
Fractured Reservoirs History Matching based on Proxy Model and Intelligent Optimization Algorithms

Sayyed Hadi Riazi¹, Ghasem Zargar^{1*}, Mehdi Baharimoghadam¹
and Ebrahim Sharifi Darani²

1. Department of Petroleum Engineering, Petroleum University of Technology (PUT), Ahwaz, Iran.

2. Department of Reservoir Evaluation, National Iranian South Oil Company (NISOC), Ahwaz, Iran.

(Received 23 December 2015, Accepted 21 February 2016)

Abstract

In this paper, a new robust approach based on Least Square Support Vector Machine (LSSVM) as a proxy model is used for an automatic fractured reservoir history matching. The proxy model is made to model the history match objective function (mismatch values) based on the history data of the field. This model is then used to minimize the objective function through Particle Swarm Optimization (PSO) and Imperialist Competitive Algorithm (ICA). This procedure leads to matching of history of the field in which a set of reservoir parameters is used. The final sets of parameters are then applied for the full simulation model to validate the technique. The obtained results showed that due to high speed and need for little data sets, LSSVM is the best tool to build a proxy model. Also the comparison of PSO and ICA showed that PSO is less time-consuming and more effective.

Keywords

Automatic fractured reservoir history matching;
ICA;
LSSVM;
Proxy model;
PSO.

1. Introduction

Numerical reservoir simulation could provide the ability to understand the real reservoir behavior. To propel the simulated data to the real data, it is necessary to carry out the history matching operations and tune the reservoir parameters [1].

The main stages of the history matching process involve selecting parameters, defining the mathematical model, defining the objective function, analyzing sensitivity and stop conditions. The

major problems in history matching are: 1) generally, history matching is done manually and due to the enormous number of data used, a desired result is not achieved.; 2) it would be difficult to adjust the parameters to obtain the match due to the large number of reservoir parameters; 3) optimization algorithms used in the history matching process, optimize the problem locally; thus, when there are several minimums an acceptable solution would not be provided; and 4) typical history matching procedure works for one simulation model and does not have the ability to work with several number of models. To solve the problems mentioned above, different techniques of automatic history matching were offered. In the proper procedure,

* Corresponding Author.
Email: zargar@put.ac.ir

one of the most important activities to achieve an acceptable result is to improve the optimization algorithms to achieve global minimum [2].

Two of the most famous global optimizers in the literature are employed: the PSO and ICA. These two algorithms need large number of objective function evaluation for optimization but each function evaluation needs a full simulation run which is time consuming. In order to reduce the function evaluation time, proxy models are used. Proxy models are alternatives to the reservoir simulation model. A good proxy model should have the following features [3, 4]: 1) an acceptable imitation of nonlinear behavior of the actual model, 2) a simple application, 3) straight forward construction. A number of proxy models are used for reservoir simulation by different authors and each proxy model has been used for a particular reservoir and application [5]. Proxy models can simplify the process of finding the optimal values of reservoir parameters to reach the history matching by speeding up the calculations. This is very important for fractured reservoir because of its complex behavior.

History matching of fractured reservoirs poses more challenges compared to conventional reservoirs in two main areas: the number and type of history matching parameters, and the increased computational cost. For example, in the single porosity model, relative permeability and k_v/k_h (k_v : vertical permeability; k_h : horizontal permeability) are used as matching parameters in the match of water cut and gas production, whereas the matching parameters in the dual porosity model are fracture porosity, shape factor and k_{fv}/k_{fh} (k_{fv} : fracture vertical permeability; k_{fh} : fracture horizontal permeability). The dual porosity models have longer execution time than the single porosity models because of the large number of parameters. Also, the inter-porosity flow between the matrix and fracture poses additional challenge arising from the matrix-fracture interactions because it requires extensive computation. The doubling of the number of computational cells and significant non-linearity increase the computations required to evaluate the dual porosity model compared with an equivalent single porosity model. A partial representation of the fracture networks or describing them in a simplistic way in reservoir models due to scarcity of fracture data or lack of necessary numerical tools is one of the challenges of the fractured reservoir history matching.

