form the procedure of the hybrid model, which is adjusted for a hold-out sample test process.

Step 1 Split data into training, validation and test sets. The given time series dataset should be split into a training set and a test set for evaluation. The validation set is then taken from the end of the training set and consists of a substantial time span, such as one year in this paper's application. This validation set will then be used to calculate the profit in the chosen optimization algorithm.

Step 2 Optimize Seasonal ARIMA order by executing the chosen optimization algorithm. The input parameters and procedures of GA, PSO and SA were optimized themselves by applying Iterated F-racing (see sections above). The fitness function remains the same across all three search methods: the profit measure as defined above.

Step 3 Forecast and evaluate. The model with the highest expected profit on the validation set is then applied to forecast and evaluate on the test set. This evaluation consists of several known accuracy metrics next to the new profit measure, as explained in section 3.4.

3.4. Evaluation

In order to assess the accuracy of our forecasts, we include three known evaluation measures. The first one is the Percentage Error, as explained in the section above, which constitutes the main accuracy assessment criterion for this project. Secondly, we also include the Mean Absolute Percentage Error and the Root Mean Squared Error, of which the definitions are displayed in equations 8 and 9.

$$MAPE = \frac{1}{n}\sum_{t=1}^{n} |\frac{Actual_t - Forecast_t}{Actual_t}| * 100 \tag{8}$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{t=1}^{n}(Actual_t - Forecast_t)^2} \tag{9}$$

The evaluation of ProfARIMA is compared to the traditional Box-Jenkins method for setting the lags of Seasonal ARIMA models [12]. This well-known approach to ARIMA modeling consists of three stages that should lead to the most suitable model for a given time series. Firstly, all parameters need to be set, including trend and seasonal differencing. Therefore, stationarity testing is usually the first step in the process by means of unit root tests. Next, the AR and MA parameters can be decided on by taking the autocorrelation function and the partial autocorrelation function into consideration. The second step estimates the weights of these parameters by means of universal algorithms, such as Maximum Likelihood Estimation. Finally, model diagnostics test the goodness of fit of the model by looking at information criteria, such as the AIC, and the significance of the residuals. When these results are satisfactory, the selected ARIMA model is used to forecast and we then apply the same evaluation measures as for the ProfARIMA algorithm.

4. Applications

The applications of this paper consist of forecasts for two public datasets and a sales forecast for the Coca-Cola Company, which is executed by country and by product category. In this section, we will discuss the general experimental set-up for all of these use cases and we will discuss the results of the forecasts separately.

4.1. Experimental set-up

The general experimental set-up starts with splitting up the dataset into training, validation and test sets. Firstly, the test sets were defined. For the US petroleum datasets, we will predict all quarters of 1991, while for the Australian wine datasets, we do the same for 1994. For The Coca-Cola Company use case, the test set consists of the first three quarters of 2015. Secondly, we decided that the validation set always includes the entire year before the period that we are forecasting. In this way, we try to ensure that the model fits the time series as well as possible, even over a longer period of time. Therefore, the training set contains all data from the beginning date of the given dataset until the starting point of the validation set when selecting the model, while the training set for the actual forecast consists of both the original training set and the validation set together. Essentially, we perform a hold-out sample model selection, followed by a hold-out sample forecast. The final evaluation is then comprised of the aforementioned evaluation measures, together with the profit measure. Additionally, all results are compared with the ARIMA models that were estimated using the traditional Box-Jenkins method, as described in the methodology above.

As mentioned above, the profit margin of each product category differs from the other, based on the product categories. However, the weight in the profit function is kept the same for all three datasets at a penalization rate of 1.5% per forecast error of 1% in both directions.

4.2. Results

The results of the testing process consist of a total of 160 forecasts for each of the four models that we consider, i.e. Genetic Algorithms (GA), Particle Swarm Optimization (PSO), Simulated Annealing (SA) and the traditional Box-Jenkins methodology (B-J).

