Arab J Sci Eng
DOI 10.1007/s13369-015-1587-z

RESEARCH ARTICLE - COMPUTER ENGINEERING AND COMPUTER SCIENCE

# Forecasting Oil Production Time Series with a Population-Based Simulated Annealing Method

**Juan Frausto-Solís · Manuel Chi-Chim · Leonid Sheremetov**

Received: 8 September 2014 / Accepted: 19 January 2015

**Abstract** This paper addresses the oil production forecasting problem, closely related to the estimation of reserves remaining in oil fields. We present a new method, named SAM-oil, which combines simulated annealing metaheuristic with evolutionary computing techniques, statistical analysis and quality assessment of solutions. The method implements a learning-based scheme for pre-processing, modeling and forecasting of monthly oil production time series. Accuracy of point forecasts is compared between SAM-oil and a typical technique in petroleum engineering, known as decline curve analysis, as well as with well-established forecasting methods such as ARIMA, neural networks, and some members of the exponential smoothing family. For the study case, a clustering process has been conducted in order to map forecasting difficulty for three clusters of time series. Our experiments evidence that SAM-oil's is a very competitive method in oil production forecasting: SAM-oil's forecasts outperform, in average, those from decline curve analysis and the other forecasting methods, both for the clusters and for the whole set of the experimental time series.

**Keywords** Time series · Forecasting · Oil production · Simulated annealing · Optimization

J. Frausto-Solis (✉)
Instituto Tecnológico de Ciudad Madero, Tecnológico Nacional de México, Ciudad Madero, Tamaulipas, Mexico
e-mail: juan.frausto@gmail.com

M. Chi-Chim
Mexican Petroleum Institute - ITESM Campus Cuernavaca, Ciudad de México, México
e-mail: machi@imp.mx

L. Sheremetov
Mexican Petroleum Institute, Ciudad de México, México
e-mail: sher@imp.mx

Published online: 10 February 2015

## 1 Introduction

World economy relies heavily on hydrocarbons, particularly oil, for the provision of energy required in transportation and industry. According to [1], fossil fuels will continue to provide about 88 % of global energy until the year 2040 and it is expected that global demand for oil to grow by 56 % between the years 2010–2040. Dependence on oil and its finite nature, pose some complex problems including estimation of 1) recoverable oil, 2) depletion rate and 3) future production patterns. In petroleum engineering, reserves estimation (the evaluation for the economically recoverable hydrocarbons in a field, area or region) has been a challenge for many years.

Decline curve analysis (DCA) has been one of the most common techniques for estimating initial hydrocarbons in place, hydrocarbon reserves at some abandonment conditions and future production [2]. By identifying decline curves, DCA extrapolates past production in order to estimate expected production and remaining reserves. Extrapolation of production history has been considered the most accurate method to estimate remaining reserves [3]. However, production decline data can produce a poor fit on decline curves. Hence, to deal with under-estimation or overestimation of cumulative production, more accurate forecasting requires methods that can obtain more appropriate functions for decline analysis. Prediction functions should generate minimum prediction errors for an appropriate forecasting model. The most common error measure is the average distance between forecasted values and observations taken from the production time series. Searching for the best function for forecasting becomes an extremely complex optimization problem: the search space is too large and times series have abrupt changes in both value and direction.

In this paper, we propose a new method based on simulated annealing (SA) for determining DCA functions and their best parameters and to use such functions to forecast monthly oil production. In a previous work, we explored the use of simulated annealing for fitting of decline curves, not aimed to forecasting [4]. The new method that we propose here, named SAM-oil (Simulated Annealing based on Machine learning for oil prediction), decreases the forecasting error by applying an incremental correction, while the best parameters are searched. We assert that SAM-oil is population-based, because a simulated annealing process searches the best solution by evolving a population through crossover and mutation operations.

In this paper, we highlight the following contributions: i) SAM-oil obtains functions for oil production decline consistent with DCA, but providing more accurate forecasts, ii) such forecasts outperform several well-established forecasting methods, and finally iii) we propose a clustering procedure for distribution of the tested time series into clusters, depending upon their forecasting difficulty. In order to validate the proposed approach, we use a real data set from an oilfield located in the coastal swamps of the Gulf of Mexico. We perform a comparison of SAM-oil and other methods over the whole set of samples. In addition, the clustering procedure enables comparison of average forecasting performance over each cluster. Besides comparison with forecasts computed by DCA, SAM-oil is also compared with ARIMA, the classical forecasting method by Box and Jenkins [5], which has been very successful for a wide variety of time series [6] including oil production forecasting [7–9]. Furthermore, we compare our results with the neural networks approach, which has been frequently used for oil production forecasting [10,11]; particularly, we contrast our model with a NARX neural network—a nonlinear autoregressive exogenous model. Finally, SAM-oil is compared with some methods of the exponential smoothing family, widely used in general forecasting applications.

The rest of the paper is organized as follows. In Sect. 2, we present the conceptual background behind SAM-oil development. Section 3 describes the SAM-oil method in detail. Section 4 presents the experimental setup along with results obtained. In Sect. 5, we discuss some previous work related to forecasting of time series in petroleum engineering. Finally, Sect. 6 presents results and future work.

## 2 Fundamentals of the SAM-Oil Method

Several methods have been reported in the literature for the oil forecasting problem (see Sect. 5 below). However, performance of all of these methods is affected by challenging features of the problem. For example, even though traditional regression techniques can obtain exponential curves for DCA (least-squares method can be applied over the linearized expression of the exponential function), some traits in the time series for hydrocarbons production discourage the application of such techniques [12]. In particular, oil production values are constantly affected by technical interventions throughout extraction process and, as a result, unusual observations or outliers are frequent; furthermore, data heterogeneities in the time intervals are common. Furthermore, kurtosis and skewness values show that this type of time series does not conform to a normal distribution. In addition, although short-term forecasting is usually feasible for this type of time series, presence of such outliers largely affects accuracy.

Another side of the challenge is related to the time series themselves. In our study case, the series contain 300 or even a smaller number of points (Fig. 1). The difficulty of predicting these time series is very high due to discontinuities, spikes and segments with different trend. Furthermore, the

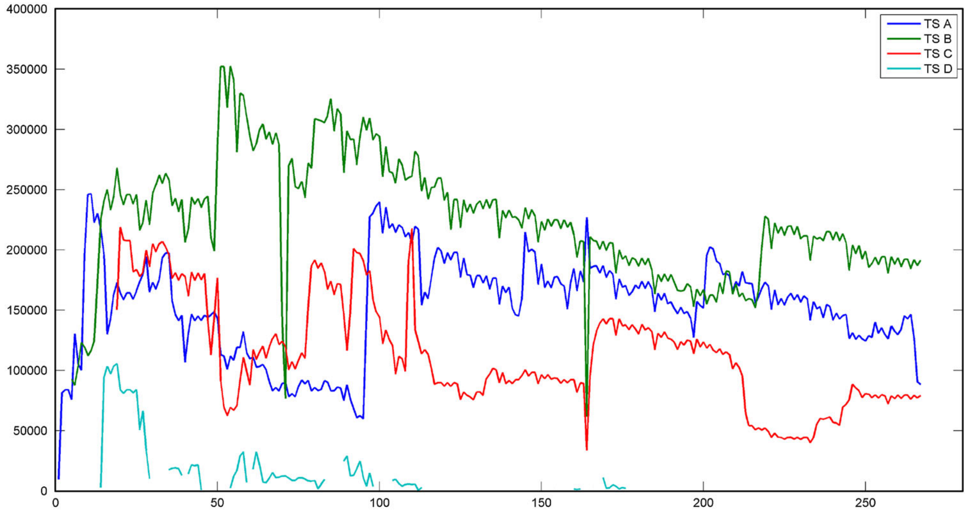

**Fig. 1** Some examples of oil production time series from the case study

time series lack features such as seasonality or cycles. In general, features of oil production time series separate them from financial or physical processes time series [13].

The following subsections briefly describe the set of methods and techniques involved in this work, related to the SAM-oil development. Our method extends DCA with techniques of metaheuristic optimization and traditional forecasting methods, in order to increase forecasting accuracy.

## 2.1 Time Series Forecasting

A time series is a sequence of observations (usually equidistant in time) about a changing phenomenon. The underlying idea of TS forecasting is that patterns associated with past values in a data series can be used to project future values [14]. If the sequence of observations from a time series is expressed by $x(t)$ $t = 1, 2, \ldots$ then the expression $\hat{x}(t+h)$ denotes the horizon of prediction for period $h$ at time $t$, where $h = 1$ means one step ahead and $h > 1$ means multi-step ahead prediction. In general, time series forecasting methods can be classified into three types: judgmental, univariate and multivariate [15]. Judgmental forecasts are based on subjective interpretation of any relevant information. In univariate methods, forecasts depend only on present and past values of a single time series. In multivariate methods, forecasts for a variable depend, at least partly, on values of additional time series variables.

In [16], univariate and multivariate methods are organized into two further subclasses: traditional statistical methods and data mining methods. This research is partially supported on traditional statical methods (exponential smoothing and ARIMA) for both development and comparison purposes, and on data mining methods (neural networks) for comparison purposes.

### 2.1.1 *Exponential Smoothing Methods*

A common approach for forecasting identifies certain components of the time series such as level, trend, cycle and seasonality, in order to obtain models suitable to predict future values. Along with an unpredictable component of the series, these components can generate models to make predictions and probability distributions for future behavior of data. Pioneering works of Brown [17], Holt [18] and Winters [19] about exponential smoothing became the basis of a powerful family of modern forecasting methods.