Considering the importance of proxy application in the history matching, many studies have been carried out in this area. Cullik et al. conduct-

ed the history matching using a nonlinear proxy and global optimization [6]. They used the neural networks as a proxy model and showed that the required number of simulation runs to obtain a good history match can be reduced by the neural network. Yu et al. used the genetic programming as a proxy model for history matching [7]. Zhang et al. provided an automatic history matching based on improved genetic algorithm [1]. They showed that the rate of convergence of the automatic history matching can be significantly increased by the improved genetic algorithm. Rammay et al. used the Adaptive Neuro-Fuzzy System (ANFIS) as a proxy to reservoir simulator [8]. They combined ANFIS and Differential Evolution (DE) algorithm to reduce the number of simulation runs and the expensive simulation time. Maschio et al. replaced the flow simulator by proxy models created by artificial neural network (ANN) to make possible the application of the sampling method in the history matching [9]. They used Markov Chain Monte Carlo (MCMC) sampling and combined it with ANN. Goodwin appraised the limitations of random walk MCMC [10]. They showed that a combination of MCMC and proxy models provide a more reliable probabilistic uncertainty quantification and a suitable ensemble of deterministic reservoir models. He et al. proposed the proxy-for-data approach [11]. In their work, the aggregated mismatch was calculated by the data values predicted by proxies. They also reduced the number of proxies needed by using of reduced order modeling.

In this paper, use of Least Square Support Vector Machine (LSSVM) as a nonlinear proxy model is proposed and a history match workflow with strong and nonlinear LSSVM proxy model to improve the history matching process is presented. One of the Iranian fractured reservoir simulation model and its history data is used as the case study.

2. LSSVM for Function Approximation

Considering the high performance of the support vector machine (SVM) in function approximation, the application of this algorithm has caused a significant growth in the field of oil reservoir modeling. SVM as a learning organization takes the nonlinear problems into high dimensional feature space and solves the problem through the kernel functions. Accordingly, SVM forecasts the functions so that the desired functions are developed on the subset of support vectors [12, 13, 14, 15 and 16]. A version of SVM for regression is called support vector regression (SVR).

The purpose of SVR is to find a function $f(x)$ that has at most ϵ deviation from the actually obtained targets $y^{(i)}$ for all the training data, and is as flat as possible simultaneously. In the case where $f(x)$ is a linear function of the form $f(x) = \omega^T x + b$, the resulting primal optimization problem is shown in the following form [17]:

$$\begin{aligned} &\text{Minimize } \frac{1}{2} \omega^T \omega + C \sum_{i=1}^m (\epsilon_i + \epsilon_i^*) \\ &\text{Subject to} \\ &\begin{cases} y^{(i)} - \omega^T x^{(i)} - b \leq \epsilon + \epsilon_i \\ \omega^T x^{(i)} - y^{(i)} + b \leq \epsilon + \epsilon_i^* \\ \epsilon, \epsilon_i, \epsilon_i^* \geq 0 \end{cases} \end{aligned} \quad (1)$$

- $\omega^T \omega$ controls the trade-off between the complexity and the approximation accuracy of the model
- $\epsilon, \epsilon_i, \epsilon_i^*$ are slack variables that measure the error of the up and down sides, respectively
- C controls the trade-off between the error and margin

This optimization problem can be transformed into the dual problem, which is easier to solve, and its solution is given by

$$\begin{aligned} f(x) &= \sum_{i=1}^{n_{SV}} (\alpha_i - \alpha_i^*) k(x_i, x) \\ &\text{subject to } 0 \leq \alpha_i^*, \alpha_i \leq C \end{aligned} \quad (2)$$