As the results for all evaluation metrics are too numerous to display and as taking averages over those results can lead to misleading numbers, we turn to ranking these four models according to the measure at hand. In order to ascertain if the difference in ranking between models is significant over all given datasets, we turn to two consecutive rank tests [55]. Firstly, we perform the Friedman test, which is a non-parametric statistical test that checks whether the difference between two treatments is due to chance or not [20]. In our case, the four approaches constitute the treatments, k in the formula below, and the datasets form the blocks, n in equation 10, which are groups of similar experiment units by definition. The Friedman test then ranks each of the blocks and compares the values thereof for each treatment. In our case, this means that the four models are ranked according to the expected profit for each of the product categories in the given dataset. The Friedman test then indicates if the difference between the average ranks of the models is significant.

$$\chi_F^2 = \frac{12N}{k(k+1)}[\sum_j R_j^2 - \frac{k(k+1)^2}{4}] \qquad (10)$$

The resulting p-value indicates whether the null hypothesis, i.e. that there is no difference between the treatments, is rejected or not. If this hypothesis is rejected, the next step in the analysis consists of a post-hoc analysis, such as the Nemenyi test [48]. This test compares all models to one another and determines whether they differ significantly. This result is obtained by establishing whether the average rankings of the models are at a critical distance of at least:

$$CD = q_\alpha \sqrt{\frac{k(k+1)}{6N}} \tag{11}$$

with q_α as critical values, which consist of the Studentized range statistic divided by $\sqrt{2}$.

We will firstly discuss the results of the public datasets before turning to the use case for The Coca-Cola Company.

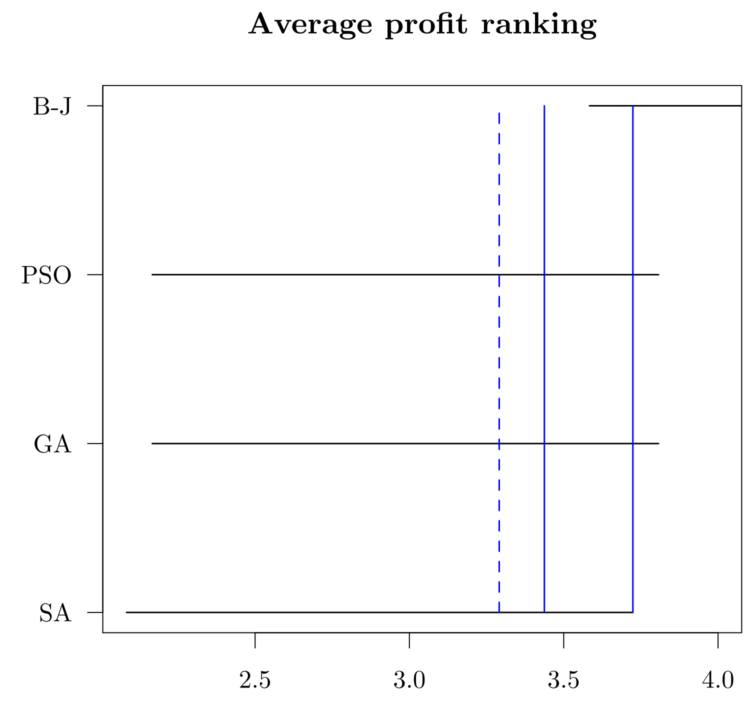

Figure 3: US petroleum: Nemenyi plot of average profit ranking

4.3. Public datasets

The tests for the first dataset, i.e. **US petroleum**, consist of 16 forecasts for each of the four models, as we predict four quarters of four product categories. As a first step, we perform the Friedman test over all four models, in which they were ranked according to the profit function. The p-value of this first test is significant at the 99% level, with a value of **0.003801**. This result therefore clearly indicates that a post-hoc Nemenyi test is necessary in order to ascertain which models differ from one another. The results of the Nemenyi test are displayed in Figure 3, where the horizontal lines connect the average rank of each model on the left side with the 99% significant critical distance on the right side. The vertical lines indicate, from left to right, the 90%, 95% and 99% critical distance with the best model.

As we can observe, all profit-based models outperform the traditional Box-Jenkins methodology and show little variation between the three of them. In terms of significance, we clearly see that all models are outperforming Box-Jenkins at the 95% significance level. When we perform the same ranking procedure according to the other evaluation metrics, we can see in Table 4 that the average rank of Box-Jenkins is always lower than the average ranks of the other models. SA has a small advantage over the other profit-based models in both profit and PE, but PSO is better than the other two in terms of RMSE and MAPE. We can therefore only conclude that for this dataset, the ProfARIMA procedure significantly outperforms the traditional methodology regardless of which search heuristic is used.