Exponential smoothing computes forecasts as weighted averages of past observations —weights decay exponentially as the observations become older. State space model approach, underlying the exponential smoothing models [20], generates the same point forecasts and also prediction intervals. The following equations define a general state space model [21]:

$$y_t = w(\mathbf{x}_{t-1}) + r(\mathbf{x}_{t-1})\varepsilon_t \tag{1}$$

$$\mathbf{x}_t = f(\mathbf{x}_{t-1}) + g(\mathbf{x}_{t-1})\varepsilon_t. \tag{2}$$

In this model, $y_t$ is an observation at time t, $\mathbf{x_t}$ is the state vector which contains unobserved components of level, trend and seasonality and $\{e_t\}$ is a white noise series. Equation (1) describes the relationship between unobserved state $\mathbf{x_t}$ and observed data $y_t$, while (2) describes state transition over time. By combining different types of trend and seasonal components fifteen methods of exponential smoothing are defined [22]; for each method, two space state models can be defined —one with additive errors and the other with multiplicative errors. To illustrate the above, consider the exponential smoothing method for data with a trend. Expressions for level $\ell$ and trend $b$ of such method are

$$\ell_t = \ell_{t-1} + b_{t-1} + \alpha\varepsilon_t \tag{3}$$

$$b_t = b_{t-1} + \beta\varepsilon_t, \tag{4}$$

where $\alpha$ and $\beta$ are the smoothing parameters for the level and trend, respectively. Then, expressions for the corresponding space state model are

$$y_t = \begin{bmatrix} 1 & 1 \end{bmatrix} \mathbf{x}_{t-1} + \varepsilon_t$$

$$\mathbf{x}_t = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \mathbf{x}_{t-1} + \begin{bmatrix} \alpha \\ \beta \end{bmatrix} \varepsilon_t,$$

where $\mathbf{x_t} = [\ell_t, b_t]^\intercal$ is a state vector.

The models obtained by the exponential smoothing and state space approaches explicitly state the components for the level, trend and seasonality, unlike the ARIMA approach [21]. However, forecasting results can be sensitive to initial states and parameter settings, i.e., $\alpha$ and $\beta$ in the above model.

## 2.2 Decline Curve Analysis Method

In petroleum engineering, for long time, the classical way of production forecasting has been made through decline curve analysis [23], a kind of graphical method [24]. DCA can also determine a rate known as Estimated Ultimate Recovery (EUR), the sum of proved reserves at a certain point of time and the cumulative production up to that time. Appraisement of EUR is perhaps the most important research problem in the assessment of hydrocarbon resources [25]. A particular pattern of decline trend reflects on production rates and operating conditions that would influence the performance, such as natural drive, rock and fluid properties, and well completion [26]. More recent works also use DCA for estimation of other parameters of the wells [27].

A well's declination curve plot shows the production rate in the $Y$ axis (logarithmically scaled) and the time in the $X$ axis. Based on the so called Arps' equations [28], the DCA method can be seen as a method of curve fitting to one of three types a) exponential, when $D_i$, the decline rate, is

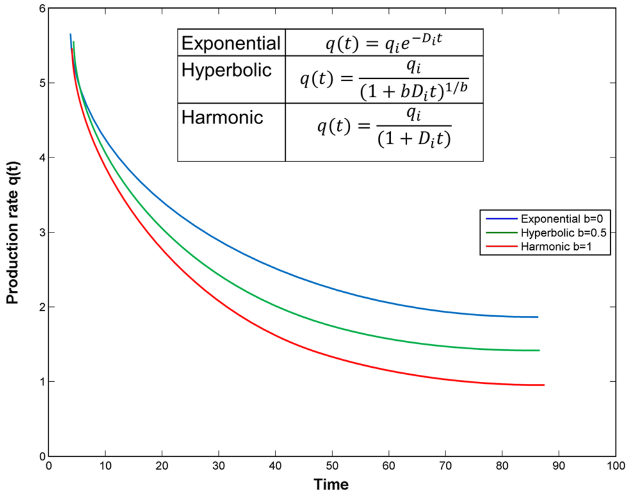

**Fig. 2** Decline curve types

constant; b) hyperbolic, when $b$ varies in range $0 < b < 1$, to account the decline rate; and c) harmonic, if $b = 1$ equals one (Fig. 2). The first two are more common than the last one [29]. The $b$ parameter, known as the Arps depletion-decline exponent, is a relation between single-phase flow regime, high-pressure gas, gas relative permeability, tubing-restricted gas production, pressure under or above bubble point, and waterflood performance [24]. When the depicted pattern shows a straight line, the exponential decline model inferred is

$$q(t) = q_i e^{-D_i t} \tag{5}$$

where $q(t)$ is the production rate at $t$ time, $q_i$ is the initial production rate and $D_i$ is the initial decline rate.

However, in decline curve analysis, it is difficult to anticipate the behavior (exponential, hyperbolic, harmonic) that a reservoir will follow. Furthermore, an exponential curve tends to under-estimate and a harmonic curve tends to over-estimate the production rates [30].

## 2.3 Metaheuristic Optimization Algorithms

Simulated annealing and genetic algorithms are two successful metaheuristic algorithms to find good solutions (not necessarily optimal) for optimization problems, in which the large space of possible solutions makes unfeasible an exhaustive search. We briefly describe these metaheuristics, as the SAM-oil method uses them to search for the best functions of oil production decline, which in turn will be used for oil production forecasting.

### *2.3.1 Simulated Annealing*

The simulated annealing optimization algorithm is a very successful algorithm for a wide variety of optimization problems [31] inspired by the annealing process of solids. In this process, a solid is i) heated to the melting point and then ii) its temperature is decreased slowly until the desired energy state is achieved [32]. Proposed by Kirkpatrick [33] and Cerny [34], the SA algorithm is an analogy for the solid annealing process in which the states and energy correspond, respectively, to feasible solutions and to the objective function value in an optimization problem [35]. In other words, simulated annealing algorithm goes through different states (solutions), accepting new ones with lower energy levels (solutions with better values of objective function). The algorithm can also accept "bad" states, according to the Boltzmann probability distribution, in order to escape from local optima. This search for better solutions is done within the called Metropolis cycle, which runs repeatedly until a point called thermal equilibrium is reached. The Metropolis cycle controls a descending sequence of a temperature parameter, initially set to a conveniently high value. In other words, an external cycle controls the temperature value through a cooling function, whereas the internal cycle creates neighbors to the current solution. SA (Algorithm 1) accepts solutions if the objective function value is better (i.e., lower for a minimizing problem); otherwise, it can accept worse values according to the Boltzmann distribution. Control parameters of the algorithm include initial temperature $c_i$, final temperature $c_f$, length $L$ of Markov chain and parameter $\beta$ of the cooling function.

**Algorithm 1** Simulated annealing classic algorithm

```
Initialize $(c = c_i, c_f, S_{best}, \beta, L)$
Do while $c > c_f$
  Repeat L times
    Generate $S_{new}$
    If $E(S_{new}) < E(S_{best})$
      $S_{best} = S_{new}$
    Else
      If random < Boltzman
        $S_{best} = S_{new}$
      End
    End
  End
  $c$ = Cooling–function$(\beta, c)$
End
```

Tunings of the algorithm affect both, solutions quality and execution time. One technique, called reheat or reannealing, attempts to avoid local optima by resetting current temperature to some convenient and higher value. Then annealing process resumes, looking for a better exploration of solution space. Another tuning procedure is related to the cooling scheme of the temperature (the $\beta$ parameter). A fast-cooling procedure leads to fast exploration of solutions with the risk of falling into local optima while a slow-cooling procedure allows better exploration at the expense of longer execution time.

**Algorithm 2** A basic genetic algorithm

```
(Initialize) Create random population P of $n$ individuals
Do While Stop condition is not satisfied
  Evaluate fitness $f(x)$ for each individual $x$ in P
  Do While P' is not complete
    Select parents $p_1$, $p_2$ from P
    ($Child_1$, $Child2$) = Crossover($p_1$, $p_2$)
    Mutate ($Child_1$, $Child2$)
    Add ($Child_1$, $Child2$) to new population P'
  End
  Replace P with P'
End
```

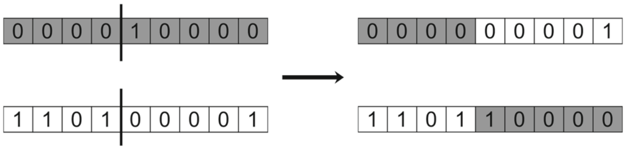

**Fig. 3** One-point crossover to merge two solutions in a genetic algorithm

### 2.3.2 Genetic Algorithms

A genetic algorithm [36] finds exact or approximate solutions in optimization, search and learning problems. Inspired by Darwin's evolution theory, genetic algorithm's solutions evolve in time by means of selection, crossover and mutation operations, analogous to those from nature. Each problem solution is codified as a finite-length string (chromosome) over a finite alphabet [37]. The algorithm defines a population of solutions or individuals. Individuals create successive offsprings looking for better solutions; the selection operation chooses parents according to their fitness value—an evaluation of the objective function in the problem domain. The whole process (Algorithm 2) iterates for a certain number of offsprings, or stops when a predefined fitness threshold is reached for the current best solution.

Unlike other search methods, genetic algorithms only require fitness values associated with each individual and use probabilistic transition rules to search for improvements in the solution space. These rules are implicit in the three basic operations:

- Selection. The algorithm chooses some individuals to create a new offspring. A common criterium defines a roulette, where each individual is assigned a section whose size is proportional to its fitness value. The roulette "spins" by generating a random number between (0,1) and then selects as a parent the individual whose section is pointed by the number. In this way, individuals with better fitness are more likely to be selected to generate the new offspring.
- Crossover. The algorithm combines or merges the codification of two parents to create new individuals. A common operator, called one-point crossover, randomly selects a point in the chromosome string and then exchanges values from that point to create two children (Fig. 3).
- Mutation. The algorithm changes probabilistically some elements of the codification string, searching for new characteristics unavailable in individuals from earlier populations. Mutation prevents a genetic algorithm to converge quickly without a sufficient exploration of search space.