Where α_i^* and α_i are called the Lagrangian multipliers in Eq. (2), which satisfy the equalities $\alpha_i^* \alpha_i = 0, \alpha_i > 0$ and $\alpha_i^* \geq 0$ and n_{SV} is the number of Support Vectors (SVs) and the kernel function

$$K(x, x_i) = \sum_{j=1}^m g_j(x) g_j(x_i) \quad (3)$$

In order to reduce complexity and increase computing speed, modified SVM as LSSVM is offered [18]. LSSVM as an approximation function is used to estimate a function $y(x)$ from a given training set of N samples $\{x_i, y_i\}_{i=1}^N$ in which $x_i \in R^N$ (N dimensional vector space) as input data and $y_i \in R$ (one dimensional vector space) as corresponding output data [19]. LSSVM suggests the following equation to estimate $y(x)$:

$$y(x) = w^T \varphi(x) + b \quad (4)$$

Where the nonlinear function $\varphi(x)$ takes the input data into a high dimensional feature space to reduce the complexity and increase the speed of problem solving; b is the bias value and w is a

weight vector having the similar dimension with the defined space dimension. To approximate LSSVM, $y(x)$ should optimize the following problem [19]:

$$\frac{1}{2} w^T w + \gamma * \frac{1}{2} \sum_{i=1}^N e_i^2 \rightarrow \text{Must be minimized}$$

Where γ^* = regularization parameter and e_i = error variable. After minimization the above problem, $y(x)$ can be obtained as follows [7]:

$$y(x) = \sum_{i=1}^N \alpha_i K(x, x_i) + b \quad (5)$$

Where $k(x, x_i)$ is the kernel function and α_i is Lagrange multiplier called the "support value" which α_i and b are obtained from optimization problem described above. There are different forms of kernel functions such as linear, polynomial and radial basis function (RBF) [20]. Table 1 shows common kernel function and corresponding mathematical expression.

Table 1. Common kernel function and corresponding mathematical expression.

Kernel function	Mathematical expression
Linear function	$K(x, x) = \langle x, x \rangle$
Radial basis function	$K(x, x) = \exp(-\ x_i - x\ ^2 / 2\sigma^2)$
Polynomial function	$K(x, x) = (x^T x + 1)^d, d = 1, 2, 3, \dots$

Among the forms available for the construction of kernel function, RBF has the maximum efficiency and can improve the performance of the LSSVM [21]. Table 2 shows the performance of LSSVM with different kernels.

Table 2. The performance of LSSVM with different kernels.

Model	RMSE	ARE	R ²
LSSVM (linear)	48.274	0.9821	0.7856
LSSVM (radial)	25.639	0.7302	0.9491
LSSVM (polynomial)	37.821	0.8329	0.8635

The RBF is defined as follows [7]:

$$K(x_1, x_2) = \exp(-\|x_1 - x_2\|^2 / 2\sigma^2) \quad (6)$$

Where σ^2 is the width of RBF. The values of γ^* and σ^2 are gained during the training of LSSVM [19].

3. Particle Swarm Optimization (PSO)

The Particle Swarm Optimization (PSO), which was first presented by Kennedy and Eberhart [22], could be a strong competitor to other evolutionary algorithms that solve global optimization problems [23]. PSO as a stochastic optimization technique is the model of the motion of a group of birds and fishes [23]. Works done by other authors showed that PSO acts quickly and more efficiently compared to other optimization techniques like Genetic Algorithms (GA) and DE [24]. PSO has the same effectiveness (finding the true global optimal solution) as the GA but with significantly better computational efficiency (less function evaluations) by executing statistical analysis and formal hypothesis testing [25]. Another reason that PSO is interesting is that it has a small number of parameters to tune, its formula is simple and easy to implement in computer [26].