	Average ranking in:			
	Profit	**PE**	**MAPE**	**RMSE**
GA	2.17	2.42	2.33	2.42
PSO	2.17	2.25	**2.17**	**2.08**
SA	**2.08**	**2.08**	2.33	2.25
Box-Jenkins	3.58	3.25	3.17	3.25

Table 4: US petroleum: Average ranks for all evaluation metrics

The testing process of the second dataset, i.e. **Australian wines**, consists of 24 forecasts for each of the four models, since we are forecasting four quarters of six product categories. Again, we firstly perform the Friedman test according to the profit function. The p-value for this dataset is not significant, with a value of **0.3284**. This result therefore indicates that a post-hoc test is not necessary, as there is no significant difference between the four models. When we plot the results of the Nemenyi test in Figure 4, we indeed see that none of the models differ significantly from one another. SA performs slightly better than the Box-Jenkins methodology, which in turn performs slightly better than the GA and PSO profit-based models.

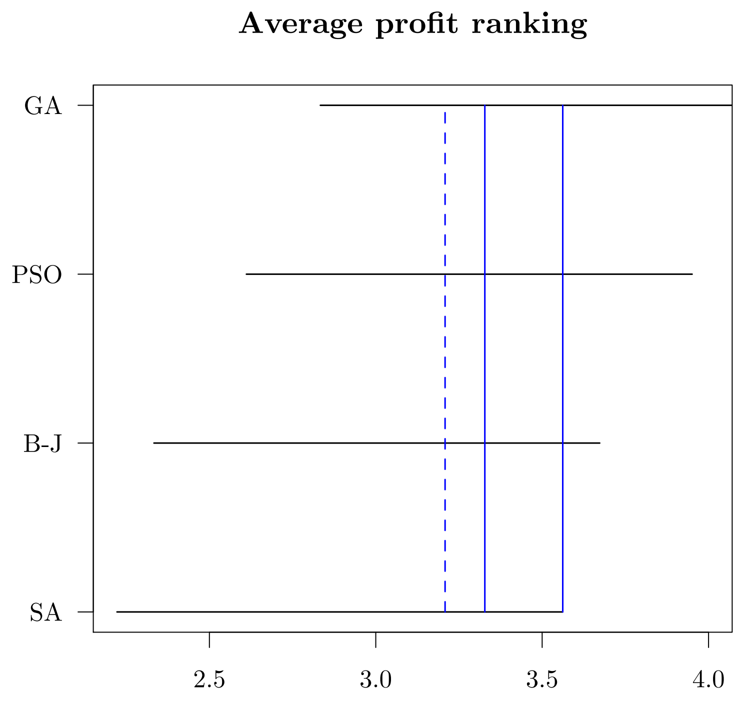

Figure 4: Australia wine: Nemenyi plot of average profit ranking

Finally, we also compare the rankings according to the other evaluation metrics in Table 5. SA outperforms all others in terms of profit, PE and RMSE, while Box-Jenkins is the best model in terms of MAPE. However, all rankings are extremely close to one another and are not significantly different, as proven above. We can therefore conclude that in this case, the ProfARIMA procedure performs equally to the traditional Box-Jenkins methodology, especially the profit-based model with SA as its search heuristic.

	Average ranking in:			
	Profit	**PE**	**MAPE**	**RMSE**
GA	2.83	2.81	3.00	2.97
PSO	2.61	2.67	2.72	2.78
SA	**2.22**	**2.25**	2.17	**2.00**
Box-Jenkins	2.33	2.28	**2.11**	2.25

Table 5: Australian wine: Average ranks for all evaluation metrics

4.4. The Coca-Cola Company use case

The real-life datasets from **The Coca-Cola Company** have a testing process of 120 forecasts for each of the four models, as we predict three quarters of 40 product categories. Firstly, we again perform the Friedman test according to the profit function. The p-value of this test is significant at the 90% level, with a value of **0.0948**. This result indicates that a post-hoc test will be interesting, as the Friedman test picked up on a significant effect.