## 3 SAM-Oil: A Population-Based Hybrid Method for Oil Production Time Series Forecasting

SAM-oil obtains a characterization of hydrocarbons production decline through time series (TS) forecasting. Generally speaking, SAM-oil combines statistical pre-processing of data, simulated annealing techniques and residual modeling to find forecasting functions for oil production time series. The method finds the best parameters for DCA functions and uses them to forecast monthly oil production. In order to obtain a characterizing curve of an entire time series, the method takes into account outliers and heterogeneity of time intervals. SAM-oil uses operators from genetic algorithms to guide the search through populations of such forecasting functions. SAM-oil 1) determines exponential decline curves consistent with those obtained in traditional DCA and 2) obtains a more general regression function based on an exponential decline curve incrementally corrected for forecasting purposes. The approach can be easily extended for the hyperbolic or harmonic decline cases.

The general steps of SAM-oil are:

1. Data filtering with deviation analysis and exponential regression
2. Search of decline curves based on simulated annealing and a population-based approach
3. Residual modeling for refinement of forecasting

The method works by splitting a time series into three segments: *training*, *pre-testing* and *testing*. The method searches for a model that best fits training segment data, while assesses forecasting accuracy on some pre-testing segment data to improve the model. The obtained model computes forecasts for the the testing segment, which contains data unknown to the modeling process. In this way, the method compares forecasted values with test values in order to measure model accuracy.

**Algorithm 3** Approach for data filtering in time series

```
Obtain training segment T of the time series
Fit an exponential function $f(t)$ to the T segment
For each $(t_i, y_i)$ in T
  Compute value $\hat{y}_i = f(t_i)$
  Compute $MAD = |(y_i - \hat{y}_i)/y_i|$
  If $MAD > \epsilon$
    Replace $y_i$ in T with closest margin value
  End
End
```

### 3.1 Data Filtering with Deviation Analysis and Exponential Regression

The first step of the method filters observations in time series in order to reduce impact of outliers (Algorithm 3). The process first determines a regression function for the complete set of training data —an exponential function determined by a least-squares regression. The obtained function computes an adjusted value for each training point. Then, the process calculates values of mean absolute deviation (MAD) between adjusted and observed values. Training points whose MAD values exceed a reference value are marked as outliers. Filtering process defines the upper and lower bounds of tolerance margins for the training points as $\hat{y}_i \pm \epsilon$, where $\hat{y}_i$ is the adjusted value for each point and $\epsilon$ is a percentage of tolerance for deviations. At the end, the values of closest margin bounds replace outliers.

### 3.2 Search of Decline Curves Based on Simulated Annealing and a Population-Based Approach

SAM-oil uses simulated annealing metaheuristic to obtain a function for decline trend analysis in hydrocarbon production, particularly useful for forecast purposes. This function has the following form:

$$f(t) = \alpha e^{\beta t} + \Phi(t). \tag{6}$$

In (6), we can distinguish an exponential part (taken from typical DCA analysis) followed by a general expression. More precisely, $t$ stands for time-dependent variable, $\{\alpha, \beta\}$ defines the exponential part, and $\Phi(t)$ represents a correction that improves fitting on the time series. In our current implementation, $\Phi(t)$ is a polynomial of degree $n$ defined as follows:

$$\Phi(t) = \sum_{i=0}^{n} a_i t^i. \tag{7}$$

Our algorithm improves fitting of (6) by exploring search space for the solutions $S = \{\alpha, \beta, a_0, a_1, \ldots, a_n\}$. SAM-oil performs a stochastic walk in order to find new solutions $S$; a process based on simulated annealing accepts or rejects new solutions until the best one is found. The obtained solution represents the best model for (6) in terms of lowest values for both fit error for training data and forecast error for validation data. Search for new solutions revolves around a $P_i$ population of solution vectors where $P_i = \{S_{i0}, S_{i1}, \ldots, S_{i(k-1)}\}$ and $k$ is the population size. The algorithm creates subsequent offspring population $P_j$ with new solutions $S_j$ by performing crossover and mutation operations over the solution traits $\{\alpha, \beta, a_0, a_1, \ldots, a_n\}$, like in genetic algorithms approach. Referring to Algorithm 2, when a new population $P$' has been generated, individual $S_j$ solutions are evaluated with respect to some criteria; in case of fail, they still have a chance to be part of the next generation according to the Boltzmann probability distribution. Diagram of Fig. 4 describes how SAM-oil searches for the best solution for (6):

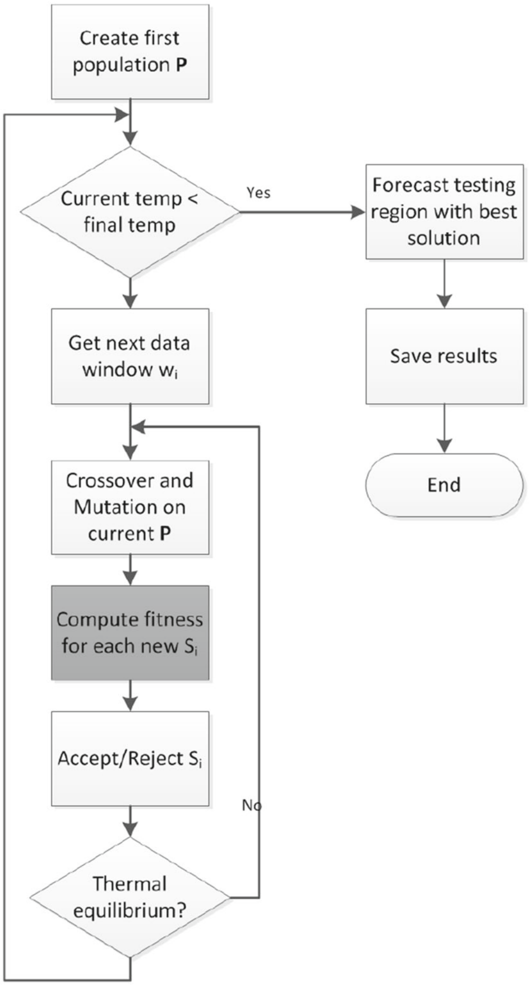
Create first population **P**
Current temp < final temp
Yes
Forecast testing region with best solution
Save results
End
Get next data window $w_i$
Crossover and Mutation on current **P**
Compute fitness for each new $S_i$
Accept/Reject $S_i$
No
Thermal equilibrium?

**Fig. 4** General procedure for incremental refinement of decline curves

***Initialization.*** A least-squares regression pass obtains the first $\alpha_0, \beta_0$ parameters over the whole training data. The algorithm sets random values (in the range [-1,1]) for the remaining parameters $\{a_0, a_1, \ldots, a_n\}$. In this way,

$$S_{Init} = \{\alpha_0, \beta_0, a_{00}, a_{01}, \ldots, a_{0n}\}$$

constitutes a first solution which models (6). In order to create the first population $P_0 = \{S_{00,} S_{01,} \ldots, S_{0(k-1)}\}$, $S_{Init}$ mutates $k$ times to create each $S_{0j}$, $j = 0, 1, \ldots, k-1$. The simulated annealing process establishes values for initial and final temperature according to the maximum/minimum deterioration criteria [38] for the cost function (fitness) of solutions.

***Crossover and mutation operations over current population.*** For each loop in the external cycle of annealing (for a given temperature), the algorithm obtains a current window $w_i$ in the time series and operates over the current population to obtain the next one. SAM-oil implements a one-point crossover operator and selects parents using the roulette criterium. Our algorithm accepts crossover rate as an input; however, our experiments use a value of 0.8. Regarding mutation operator, its rate value changes along the simulated annealing process. Higher mutation rates correspond to high temperatures and the mutation rate decreases geometrically to a minimum value. At the beginning of the annealing process (higher mutation rates), mutation operations perform a wider exploration of solution space. At final stages of annealing (lower mutation rates), SAM-oil performs less mutation operations, looking for better solutions only around a possible optimum point. Mutation rates for SAM-oil range between 0.9 and 0.25. For each trait or gene, SAM-oil measures mutation impact along the whole search process. After crossover and just before mutation, the algorithm computes the parents average fitness for each $S_j$. After mutation, the algorithm determines the value for improvement or deterioration between $S_j$'s fitness and his parents. In this way, we can track the improvement/deterioration of accumulated fitness for each mutated gene (Fig. 5). We describe later a strategy for improvement of solutions, based on this structure.

***Solution fitness.*** Measurement of fitness or quality for new $S_j$ solutions computes values for two features (highlighted block in Fig. 4):

1. $S_j$'s goodness of fit, relative to current $w_i$.
2. $S_j$'s forecast accuracy, relative to another region (independent from $w_i$) of the time series –a pre-testing window.