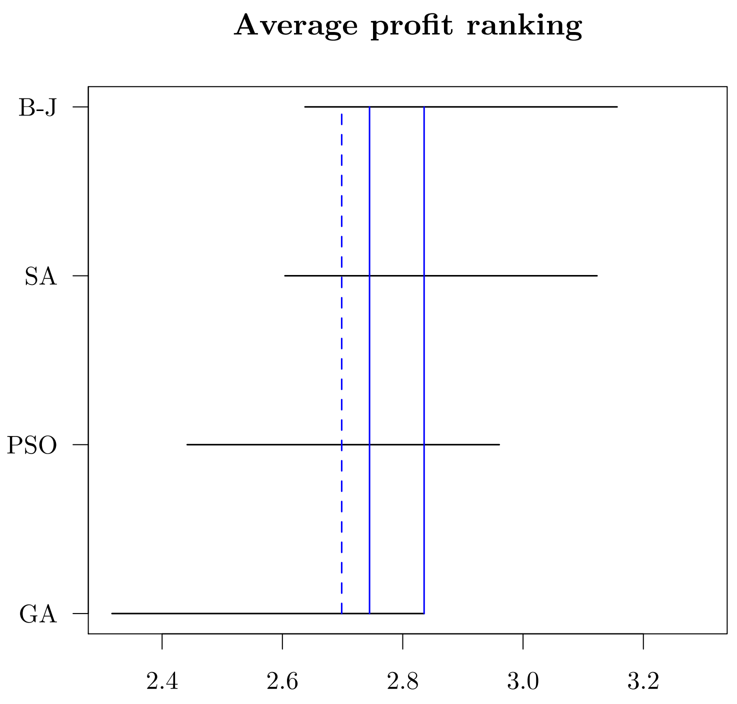

Figure 5: The Coca-Cola Company: Nemenyi plot of average profit ranking

As we can observe, the profit-based model with GA as its search heuristic, outperforms all other models, while the traditional Box-Jenkins methodology has the lowest average ranking. In terms of significance, we clearly see that even when all models are considered together, GA is just short of outperforming Box-Jenkins at the 90% significance level. It therefore seems logical to perform Friedman tests for the traditional approach and each of the other models separately, in order to determine to which level each of them outperforms the traditional methodology. The result of the pairwise tests can be found in Table 6.

	Box-Jenkins
GA	**0.01**
PSO	0.43
SA	0.98

Table 6: The Coca-Cola Company: Pairwise Friedman tests

As we can see, GA is significantly better than the Box-Jenkins approach at the 99% significance level. All other profit-based methods are not significantly better than the traditional approach, but they do still outperform the Box-Jenkins methodology on average. In order to examine the exact size of these differences, we take a look at Table 7, where the average ranks of the separate comparisons are displayed. The rows contain the average ranks of GA, PSO and SA in the left column, while the right column consists of the average rank of Box-Jenkins for each separate comparison. We observe that the difference is always in favour of the profit-based algorithm, but Box-Jenkins and the model with SA perform almost equally.

		Box-Jenkins
GA	**1.39**	1.61
PSO	**1.46**	1.54
SA	**1.49**	1.51

Table 7: The Coca-Cola Company: Average ranking of separate comparisons

Finally, we take all evaluation metrics into account, as we consider the average ranking according to PE, MAPE and RMSE as well. These results consist of the ranking with all four models and can be found in Table 8. Clearly, the model with GA outperforms all others for all evaluation metrics. PSO is not far behind, but SA only outperforms Box-Jenkins in profit.

For the Coca-Cola Company datasets, we can clearly conclude that the ProfARIMA procedure in combination with Genetic Algorithms outperforms the traditional Box-Jenkins methodology on a significant level, both in accuracy and profit. PSO and SA do not achieve the same results, but they still perform slightly better or equal to the traditional methodology, while providing an automated solution to the Seasonal ARIMA order identification problem.

	Average ranking in:			
	Profit	**PE**	**MAPE**	**RMSE**
GA	**2.31**	**2.34**	**2.35**	**2.35**
PSO	2.44	2.38	2.43	2.41
SA	2.60	2.73	2.68	2.67
Box-Jenkins	2.63	2.53	2.52	2.55