To estimate such features, the algorithm can use the mean absolute percentage error (MAPE) or the symmetric mean absolute percentage error (SMAPE), two well-known measures in forecasting applications, defined as

$$MAPE = \frac{1}{n}\sum_{i=1}^{n} \frac{|y - \hat{y}|}{|y|}$$

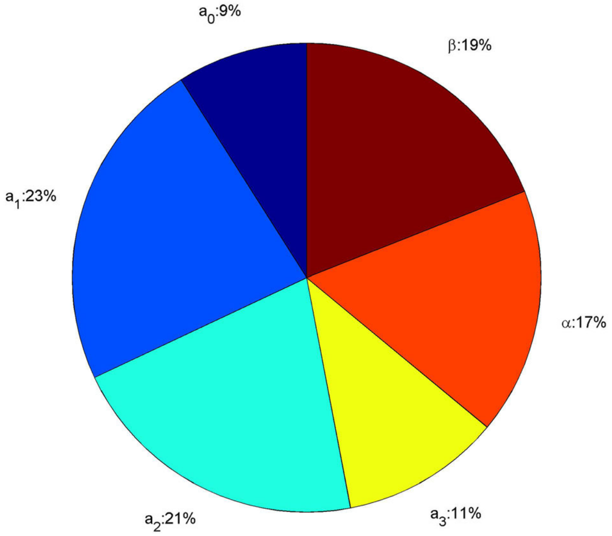

**Fig. 5** Relative impact for each mutated gene on fitness improvement or deterioration

**Algorithm 4** Computation for Confidence Percentage measure

```
For each point $i$ in current window
  Compute $\sigma$, the square root of the conditional variance
      for that point
  Obtain adjusted value $\hat{y}_i$ with current solution $S_j$ of (\ref
      {eq:ecuacionGeneral})
  If $y_i - K\sigma \le \hat{y}_i \le y_i + K\sigma$
    Increment counter for hits
  End
End
Compute Confidence Percentage as proportion between hits and
     total number of points in window
```

$$SMAPE = \frac{1}{n}\sum_{i=1}^{n} |\frac{y - \hat{y}}{y + \hat{y}}|$$

where $y$ and $\hat{y}$ are the actual and forecasted values, respectively. In addition, we use another measure, which we call Confidence Percentage (CP). Assessing of quality for new $S_j$ solutions takes into account such measures by themselves or some combination of them. In particular, Confidence Percentage estimates goodness of fit, related to the training data, by computing the proportion between well-adjusted values and the total number of training points, where a "well-adjusted" means that the fitted value lies within a region around the real value, defined by a margin of tolerance (Algorithm 4). For forecasting accuracy in per-testing window, the Confidence Percentage value is analogously computed. Parameter $K$ represents a tolerance margin for hits; different values will lead to more relaxed or stricter solutions.

SAM-oil searches for solutions with good fit relative to training data and good forecast accuracy relative to pre-testing data. In terms of an optimization problem, the objective function to be minimized consists of some type of measurement for fit errors (relative to training data) of $S_j$ combined with some measurement of forecast accuracy (relative to pre-testing data). By doing so, we hypothesize that the best solutions will have high accuracy on real forecast tests. The optimization problem can be stated as follows:

$$Min\, I_1(S_j, w_i)$$
$$s.t.\ Min\, I_2(S_j, w_k).$$

In this formulation, $w_i$ and $w_k$ are current windows of training and pre-testing, respectively. All functions $CP$, *MAPE* and *SMAPE* can measure both $I_1$ (fitting error for training) and $I_2$ (forecast error for pre-testing). Therefore, the fitness (the final cost value) for some $S_j$ is computed as a linear combination, such as

$$p_1 * I_1 + p_2 * I_2 \tag{8}$$

where $p_1$ and $p_2$ are weights such as $p_1 + p_2 = 1$. In the algorithm, $I_1$ and $I_2$ functions work together in such a way that enforce a more rigorous acceptance of new solutions, where $I_1$ poses the learning process and $I_2$ the testing process, respectively. In this way, the algorithm looks for solutions with real chances of making good forecasts because $I_2$ measures solution forecast accuracy in some pre-testing region. Components $I_1$ and $I_2$ for fitness function (8) could also be computed by additional measures, such as RMSE, R-squared or some portmanteau test for autocorrelation of residuals (e.g., Box-Pierce or Ljung-Box tests). In this way, SAM-oil could explore the solution space with a more diverse family of metrics. However, at this stage of the implementation, we have opted for the aforementioned *CP*, *MAPE*, and *SMAPE* measures.

***Acceptance/rejection for $S_j$ solutions.*** Once the algorithm creates a new population of $S_j$ solutions and computes their fitnesses, in the next step it performs an evaluation to decide if they will pass to the next generation. SAM-oil defines three criteria for making that decision, which are 1) current best fitness, 2) global fitness average and 3) global fitness average plus threshold value. A $S_j$ solution is accepted if its fitness is less than the reference value. If this condition is not met, the algorithm computes $\Delta = (Fitness(S_j) - reference\, value)$ in order to apply the Boltzmann criterium, accepting $S_j$ if a random value is less than $e^{-\Delta/c}$ and rejecting it otherwise ($c$ is the current temperature in the annealing process). In case of $S_j$ is rejected, the algorithm creates a new random solution to complete the next generation.

***Strategies for solutions improvement.*** As stated before, SAM-oil measures impact for each mutation relative to both improvement and deterioration for fitnesses. The structure depicted in Fig. 5 shows the improvement relative percentage of mutation for all genes; the higher the value, the respective gene mutation should produce a better fitness value. SAM-oil uses this structure as follows. At some point of the annealing process, SAM-oil detects potential local optima (the last solutions show no improvement). In that case, the method applies a "shock" of multiple mutation for creating the next population, i.e., while in "normal" mutation a single gene mutates, in multiple mutation several genes can mutate at the same time. A roulette-type selection based on the structure of the Fig. 5 determines which genes will mutate. As can be seen, genes with higher improvement relative percentage have more chances to mutate. In addition to multiple mutation, SAM-oil applies reheating to the annealing process [39], by resetting the temperature value to some higher $C$, where 1) $C = \sum_{i=1}^{n} c_i$, the average of all temperatures in the annealing process, 2) $C = \prod_{i=1}^{n} c_i$, the geometric mean of the same temperature values or 3) $C = 2c_{best}$, where $c_{best}$ is the process temperature when the current best solution was found.

### 3.3 Residual Modeling for Refinement of Forecasting

The final step of SAM-oil method deals with residuals computed by the model obtained for (6) in the previous step. In this way, the method looks for more accuracy in final forecasts. The time series of residuals $\{d_t\}$, such that

$$d_t = \hat{y}_t - y_t = (\alpha e^{\beta t} + \Phi(t)) - y_t\ t = 1, 2, \ldots n$$

is modeled by some space state model identified by the methodology described in [40]. The best space state model is identified by optimizing smoothing parameters and the initial state in formulation of (1) and (2). In the experiments section, we provide details of application of such methodology. Once the method obtains a $\Theta$ model of residuals, final forecasts over a test interval are computed through a linear combination with the model of (6) according to

$$\hat{y}_t = \alpha e^{\beta t} + \Phi(t) + \hat{d}_t, \tag{9}$$

where $\hat{d}_t = \Theta(t)$, the error value estimated by the $\Theta$ model.

### 3.4 Remarks on SAM-Oil Implementation

In order to search for the best model of (9), inputs of the SAM-oil method include three groups of parameters: filtering, decline curve modeling, and residuals modeling. The main parameter of filtering is the margin of tolerance for deviations that is used to recognize outliers. Parameters for decline curve modeling (Table 1) include type of reheat, length of pre-testing region, degree of $\Phi$, among others. We

**Table 1** SAM-oil parameters

| Feature | Choices |
|---|---|
| Tolerance margin for hits in *CP* measure | Fixed |
| Degree of polynomial $\Phi$ | {1, 2, 3} |
| Length of per-testing region | Length of test region, half the length of test region |
| $p_1$ and $p_2$ weights for fitness evaluation | Fixed |
| Temperature reset type for reheat | Arithmetic mean, geometric mean, non-monotonic assignment |
| Population size | Fixed |
| Acceptance/rejection criteria for $S_j$ in populations | Current best fitness, Global fitness average, Global fitness average plus threshold value |
| Point for no improvements detection | Fixed (percentage of total temperature cycles) |
| Maximum number of multiple mutation shocks | Fixed |

provide details of the parameters for residuals modeling in the experiments section.

Observe that the model of (9) is inspired on the time series decomposition approach [21]. In particular, this additive model leverages the knowledge on the exponential decline trend, commonly found in oil production. SAM-oil implicitly searches for another components of the decomposition, i.e., the seasonal and cycle terms, by abstracting them within $\Phi(t)$; the error term is captured by $\hat{d}_t$.

Finally, observe that $p_1$ and $p_2$ weights are fixed. Given the relevance of these weights, SAM-oil could be tuned even more by implementing an external layer which searches their best values. In this way better forecasts could be expected; however, for the sake of fast runs, we currently assign for them the values of 0.8 and 0.2, respectively. Additionally, that external layer could include the tuning of the components $I_1$ and $I_2$ with different combinations of RMSE, R-squared, Box-Pierce or Ljung-Box functions.

## 4 Experimental Results

Our experiments include a set of 50 time series of monthly oil production. The distinctive feature of this case study is that all the wells active at a certain time period are considered, including those hard to predict. Features of this difficulty include: small data sets, spikes, abrupt changes and gaps in the data. Figure 1 illustrates these features for a subset of the tested data. For each time series, we have run a 12-step ahead forecasting; we have taken out the last 12 points in each case for testing purposes. Thus, SAM-oil and the other tested methods have available $n-12$ points for modeling, where $n$ is the size for each time series. In the case of SAM-oil, modeling data include training and pre-testing segments. As mentioned, SAM-oil method measures fit goodness of model respect to training and uses pre-testing data for obtaining "preview" forecasts in order to improve model accuracy. The stochastic nature of SAM-oil, derived from the use of simulated annealing and genetic operators, requires us to run multiple trials of forecasts. Hence, SAM-oil forecasts 30 times each time series and calculates the average SMAPE measure for the forecast accuracy as

$$SMAPE = \frac{1}{30}\sum_{i=1}^{30}\left[\frac{1}{n}\sum_{j=1}^{n}\left|\frac{y_j-\hat{y}_j}{y_j+\hat{y}_j}\right|\right] \quad i = 1, 2, .., 30.$$

According to Table 1, many combinations of input parameters for SAM-oil can be defined, so a full, exhaustive run can take a long time. In order to define method settings, we performed a fast sampling over settings space and took the best ones for final experimentation. In respect to SAM-oil's modeling of residuals, same fast sampling leads us to determine that their behavior is well suited to state space models which takes into account trend. Specifically, we have used a space state model for residuals such as

$$\ell_t = \ell_{t-1} + \phi b_{t-1} + \alpha e_t \tag{10}$$

$$b_t = \phi b_{t-1} + \alpha\beta e_t \tag{11}$$

where $\ell_t$ stands for the level of time series for residuals and $e_t \sim NID(0, \sigma^2)$, $\alpha$ is the series smoothing parameter, $\beta$ is the trend smoothing parameter and $\phi$ is a trend damping parameter. Such specification corresponds to an exponential smoothing model with a damped additive trend, a popular and very successful method for time series automatic forecasting [41]. In this way, $\alpha$, $\beta$, $\phi$ are considered as additional inputs to SAM-oil and supplement Table 1; current SAM-oil implementation uses fixed values of 0.5 for each one.