Table 8: The Coca-Cola Company: Average ranks for all evaluation metrics

5. Conclusion

The goal of this paper was to achieve a new way of automated ARIMA order identification that connects to the business context of a forecast by including profit in its estimation. We turned to three different evolutionary search heuristics, as they allowed us to specify this profit as their fitness function in order to find the optimal model. The profit function that we defined, which is applicable to any sales forecast, is a combination of business expertise and the traditional view of profit as the balance between costs and the volume of sales. This allows businesses to adjust this profit measure to their own situation. ProfARIMA can then be used in both forecasting exercises and in post-hoc analyses in order to determine which model has forecast the most profitably in the past. In this paper, the forecasting part was evaluated by applying the technique to both forecasts of public datasets and a sales forecast for the Coca-Cola Company, which consisted of forecasting 50 product categories in total. The results indicated that ProfARIMA selects a more profitable model in comparison to the Box-Jenkins methodology in most cases. The Friedman tests on these results were significant for two out of three applications, which indicates that optimizing according to the profit function often leads to the desired result of a higher profit. Although this test was not significant for the third dataset, ProfARIMA is still able to select an equally suitable model as the traditional methodology and also adds the advantage of complete automation of the parameter selection process. ProfARIMA has proven to be a successful way of automated ARIMA order identification that is based on business expertise, but does not require the manual labour of the traditional Box-Jenkins approach.

In terms of *generalizability*, the profit function was able to adjust to different situations with other products, as we have proven by taking 50 different cases into account that stem from three different datasets and application domains. However, the same function can also easily be deployed for different product ranges. Experts have the flexibility of setting the profit margins themselves, even in relative terms if exact numbers are difficult to obtain, as well as the accuracy margins that need to be penalized. Furthermore, the size of this penalization factor can be set by performing a sensitivity analysis on a sample of the given datasets in order to ensure its correctness for the application in question. Lastly, the use of publicly available datasets also increases the generalizability of the ProfARIMA procedure. In terms of *reliability*, this study was enhanced by applying multiple search algorithms, of which the parameters were all optimized according to an established procedure. The ProfARIMA process was then tested on both publicly available and company-specific datasets that contained a wide range of products from different markets.

However, some limitations still remain, as further analysis can always be done. Firstly, ProfARIMA should be tested on other applications by extending the research to other sales domains with different characteristics, such as retail. Future research also includes the expansion of the present profit function in order to include additional variation between products. For example, the purpose of a product can play a role in its profitability. In the application of this paper, certain drinks can be meant for immediate consumption, while others are intended for home use. In other situations, a certain product might be bought for short-term or long-term use. In both examples, the profit margin of the second type of products will generally be slightly lower than for the first one. Furthermore, even though this paper already considered three different search algorithms, there is still room for expansion there as well. Lastly, the same profit function can be tested to optimize the parameters of other forecasting techniques, such as Artificial Neural Networks. However, many of these limitations are simply part of future research on the integration of profit measures into model selection in forecasting.

The main contribution of this paper consists of the flexible profit function that can be adjusted to any sales forecast and that combines business expertise with a traditional accuracy measure. This function can be used to guide ARIMA model selection if it is used as a fitness function for evolutionary search heuristics. Depending on the dataset, different search heuristics were more successful than others. Especially Genetic Algorithms and Simulated Annealing were extremely successful in selecting more profitable and more accurate models than the traditional Box-Jenkins manual parameter selection in the applications of this paper, although the performance of both was highly dependent on the given dataset. While PSO was slightly less convincing, it still succeeded in selecting models that performed at least equally to the traditional approach. Therefore, the ProfARIMA procedure seems to be a reliable methodology for ARIMA order identification in sales forecasting that is flexible, completely automated and easy to maintain, and that takes into account the business aspect of a forecast.

Acknowledgements

We would like to acknowledge The Coca-Cola Company for funding this research and providing us with the necessary business expertise and data to conduct our experiments.

Vitae

Tine Van Calster is a PhD candidate at the Faculty of Economics and Business (FEB) at KU Leuven. She has obtained two Master degrees at KU Leuven in both Computational Linguistics and Artificial Intelligence, with a specialization in Natural Language Processing. Her current research topics include forecasting and time series prediction with applications in sales forecasting. She conducts her research in cooperation with The Coca-Cola Company.

Professor Bart Baesens is a professor at KU Leuven (Belgium), and a lecturer at the University of Southampton (United Kingdom). He has done extensive research on big data & analytics, customer relationship management, web analytics, fraud detection, and credit risk management. His findings have been published in well-known international journals (e.g. Machine Learning, Management Science, IEEE Transactions on Neural Networks, IEEE Transactions on Knowledge and Data Engineering, IEEE Transactions on Evolutionary Computation, Journal of Machine Learning Research,) and presented at international top conferences. His research and publications are summarized at www.dataminingapps.com. He also regularly tutors, advises and provides consulting support to international firms.

Professor Wilfried Lemahieu is full professor and Vice Dean for Education at the Faculty of Economics and Business (FEB) at KU Leuven. His teaching includes Database Management, Enterprise Information Management and Management Informatics. His current research focuses on big data storage & analytics, data quality, business process management and service oriented architectures. His research is published in international journals such as Decision Support Systems, IEEE Software, Information & Management and Data & Knowledge Engineering. He is also a frequent lecturer for both academic and business audiences and has an extensive track record in research collaborations with industry. See feb.kuleuven.be/wilfried.lemahieu for further details.

Bibliography

[1] Abo-Hammour, Z. S., Alsmadi, O. M., Al-Smadi, A. M., Zaqout, M. I., Saraireh, M. S., 2012. Arma model order and parameter estimation using genetic algorithms. Mathematical and Computer Modelling of Dynamical Systems 18 (2), 201–221.

[2] Adhikari, R., Agrawal, R., 2014. A combination of artificial neural network and random walk models for financial time series forecasting. Neural Computing and Applications 24 (6), 1441–1449.

[3] Akaike, H., 1974. A new look at the statistical model identification. IEEE transactions on automatic control 19 (6), 716–723.

[4] Akın, M., 2015. A novel approach to model selection in tourism demand modeling. Tourism Management 48, 64–72.

[5] Awan, S. M., Aslam, M., Khan, Z. A., Saeed, H., 2014. An efficient model based on artificial bee colony optimization algorithm with neural networks for electric load forecasting. Neural Computing and Applications 25 (7-8), 1967–1978.

[6] Babu, C. N., Reddy, B. E., 2014. A moving-average filter based hybrid arima–ann model for forecasting time series data. Applied Soft Computing 23, 27–38.

[7] Balaprakash, P., Birattari, M., Stützle, T., 2007. Improvement strategies for the f-race algorithm: Sampling design and iterative refinement. In: International Workshop on Hybrid Metaheuristics. Springer, pp. 108–122.

[8] Baragona, R., Battaglia, F., Poli, I., 2011. Evolutionary Statistical Procedures: An Evolutionary Computation Approach to Statistical Procedures Designs and Applications. Springer Science & Business Media.

[9] Battaglia, R., Baragona, F., Cucina, D., 2004. Estimating threshold subset autoregressive moving-average models by genetic algorithms. Metron 62 (1), 39–61.

[10] Birattari, M., Yuan, Z., Balaprakash, P., Stützle, T., 2010. F-race and iterated f-race: An overview. In: Experimental methods for the analysis of optimization algorithms. Springer, pp. 311–336.

[11] Bonyadi, M. R., Michalewicz, Z., 2016. Particle swarm optimization for single objective continuous space problems: a review. Evolutionary computation.

[12] Box, G. E., Jenkins, G. M., Reinsel, G. C., Ljung, G. M., 2015. Time series analysis: forecasting and control. John Wiley & Sons.

[13] Chang, P.-C., Wu, J.-L., Lin, J.-J., 2016. A takagi–sugeno fuzzy model combined with a support vector regression for stock trading forecasting. Applied Soft Computing 38, 831–842.

[14] Chen, R., Liang, C.-Y., Hong, W.-C., Gu, D.-X., 2015. Forecasting holiday daily tourist flow based on seasonal support vector regression with adaptive genetic algorithm. Applied Soft Computing 26, 435–443.

[15] Dickey, D. A., Fuller, W. A., 1979. Distribution of the estimators for autoregressive time series with a unit root. Journal of the American statistical association 74 (366a), 427–431.

[16] Du, W., Leung, S. Y. S., Kwong, C. K., 2015. A multiobjective optimization-based neural network model for short-term replenishment forecasting in fashion industry. Neurocomputing 151, 342–353.

[17] Eberhart, R. C., Kennedy, J., et al., 1995. A new optimizer using particle swarm theory. In: Proceedings of the sixth international symposium on micro machine and human science. Vol. 1. New York, NY, pp. 39–43.

[18] Eiben, A. E., Smit, S. K., 2011. Evolutionary algorithm parameters and methods to tune them. In: Autonomous search. Springer, pp. 15–36.

[19] Fard, A. K., Akbari-Zadeh, M.-R., 2014. A hybrid method based on wavelet, ann and arima model for short-term load forecasting. Journal of Experimental & Theoretical Artificial Intelligence 26 (2), 167–182.