The results (Table 2) include forecast errors values for the 50 time series, according to the SMAPE measure. The bolded value corresponds to the lowest error, obtained by SAM-oil. For comparison purposes, the table also shows the results of the forecasts computed by the following methods:

**Table 2** Forecasting accuracy of SAM-oil and other methods

| Method | SMAPE |
|---|---|
| DCA | 0.3209 |
| ARIMA | 0.3043 |
| NARX | 0.1611 |
| Single SE | 0.1727 |
| Double SE | 0.6540 |
| Damped trend SE | 0.3233 |
| SAM-oil | **0.1586** |

- Decline curve analysis: Assuming an exponential decay trend, we obtained a linear model for each time series, using time versus logarithmic values of monthly oil production. The obtained regression function computes the forecasting for the next 12 points.
- ARIMA: Models for each time series are automatically obtained with the function *auto.arima* included in the *forecast* package of R.
- NARX: A neural network has been trained with an structure including one hidden layer with 25 neurons, a hyperbolic tangent sigmoid transfer function in the hidden layer, and a linear transfer function in the output layer.
- Single exponential smoothing: The traditional model was implemented here using an adaptive smoothing parameter.
- Double exponential smoothing: We implemented the classical approach defined before in (3) and (4).
- Damped trend exponential smoothing: The implementation used here corresponds to the classical model defined by (10) and (11).

Except for the decline curve analysis, which in this work is considered as the reference method in the oil industry, all the other methods have undergone some tuning, in order to improve their performance. In this way, we looked for fairness in the results comparison. For example, the NARX neural network has been experimentally tuned using the Neural Network Toolbox from MATLAB. In our implementation of single exponential smoothing, the smoothing parameter is computed as $\alpha = \mid y_t - \hat{y}_t \mid / y_t$ for each $t$ in the modeling stage, in order to adapt itself to the time series changes. For the double exponential smoothing, an algorithm based on simulated annealing searches for the optimum $\alpha$, $\beta$ parameters (the smoothing and trend parameters, respectively). Our implementation of damped trend exponential smoothing searches for the optimum parameters $\alpha, \beta, \phi$ (the smoothing, trend and damping parameters, respectively) also using a simulated annealing algorithm. The ARIMA results take advantage from the forecast package of R, which automatically fits the best model [40].

Because of the stochastic nature of implementations for SAM-oil, double exponential smoothing and damped trend exponential smoothing methods, 30 replications were performed in each case. In this way, SMAPE measures given for those methods correspond to average values.

### 4.1 Time Series Clustering

Due to the high dispersion of the errors among the sampled time series (ranging from 0.0121 to 0.5747 in the case of SAM-oil, for example), a further analysis for these time series drove us to identify three clusters, which we mapped to three levels of forecasting difficulty: easy, moderate and hard.

**Table 3** Time series features for analysis

| | |
|---|---|
| Autocorrelation (lag 1) | Step change percentage |
| Partial autocorrelation (lags 1,2,3,6,9,12) | Turning points percentage |
| Skewness | Runs test |
| Kurtosis | Length, number of segments |
| Hurst exponent | Basic tendency |

**Table 4** Time series categorization by clustering

| Cluster | Skewness | Kurtosis | Step change (%) | Runs test | Segmentation ratio (%) |
|---|---|---|---|---|---|
| 1 | 0.726 | 0.746 | 0.042 | −13.302 | 0.1038 |
| 2 | 0.688 | 1.419 | 0.037 | −11.655 | 0.1140 |
| 3 | −0.712 | 4.118 | 0.135 | −6.036 | 0.1494 |

In the first step of this analysis, we collected measures of several features (Table 3) for each time series, as suggested in meta-learning approaches for selecting suitable forecasting methods (see [16,42,43]). Using all features targeted to the winner method for each time series, then we run an attribute selection algorithm in Weka. The result set of relevant features included partial autocorrelation values (lags 2,3,6,12), basic tendency, turning points percentage, step change percentage, and runs test. We use those features to feed a k-means clustering algorithm, also in Weka, in order to obtain those three clusters.

Table 4 shows average values for skewness, kurtosis, percentage of step changes and runs test in each cluster. The first two measures can be taken as a empirical evidence of complexity for time series [44]; in particular, kurtosis average values (as computed by STATISTICA and the package e1071 in R) indicate that clusters progressively move away from a normal distribution, where the time series from cluster 3 are the most peaked. However, we mainly consider here the step change and runs test values to assign a nominal category for each cluster, related to forecasting difficulty for the time series in each group. In this way, we want to 1) obtain a complexity characterization to the whole set of time series and 2) observe behavior for each method of Table 2 respect to each cluster. The average value of the runs test suggests that cluster 1 corresponds to the least random time series, i.e., the statistic more strongly rejects the null hypothesis of randomness; conversely, the respective value for the cluster 3 shows less evidence to reject the null hypothesis, so we consider its time series as the most random. The percentage of step changes quantifies the structural breaks; as can be seen, time series in cluster 3 presents more breaks with 13.5 % of their points, in average. Therefore, according to values for step changes percentage and runs test, we label cluster 1 as “easy” to forecast and cluster 3 as “hard”, with cluster 2 between them.

**Table 5** Forecast results per cluster/method

| Method | Mean SMAPE | | |
|---|---|---|---|
| | Easy | Moderate | Hard |
| DCA | 0.2592 | 0.3038 | 0.4345 |
| ARIMA | 0.2012 | 0.1638 | 0.6496 |
| NARX | **0.1328** | **0.1496** | **0.2183** |
| Single SE | **0.1113** | 0.1889 | 0.2401 |
| Double SE | 0.4066 | 0.5161 | 1.2065 |
| Damped trend SE | 0.2628 | 0.4822 | **0.1915** |
| SAM-oil | **0.1346** | **0.1456** | **0.2116** |
| Average | 0.2155 | 0.2785 | 0.4503 |

We include an additional average value for each cluster in Table 4, which reinforces the categorization assigned to each cluster. The last column shows the average of the ratio between segments and the length of each time series; in order to obtain this metric, we implemented an algorithm to search for the optimum number of the contiguous, decline exponential segments along the production span of each time series. Based on simulated annealing, the algorithm defines for a time series an initial solution with contiguous random segments, and iteratively searches for new ones varying both their number and length; the optimum solution minimizes the average R-squared of the exponential curves corresponding to the contiguous segments found. Cluster 1 has the lowest segmentation ratio and cluster 3 the greatest, with cluster 2 between them. In this way, we believe that with respect to this metric, time series of cluster 3 are the hardest ones for forecasting because a more significant heterogeneity in the oil production along time.

Table 5 shows details and results of forecasting in each cluster. The lowest mean SMAPE values for each cluster, among all the methods, are highlighted in bold. Average forecasting values for all of the methods seem to support the labeling for each cluster: the lowest average SMAPE corresponds to the "easy" cluster, while the largest corresponds to the "hard" cluster. SAM-oil improves DCA's accuracy in all the clusters and along with NARX seems to lead each group. More precisely, SAM-oil leads the methods in the "moderate" cluster, while it comes second in the "hard" cluster and third in the "easy" cluster. Therefore, we assert that these results support the reliability of SAM-oil concerning these time series since the method performed well on different levels of forecasting difficulty.

We assert that there are significant differences in the mean performance of all the methods based on a $p$ value $= 6.01e-06$ from a Friedman's test. Furthermore, we run a Bonferroni's post hoc test which determined a significant difference between DCA and the rest of the methods. Figure 6 compares the mean ranks for each method and shows that SAM-oil's performance is very competitive in respect of the other tested methods. As a matter of fact, SAM-oil ranks first for the 22 % of the whole set of time series, followed by double exponential smoothing and simple SE (Fig. 7).

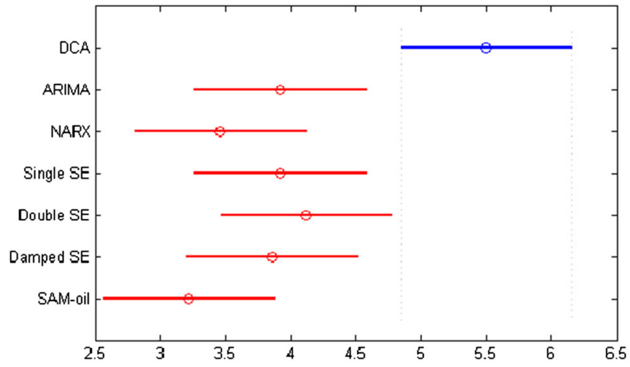

**Fig. 6** Mean ranks of forecasting methods

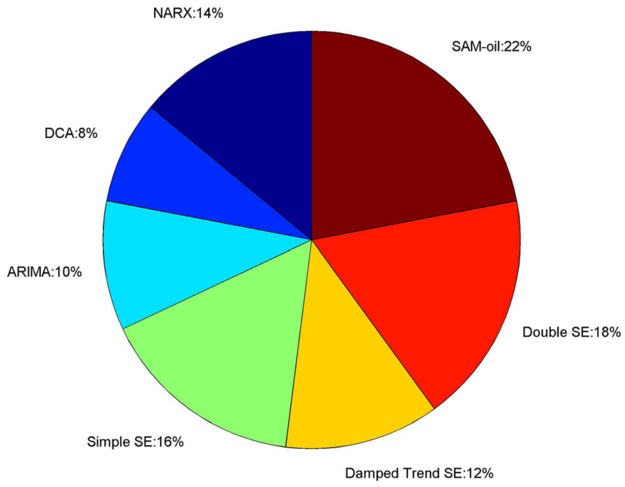

**Fig. 7** Percentage of wins of forecasting methods

### 4.2 Final Remarks on Forecasting with SAM-Oil

Figure 8 illustrates the 12-step ahead forecasting for a particular instance from each of these clusters. In this sample, the mean SMAPE (obtained among all of the forecasting methods) for the "easy" time series was 0.0836, while for the "moderate" one was 0.1402 and for the "hard" one was 0.4030. In this sample, we have included the three best methods for performance (SAM-oil, NARX, and Single SE) and the reference method in oil industry (DCA).