[20] Friedman, M., 1940. A comparison of alternative tests of significance for the problem of m rankings. The Annals of Mathematical Statistics 11 (1), 86–92.

[21] Gaetan, C., 2000. Subset arma model identification using genetic algorithms. Journal of Time Series Analysis 21 (5), 559–570.

[22] Gan, M., Cheng, Y., Liu, K., Zhang, G.-l., 2014. Seasonal and trend time series forecasting based on a quasi-linear autoregressive model. Applied Soft Computing 24, 13–18.

[23] Geng, J., Huang, M.-L., Li, M.-W., Hong, W.-C., 2015. Hybridization of seasonal chaotic cloud simulated annealing algorithm in a svr-based load forecasting model. Neurocomputing 151, 1362–1373.

[24] Ghasemi, A., Shayeghi, H., Moradzadeh, M., Nooshyar, M., 2016. A novel hybrid algorithm for electricity price and load forecasting in smart grids with demand-side management. Applied Energy 177, 40–59.

[25] Guo, W., Xu, T., Lu, Z., 2016. An integrated chaotic time series prediction model based on efficient extreme learning machine and differential evolution. Neural Computing and Applications 27 (4), 883–898.

[26] Haseyama, M., Kitajima, H., 2001. An arma order selection method with fuzzy reasoning. Signal processing 81 (6), 1331–1335.

[27] Hochreiter, R., Waldhauser, C., 2015. Evolving accuracy: A genetic algorithm to improve election night forecasts. Applied Soft Computing 34, 606–612.

[28] Holland, J. H., 1975. Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence. U Michigan Press.

[29] Hung, J.-C., 2008. A genetic algorithm approach to the spectral estimation of time series with noise and missed observations. Information Sciences 178 (24), 4632–4643.

[30] Hung, J.-C., 2011. Applying a combined fuzzy systems and garch model to adaptively forecast stock market volatility. Applied Soft Computing 11 (5), 3938–3945.

[31] Jain, C. L., 2004. How to measure the cost of a forecast error. The Journal of Business Forecasting Methods & Systems 22 (4), 2.

[32] Jiang, H., Zhang, Y., Muljadi, E., Zhang, J., Gao, W., 2016. A short-term and high-resolution distribution system load forecasting approach using support vector regression with hybrid parameters optimization. IEEE Transactions on Smart Grid.

[33] Kahn, K. B., 2003. How to measure the impact of a forecast error on an enterprise? The Journal of Business Forecasting 22 (1), 21.

[34] Kankal, M., Uzlu, E., 2016. Neural network approach with teaching–learning-based optimization for modeling and forecasting long-term electric energy demand in turkey. Neural Computing and Applications, 1–11.

[35] Kennedy, J., Russell, E., 1995. Particle swarm optimization. In: Proceedings of 1995 IEEE International Conference on Neural Networks. pp. 1942–1948.

[36] Kirkpatrick, S., Gelatt, C. D., Vecchi, M. P., et al., 1983. Optimization by simmulated annealing. science 220 (4598), 671–680.

[37] Krol, R., 2013. Evaluating state revenue forecasting under a flexible loss function. International Journal of Forecasting 29 (2), 282–289.

[38] Li, H.-Z., Guo, S., Li, C.-J., Sun, J.-Q., 2013. A hybrid annual power load forecasting model based on generalized regression neural network with fruit fly optimization algorithm. Knowledge-Based Systems 37, 378–387.

[39] Li, M.-W., Geng, J., Hong, W.-C., Chen, Z.-Y., 2016. A novel approach based on the gauss-vsvr with a new hybrid evolutionary algorithm and input vector decision method for port throughput forecasting. Neural Computing and Applications, 1–20.

[40] Li, M.-W., Han, D.-F., Wang, W.-l., 2015. Vessel traffic flow forecasting by rsvr with chaotic cloud simulated annealing genetic algorithm and kpca. Neurocomputing 157, 243–255.

[41] Lin, W.-Y., 2015. A novel 3d fruit fly optimization algorithm and its applications in economics. Neural Computing and Applications, 1–23.

[42] Liu, D., Niu, D., Wang, H., Fan, L., 2014. Short-term wind speed forecasting using wavelet transform and support vector machines optimized by genetic algorithm. Renewable Energy 62, 592–597.

[43] Lopez-Garcia, P., Onieva, E., Osaba, E., Masegosa, A. D., Perallos, A., 2016. A hybrid method for short-term traffic congestion forecasting using genetic algorithms and cross entropy. IEEE Transactions on Intelligent Transportation Systems 17 (2), 557–569.

[44] Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., Teller, E., 1953. Equation of state calculations by fast computing machines. The journal of chemical physics 21 (6), 1087–1092.

[45] Minerva, T., Poli, I., 2001. Building arma models with genetic algorithms. In: Workshops on Applications of Evolutionary Computation. Springer, pp. 335–342.

[46] Montero, E., Riff, M.-C., Neveu, B., 2014. A beginner's guide to tuning methods. Applied Soft Computing 17, 39–51.

[47] Mustaffa, Z., Yusof, Y., Kamaruddin, S. S., 2014. Enhanced artificial bee colony for training least squares support vector machines in commodity price forecasting. Journal of Computational Science 5 (2), 196–205.

[48] Nemenyi, P., 1963. Distribution-free multiple comparisons. Ph.D. thesis, Princeton University.

[49] Ong, C.-S., Huang, J.-J., Tzeng, G.-H., 2005. Model identification of arima family using genetic algorithms. Applied Mathematics and Computation 164 (3), 885–912.

[50] Ord, K., Fildes, R., 2013. Principles of business forecasting. Cengage Learning.

[51] Osborn, D. R., Chui, A. P., Smith, J. P., Birchenhall, C. R., 1988. Seasonality and the order of integration for consumption. Oxford Bulletin of Economics and Statistics 50 (4), 361–377.

[52] Shi, Y., Eberhart, R., 1998. A modified particle swarm optimizer. In: Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference on. IEEE, pp. 69–73.

[53] Tsallis, C., Stariolo, D. A., 1996. Generalized simulated annealing. Physica A: Statistical Mechanics and its Applications 233 (1), 395–406.

[54] ul Islam, B., Baharudin, Z., Raza, M. Q., Nallagownden, P., 2014. Optimization of neural network architecture using genetic algorithm for load forecasting. In: Intelligent and Advanced Systems (ICIAS), 2014 5th International Conference on. IEEE, pp. 1–6.

[55] Verbeke, W., Dejaeger, K., Martens, D., Hur, J., Baesens, B., 2012. New insights into churn prediction in the telecommunication sector: A profit driven data mining approach. European Journal of Operational Research 218 (1), 211–229.

[56] Verbraken, T., Verbeke, W., Baesens, B., 2013. A novel profit maximizing metric for measuring classification performance of customer churn prediction models. IEEE Transactions on Knowledge and Data Engineering 25 (5), 961–973.

[57] Wang, F.-K., Du, T., 2014. Implementing support vector regression with differential evolution to forecast motherboard shipments. Expert Systems with Applications 41 (8), 3850–3855.

[58] Whitfield, R., Duffy, A., 2013. Extended revenue forecasting within a service industry. International Journal of Production Economics 141 (2), 505–518.

[59] Whitle, P., 1951. Hypothesis testing in time series analysis. Vol. 4. Almqvist & Wiksells.

[60] Wu, J., Long, J., Liu, M., 2015. Evolving rbf neural networks for rainfall prediction using hybrid particle swarm optimization and genetic algorithm. Neurocomputing 148, 136–142.

[61] Wu, L., Liu, S., Yang, Y., 2016. Grey double exponential smoothing model and its application on pig price forecasting in china. Applied Soft Computing 39, 117–123.

[62] Yeh, W.-C., Yeh, Y.-M., Chang, P.-C., Ke, Y.-C., Chung, V., 2014. Forecasting wind power in the mai liao wind farm based on the multilayer perceptron artificial neural network model with improved simplified swarm optimization. International Journal of Electrical Power & Energy Systems 55, 741–748.

[63] Yu, S., Wang, K., Wei, Y.-M., 2015. A hybrid self-adaptive particle swarm optimization–genetic algorithm–radial basis function model for annual electricity demand prediction. Energy Conversion and Management 91, 176–185.

[64] Yu, X., Gen, M., 2010. Introduction to evolutionary algorithms. Springer Science & Business Media.

[65] Zhang, G. P., 2003. Time series forecasting using a hybrid arima and neural network model. Neurocomputing 50, 159–175.