In spite of the well-focused context of this study case (monthly oil production), we claim that SAM-oil conforms a framework which can be exploited in other forecasting domains. According to the No Free Lunch theorem [31], there is not a universal method to forecast all domains. Therefore, in order to find the best forecasting model for the oil production time series, we take advantage of the knowledge about

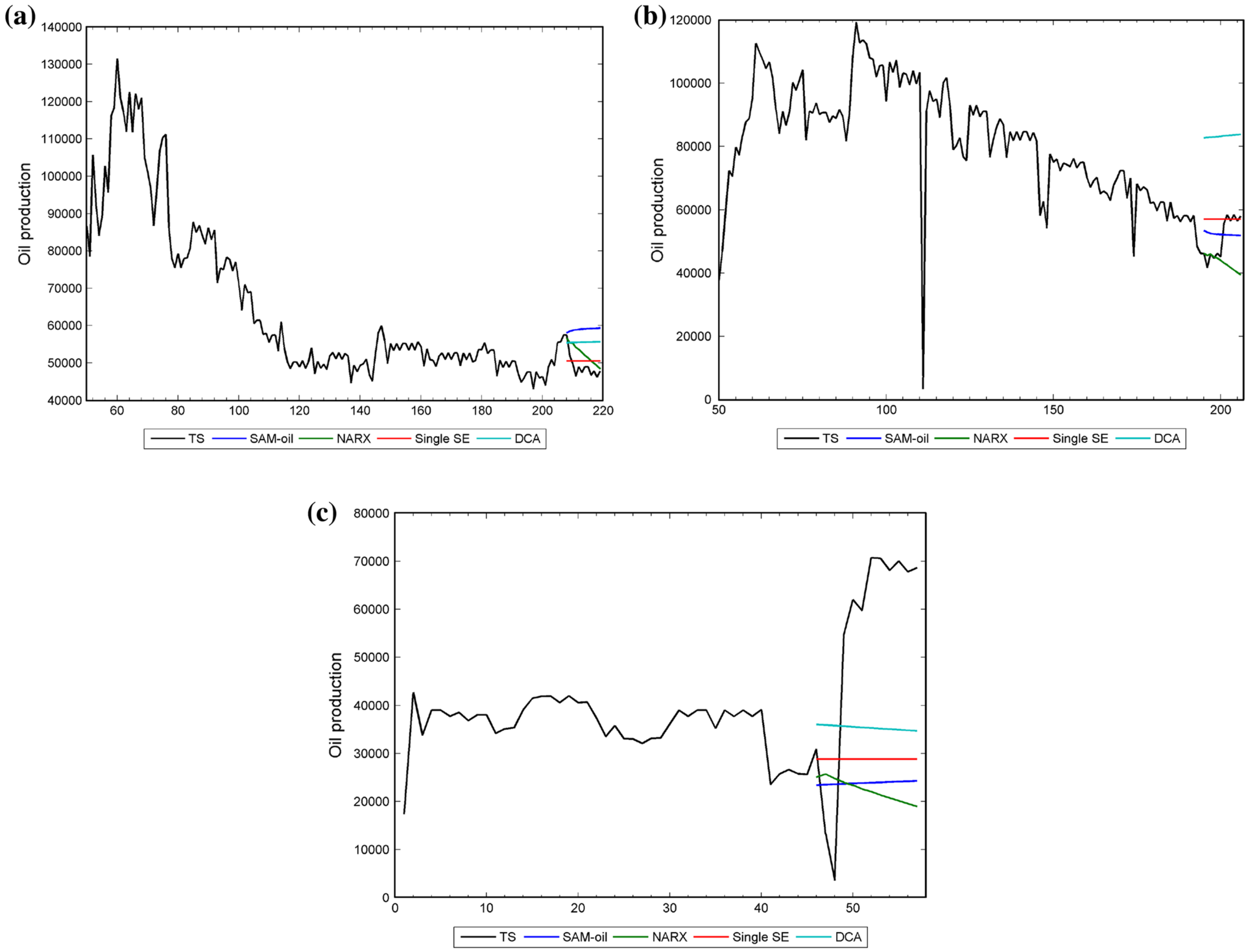

**Fig. 8** 12 step-ahead forecasts of oil production: **a** a TS from the "easy" cluster, **b** a TS from the "moderate" cluster, and **c** a TS from the "hard" cluster

their possible decline behavior. In this way, the applicability of the method in other domains (e.g. pricing, demanding, supplying, etc), would require to redefine the general forecasting model, specifically the objective function (6), with different problem-specific information.

## 5 Related Work: TS Forecasting in Petroleum Engineering

In the following, we analyze the main advantages and drawbacks of the reported methods from the point of view of their application to oil production TS forecasting and compare them with the method described in this paper. This analysis is also used to justify the selection of the reference models for our case study.

### 5.1 Forecasting Based on Traditional Techniques

The most common family of methods is derived from Autoregressive (AR) approach, which measures correlation entries in the series. Box and Jenkins [5] proposed Autoregressive Moving Average (ARMA) models; they compute correlation among lagged values of the series. ARIMA, an extension of ARMA for non-stationary time series, is a very popular forecasting tool in many areas of science [6]. An ARIMA model forecasts future values of a time series through a linear combination of past observations and random errors. Equation (12) defines an ARIMA($p$,$d$,$q$) model

$$\begin{aligned} y_t &= c + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \ldots \phi_p y_{t-p} + e_t \\ &\quad - \theta_1 e_{t-1} - \theta_2 e_{t-2} - \cdots - \theta_q e_{t-q}. \end{aligned} \tag{12}$$

Integers $p$ and $q$ represent the autoregressive and moving average orders, respectively; $d$ is the degree of differentiation required for the time series becomes stationary; $\phi_i$ $(i = 1, 2, \ldots, p)$ and $\theta_j$ $(j = 1, 2, \ldots, q)$ are model parameters; $\{e_t\}$ are random errors considered as independently and identically distributed with zero mean and $\sigma^2$ variance. To find the best model, Box and Jenkins developed a methodology consisting of an iterative process, with steps for model identification (i.e., $p$ and $q$ values), estimation of parameters and model validation. In the identification step, time series must become stationary by applying a process of differentiation, which requires knowledge of the value of $d$. In other words, degree $d$ of differencing is determined, usually by means of calculation of autocorrelation function value. The methodology computes $p$ and $q$ values from observations, by means of autocorrelation and partial autocorrelation functions. In the estimation step, once a potential model is defined, an error minimization process calculates the remaining model parameters, typically by using either least-square errors or maximum likelihood. The final step of the methodology verifies properties for model errors, which must be independent with constant mean and variance. Failure to comply with these properties entails a new iteration of the methodology.

An application of the ARIMA method for monthly oil production forecasting is described in [8]. The work was applied on time series coming from twelve wells. The time series from the study case included at least 60 months of production. For each well, the last 10 months were kept apart in order to measure accuracy of forecasts, as well as to determine the remaining crude oil reserves. The applied ARIMA modeling required removing of trends by means of differencing and data log transformations (where needed). Different ARIMA models were obtained for each well; adequacy of each model was checked by means of residual analysis tests. The authors concluded, for the study case, that ARIMA method provided better results than decline curve analysis. Moreover, ARIMA obtained good results when decline curve analysis failed (e.g. for waterdrive reservoirs). Another application is reported in [9], whose study case consists on forecasting three leading months in Malaysia, given a production period from January 2005 to May 2010. The authors identified two tentative models, ARIMA(1,0,0) and ARIMA(2,0,0), which were evaluated by AIC and BIC criteria. However, the authors do not provide much more details about the validity of forecasts. Finally, a related work in [7] compares different models to forecast the production of different fuel fossil sources (oil, gas natural, hard coal) in Turkey, on an annual basis. The time series included 54 observations from 1950 to 2003; the authors conclude that, in particular, a seasonal ARIMA (SARIMA) model is the best for oil production forecasting.

As we can see, the ARIMA model has proved to be a very useful tool especially for short-term forecasting. Being an extrapolation method, it requires only the historical time series and does not assume any particular pattern in the historical data. Nevertheless, the model is quite difficult to develop and apply as it involves transformation of the variable, identification of the model, estimation through nonlinear method, verification of the model and derivation of forecasts. On the other hand, the uncertainty of a prediction is quite different for stationary and non-stationary models. In a stationary model, the long-term variance of the prediction converges at a constant value as a consequence of the long-term prediction being the mean of the process. For non-stationary models of the series, the long-term uncertainty of the prediction increases without limit. This means that in the long-term we cannot predict the behavior of a non-stationary process. The experimental results reported in the previous section are consistent with this fact.

Exponential smoothing techniques were conceived as extrapolation methods for univariate time series. However, in the last 30 years have been widely used in forecasting, being able to deal with nonlinearities, discontinuities and constraints [6,41]. We described the basics of this method in Sect. 2.1.1. Since the simplest form of exponential smoothing (single) has a "flat" forecast function, it can generate suitable forecasts for time series with no trend or seasonal pattern [21]. If this is not the case, double and damped trend smoothing should be used. As an example of the applications to oil production forecasting, exponential smoothing has been used in the Oilfield Production Forecasting toolkit developed for Saudi ARAMCO National Oil Company [45]. As in the case of regression-based techniques, smoothing shows good short-term accuracy but is only valid if we assume that the error terms are random.

### 5.2 Forecasting Based on Computational Intelligence (CI) Techniques

Another group of methods is based on the use of artificial neural networks (ANN), pattern recognition and metaheuristic techniques. ANN models are most frequently used for oil production prediction. Neural networks are nonparametric and nonlinear models which make no assumptions on the underlying behavior of time series. Theoretically, neural networks can approximate any complex function with arbitrary accuracy given a large enough structure [46]. The Focused Time-Delay Feedforward Networks (FTDNN) and the dynamically driven recurrent networks (RNN) are two types of neural networks successfully used in forecasting applications.

A FTDNN is a dynamic, feedforward network with a $p$-order delay line in the input. At time $n$, the vector $x(n) = [x(n), x(n-1), \ldots, x(n-p)]^{\intercal}$acts like the temporal input pattern or the short-term memory, where $x(t)$ $t = 1, 2, \ldots$ are the time series observations. In the training stage, the network parameters are adjusted to minimize some error measures between its output and the time series actual values, typically by means of a standard backpropagation algorithm. FTDNN with a multi-layer backpropagation learning algorithm often called a multi-layer perceptron (MLP) can be applied directly to problems of production TS forecasting provided the data is suitably pre-processed (pre-processing of input data includes feature extraction, dimensional reduction, outlier rejection etc.). He et al. [47] describe a back-propagation ANN forecasting model to predict existing and infill oil well performance using only production data (without any reservoir data). The experiments included two data sets (for existing and infill wells prediction) of nine wells each from a low permeable carbonate reservoir. The primary production history data were used to train and test (1.5 year period) the neural networks. For the cumulative production, for testing data points, correlation coefficient of 0.99 and mean percentage error of 4.57 % were obtained. For the 3-month interval production output, a correlation coefficient of 0.93 was obtained for testing data points, while the average percentage error was 22.82 % for training data set and 16.06 % for testing data points. The model shows relatively good predictive capacities for short-term prediction.

The dynamically driven recurrent networks have shown better results for long-term oil prediction. Unlike feedforward neural networks, RNNs capture the time series autocorrelation structure through local or global feedback loops. A specific type of these recurrent networks are the nonlinear autoregressive models with exogenous inputs (NARX), in which the feedback loop is limited to the output neuron. NARX computes the next value for the time series with the previous output values and the previous values of an independent, exogenous input signal and can be mathematically modeled as [48]

$$y(n+1) = f(y(n), \dots, y(n-d_y+1), u(n-k), \dots, \\ u(n-k-d_u+1)) \qquad (13)$$

where $u(n) \in \mathbb{R}$ and $y(n) \in \mathbb{R}$ denote the input and the output at time $n$+1, and $d_u \geq 1$, $d_y \geq 1$ and $d_u \leq d_y$ are the input memory and output memory orders, respectively, and $k \geq 0$ is the delay parameter. The nonlinear $f(\cdot)$ function can be approximated with a feedforward network.

In recent years, the use of NARX networks for forecasting applications has attracted the attention of the researchers [49,50]. In [51], the authors analyze several aspects of oil production forecasting with NARX networks: i) the applicability to the univariate and multivariate forecasting, ii) different topologies of the networks, and iii) the application of clustering techniques to improve forecasting results. The described case studies come from the naturally fractured reservoir with primary production and under gas injection for pressure maintenance. The NARX model from that work is used as a reference for the SAM-oil method.

The use of another type of ANN, higher-order neural networks (HONN), to forecast production of water, oil and gas is reported in [10]. That type of neural networks has a good capacity to capture nonlinear relationships between inputs and weights of the network. Two case studies illustrate the behavior of the proposed model with i) only one dynamic parameter data, oil production data, and ii) three dynamic parameter data, oil, gas and water production data used for forecasting. The oldest well of the field was used for the case study, for which 9-year production history was used with 77 data points for training and months 78–94 (the last 16 months) to validate HONN models. The authors report similar accuracy results but it should be noted that HONN was used as a one-step ahead predictor, which is impractical in most of the oil forecasting applications.

Recursive ANN has also been applied to oil prediction. Unlike feedforward neural networks, RNNs can use their internal memory to process arbitrary sequences of inputs. In [51], the authors analyze several aspects of oil production TS forecasting applying recurrent NARX networks: i) the applicability of the univariate and multivariate forecasting, ii) different topologies of the recurrent NARX networks, and iii) the application of clustering techniques to improve forecasting results. The described case studies come from the naturally fractured reservoir with primary production and under gas injection for pressure maintenance. This is the NARX model that is used as a reference for the SAM-oil method.

In general, one of the most evident difficulties in applying ANN-based forecasting models is that manual tuning of network parameters can affect considerably the accuracy of the prediction. To address this problem, other CI techniques are usually used. Saemi et al. [52] proposed to use a Genetic Algorithm (GA) to determine the number of neurons in the hidden layers, the momentum and the learning rates for minimizing the time and effort required to find the optimal architecture. Such hybrid approaches could further enhance the performance of ANN by choosing appropriate ANN parameters, e.g. number of layers, nodes in the hidden layer, training algorithm, etc. (see for example [53–55]).

In [13], the use of associative memories for monthly oil production forecasting is proposed. The authors propose a novel nonlinear forecasting technique based on the Gamma classifier inspired by Alpha-Beta associative memories. Besides tests on benchmark time series, the proposed method's performance is also analyzed on different input segments of 12, 18 and 24 months, over a sample of six oil production time series. Both for the benchmark and oil production time series, the technique reports better forecasting accuracy than other reference methods with average SMAPE of 15.72 % for the selected case study of oil monthly production. The use of that model for longer time intervals (more than 12 months) is the topic of further research work.

Some other CI techniques like matrix memories and Support Vector Machines (SVM) have also proved their effectiveness in solving forecasting problems. For example, in [56], a SVM is built to predict the global oil production in year 2011; penalty and gamma parameters for the SVM model are tuned by a greedy-search algorithm and a genetic algorithm. Nevertheless, to our knowledge, SVM has never been used for long-term prediction of monthly oil production.

## 6 Conclusions

In this paper, we present a new method to forecast monthly oil production. This work can be considered as an alternative or supplement to traditional techniques of petroleum engineering. In particular, SAM-oil can be regarded as a method which complements decline curve analysis because the obtained models include an exponential decline curve as that obtained with traditional DCA. Furthermore, these models can be directly used as a forecasting tool. The work can be easily extended to both hyperbolic and harmonic decline cases.

SAM-oil's forecasting accuracy in the whole set of experimental time series, and inside each found cluster, outperforms well-established forecasting models such as ARIMA and some models from the exponential smoothing family. Only in cluster 1, the single exponential smoothing outperforms SAM-oil significantly. This result agrees with conclusions from M3 Competition [57]: simpler methods can provide more accurate forecasts than more sophisticated ones. On the other hand, results show that SAM-oil outperforms DCA approximately by 50 % in each cluster; this percentage resembles the improvement proportion of SAM-oil respect of DCA for the whole set of time series (see Table 2).

SAM-oil forecasts are comparable to those obtained with a NARX neural network, a widely proven and well-established technique in the field of computational intelligence. Forecasts of both SAM-oil and NARX underperform those from single exponential models in cluster 1; however, their performance is the best in the clusters 2 and 3, the most difficult time series. As seen, SAM-oil slightly outperforms NARX in the experiments. Configuration of NARX has required a manual setup to find a well-tuned network, looking for a good structure definition, which includes number and size of hidden layers, type of transfer functions and input size. Conversely, SAM-oil settings only required a fast sampling over settings space. Furthermore, unlike the output models from NARX, SAM-oil output models have the advantage of being directly interpretable in terms of the traditional decline curve analysis.

Reliability of SAM-oil was established in the context of monthly oil production, since its good performance on forecasting time series with different levels of difficulty. The applicability of the method in other forecasting domains was profiled by suggesting the redefinition of the method's general model with different objective functions, which must take advantage of domain-specific knowledge. This applicability issue remains open here as future work.

**Acknowledgments** Authors Juan Frausto-Solís and Manuel Chi-Chim contributed equally to development of this article. Partial support for this work was provided by CONACYT-SENER-Hidrocarburos Project 146515. The authors acknowledge the support of the Mexican Petroleum Institute (IMP), the Ministry of Energy (SENER) and the National Council for Science and Technology (CONACYT) of Mexico.

## References

1. U.S. Energy Information Administration: International energy outlook 2013. Technical Report, U.S. Department of Energy (2013)
2. Rahuma, K.M.; Mohamed, H.; Hissein, N.; Giuma, S.: Prediction of reservoir performance applying decline curve analysis. Int. J. Chem. Eng. Appl. **4**, 74–77 (2013)
3. Makinde, F.A.; Orodu, O.D.; Ladipo, A.O.; Anawe, P.A.L.: Cumulative production forecast of an oil well using simplified "hyperbolic–exponential" decline models. Glob. J. Res. Eng. **12**(2), 27–40 (2012)
4. Chi, M.; Frausto-Solis, J.; Sheremetov, L.: Soft computing algorithms for nonlinear regression applied to decline curve analysis. In: Villa-Vargas, L., Sheremetov, L., Haasis, H.D. (eds) Workshop on Operations Research and Data Mining, ORADM 2012, Workshop Proceedings, pp. 210–217 (2012)
5. Box, G.E.P.; Jenkins, G.M.: Time Series Analysis: Forecasting and Control. Holden-Day, San Francisco (1976)
6. Gooijer, J.G.D; Hyndman, R.J.: 25 years of time series forecasting. Int. J. Forecast. **22**(3), 443–473 (2006)
7. Ediger, V.Ş.; Akar, S.; Uğurlu, B.: Forecasting production of fossil fuel sources in Turkey using a comparative regression and ARIMA model. Energy Policy **34**(18), 3836–3846 (2006)
8. Ayeni, B.J.; Pilat, R.: Crude oil reserve estimation: an application of the autoregressive integrated moving average (ARIMA) model. J. Pet. Sci. Eng. **8**(1), 13–28 (1992)
9. Yusof, N.M.; Rashid, R.S.A.; Zamzulani, M.: Malaysia crude oil production estimation: an application of ARIMA model. In: International Conference on Science and Social Research 2010, Kuala Lumpur, Malaysia, pp. 1255–1259 (2010)
10. Chithra Chakra, N.; Song, K.Y.; Saraf, D.N.; Gupta, M.M.: Production forecasting of petroleum reservoir applying higher-order neural networks (HONN) with limited reservoir data. Int. J. Comput. Appl. **72**, 23–35 (2013)
11. Chithra Chakra, N.; Song, K.Y.; Gupta, M.M.; Saraf, D.N.: An innovative neural forecast of cumulative oil production from a petroleum reservoir employing higher-order neural networks (HONNs). J. Pet. Sci. Eng. **106**, 18–33 (2013)
12. Darwis, S.; Ruchjana, B.N.; Permadi, A.K.: Robust decline curve analysis. J. Indones. Math. Soc. **15**(2), 105–111 (2009)
13. López-Yáñez, I.; Sheremetov, L.; Yáñez-Márquez, C.: A novel associative model for time series data mining. Pattern Recogn. Lett. **41**, 23–33 (2014)
14. Batyrshin, I.Z.; Sheremetov, L.: Perception-based approach to time series data mining. Appl. Soft. Comput. **8**(3), 1211–1221 (2008)
15. Chatfield, C.: Time-Series Forecasting. Chapman & Hall/CRC Press, Boca Raton, Florida (2002)
16. Wang, X.; Smith-Miles, K.; Hyndman, R.: Rule induction for forecasting method selection: meta-learning the characteristics of univariate time series. Neurocomputing **72**(10), 2581–2594 (2009)
17. Brown, R.G.: Statistical Forecasting for Inventory Control. McGraw Hill, New York (1959)
18. Holt, C.C.: Forecasting seasonals and trends by exponentially weighted moving averages. Technical Report (1957)
19. Winters, P.R.: Forecasting sales by exponentially weighted moving averages. Manag. Sci. **6**(3), 324–342 (1960)
20. Hyndman, R.J.; Koehler, A.B.; Snyder, R.D.; Grose, S.: A state space framework for automatic forecasting using exponential smoothing methods. Int. J. Forecast. **18**(3), 439–454 (2002)
21. Hyndman, R.; Koehler, A.; Ord, J.; Snyder, R.: Forecasting with Exponential Smoothing: The State Space Approach. Springer, Berlin (2008)
22. Taylor, J.W.: Exponential smoothing with a damped multiplicative trend. Int. J. Forecast. **19**, 715–725 (2003)

23. Rahuma, K.M.; Mohamed, H.; Hissein, N.; Giuma, S.: Prediction of reservoir performance applying decline curve analysis. Int. J. Chem. Eng. Appl. **4**, 74–77 (2013)
24. Fetkovich, M.; Vienot, M.; Bradley, M.; Kiesow, U. et al.: Decline curve analysis using type curves: case histories. SPE Form. Eval. **2**(04), 637–656 (1987)
25. Morehouse, D.F.: Intricate puzzle of oil and gas "reserves growth". Natural Gas Monthly, U.S. Department of Energy, Office of Oil and Gas, Energy Information Administration, vii-xx (1997)
26. Fetkovich, M.: Decline curve analysis using type curves. J. Pet. Technol. **32**(6), 1065–1077 (1980)
27. Mohaghegh, S.; Gaskari, R.: An intelligent system's approach for revitalization of brown fields using only production rate data. Int. J. Eng. **22**(1), 89–106 (2009)
28. Arps, J.: Analysis of decline curves. Trans. AIME **160**, 228–247 (1945)
29. Lyons, W.C.; Plisga, G.J.: Standard Handbook of Petroleum and Natural Gas Engineering, 2nd edn. Elsevier/Gulf Professional Publishing, Burlington, MA (2004)
30. Li, K.; Horne, R.N. et al.: A decline curve analysis model based on fluid flow mechanisms. SPE Reserv. Eval. Eng. **8**(03), 197–204 (2005)
31. Simon, D.: Evolutionary optimization algorithms. Wiley, Hoboken, New Jersey (2013)
32. Jayakumar, V.; Raju, R.: A simulated annealing algorithm for machine cell formation under uncertain production requirements. Arab. J. Sci. Eng. **39**(10), 7345–7354 (2014)
33. Kirkpatrick, S.; Gelatt, C.D.; Vecchi, M.P.: Optimization by simulated annealing. Science **220**(4598), 671–680 (1983)
34. Cerny, V.: Thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm. J. Optim. Theory Appl. **45**(1), 41–51 (1985)
35. Reyes, E.; Steidley, C.: Optimization using simulated annealing. In: Northcon/98 Conference Proceedings, pp. 120–126. doi:10.1109/NORTHC.1998.731523 (1998)
36. Holland, J.: Adaption in natural and artificial systems. University of Michigan Press, Ann Arbor, MI (1975)
37. Goldberg, D.E.: Genetic Algorithms in Search, Optimization and Machine Learning, 1st edn. Addison-Wesley Longman Publishing Co., Inc.,, Boston, MA (1989)
38. Sanvicente-Sánchez, H.; Frausto-Solís, J.: A method to establish the cooling scheme in simulated annealing like algorithms. In: Computational Science and Its Applications–ICCSA 2004. Springer, pp. 755–763 (2004)
39. Frausto-Solís, J.; Sánchez-Pérez, M.; Liñán-García, E.; Sánchez-Hernández, J.P.: Threshold temperature tuning simulated annealing for protein folding problem in small peptides. Comput. Appl. Math. 32(3), 471–482 (2013)
40. Hyndman, R.J.; Khandakar, Y.: Automatic time series forecasting: the forecast package for R. J. Stat. Softw. **27**(3), 1–22 (2008)
41. Hyndman, R.: Forecasting: Principles and Practice. OTexts, Heathmont, Vic (2014)
42. Lemke, C.; Gabrys, B.: Meta-learning for time series forecasting and forecast combination. Neurocomputing **73**(10), 2006–2016 (2010)
43. Prudêncio, R.B.; Ludermir, T.B.: Meta-learning approaches to selecting time series models. Neurocomputing **61**, 121–137 (2004)
44. Cristelli, M.; Zaccaria, A.; Pietronero, L.: Universal relation between skewness and kurtosis in complex dynamics. Phys. Rev. E **85**, 066,108 (2012). doi:10.1103/PhysRevE.85.066108
45. Holdaway, K.R.: Let oil and gas talk to you: predicting production performance. In: Proceedings of the SAS®Global Forum 2012 Conference (2012)
46. Zhang, G.P.: An investigation of neural networks for linear time-series forecasting. Comput. Oper. Res. **28**(12), 1183–1202 (2001)
47. He, Z.; Yang, L.; Yen, J.; Wu, C.; et al.: Neural-network approach to predict well performance using available field data. In: SPE Western Regional Meeting, Society of Petroleum Engineers (SPE-68801-MS) (2001)
48. Xie, H.; Tang, H.; Liao, Y.H.: Time series prediction based on NARX neural networks: an advanced approach. In: International Conference on Machine Learning and Cybernetics, vol. 3, pp. 1275–1279. IEEE (2009)
49. Menezes, J.M.P. Jr.; Barreto, G.A.: Long-term time series prediction with the NARX network: an empirical evaluation. Neurocomputing **71**(16), 3335–3343 (2008)
50. Yang, X.; Zhang, H.; Zhou, H.: A hybrid methodology for salinity time series forecasting based on wavelet transform and NARX neural networks. Arab. J. Sci. Eng. **39**(10), 6895–6905 (2014)
51. Sheremetov, L.; Cosultchi, A.; Gonzalez-Sánchez, A.; Martínez-Muñoz, J.; Jiménez-Aquino, M.: Data-driven forecasting of naturally fractured reservoirs based on nonlinear autoregressive neural networks with exogenous input. J. Pet. Sci. Eng. **123**, 106–119 (2014)
52. Saemi, M.; Ahmadi, M.; Varjani, A.Y.: Design of neural networks using genetic algorithm for the permeability estimation of the reservoir. J. Pet. Sci. Eng. **59**(1), 97–105 (2007)
53. Leung, F.H.F.; Lam, H.K.; Ling, S.H.; Tam, P.K.S.: Tuning of the structure and parameters of a neural network using an improved genetic algorithm. IEEE Trans. Neural Netw. **14**(1), 79–88 (2003)
54. Donate, J.P.; Li, X.; Sánchez, G.G.; de Miguel, A.S.: Time series forecasting by evolving artificial neural networks with genetic algorithms, differential evolution and estimation of distribution algorithm. Neural Comput. Appl. **22**(1), 11–20 (2013)
55. Sedki, A.; Ouazar, D.; El Mazoudi, E.: Evolving neural network using real coded genetic algorithm for daily rainfall-runoff forecasting. Expert Syst. Appl. **36**(3), 4523–4527 (2009)
56. Xiao-lin, Z.; Hai-wei, W.: Crude oil production predictive model based on support vector machine and parameters optimization algorithm. In: Zhang, W. (ed.) Advances in Intelligent and Soft Computing, vol. 162, 637–643 Springer, Berlin (2012). doi: 10.1007/978-3-642-29455-6_88
57. Makridakis, S.; Hibon, M.: The M3-Competition: results, conclusions and implications. Int. J. Forecast. **16**(4), 451–476 (2000)