In PSO, a set of randomly generated solutions, called particles, fly through the problem hyperspace. According to the following equations, the position of each particle is changed according to its own experience (pbest) and that of its neighbors (gbest) [22]:

$$v_{i+1} = w v_i + c_1 r_1 (pbest_i - x_i) + c_2 r_2 (gbest_i - x_i) \quad (7)$$

$$x_{i+1} = x_i + \Delta t v_{i+1} \quad (8)$$

- v is the particles speed
- r_1, r_2 are two random numbers generated in the interval $[0, 1]$
- c_1 (Self Confidence), c_2 (Swarm Confidence) are intensities of attraction towards pbest and gbest respectively
- Δt is a time parameter which represents the advance step of the particles
- w is a factor of inertia which controls the velocity effect. In this work, the value of 1 was used for w

At iteration $i+1$, the velocity of a particle is updated and two forces that attract the particle to pbest and gbest. The position of each particle is updated using its velocity vector in the end of the iteration.

PSO is expressed in the following simple commands [27]:

1. Initialize the swarm particles with random placement of particles in parameter space such that each particle has an acceptable random velocity;
2. Calculate and evaluate the cost function for each particle;

3. Compare the value of each particle with its personal best position (pbest). If the current value of the desired particle is better than the pbest value, the position of the particle and pbest are replaced with each other;
4. Update the position and the cost function of global best (gbest);
5. Update the position and velocity of every particle after steps 1 to 4;
6. Continue steps 1 to 5 until stopping conditions are reached such as the maximum number of iterations and/or the appropriate cost function.

Fig. 1 shows how to update the position of the particle by PSO.

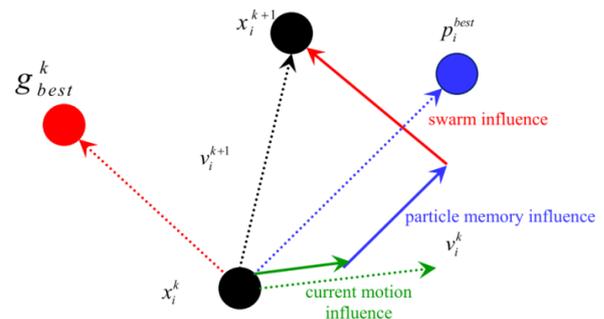


Figure 1. Concept of modification of a searching point by PSO [27].

4. Imperialist Competitive Algorithm (ICA)

Imperialism is a policy that an imperialist applies it in order to extend its power beyond its boundaries. Countries colonized by the colonizer are controlled directly or indirectly such as controlled goods or raw materials [28]. An algorithm based on this policy named: Imperialist Competitive Algorithm (ICA) was presented by Atashpaz-Gargari and Lucas [29]. ICA as a sociopolitical global search technique was presented for different optimization problems recently.

There are many studies about the application of the optimization techniques such as PSO, GA and DE in history matching while ICA is rarely used. ICA is a new optimization technique. The aim of this work is to evaluate the performance of ICA in history matching. Also, the works done in other fields showed that ICA has exhibited excellent abilities such as accuracy, faster convergence

and better global optimum attainment compared to traditional GA [30].

Like PSO, ICA starts with an initial random population. Each individual of the population called countries are divided into two types: colonies and imperialists that all together create some empires. Imperialistic competition among these empires is the core of ICA. In this step, based on the cost of the imperialists, each country is allocated to an empire. Firstly, the total cost of every empire is calculated and normalized according to following equations [29]:

$$T.C._n = \text{Cost}(\text{imperialist}_n) + \xi \cdot \text{mean}\{\text{Cost}(\text{colonies of empire}_n)\} \quad (9)$$

$$N.T.C._n = T.C._n - \max\{T.C._i\} \quad (10)$$

Where

- $T.C._n$ is the total cost
- $N.T.C._n$ is the normalized total cost of the nth empire
- ξ is a little positive number which is considered to be less than 1. This value determines the role of the colonies in determining the total power of an empire.

After creating initial empires, the colony moves toward the imperialist by x units, and the moving model is shown in Fig. 2 [29].

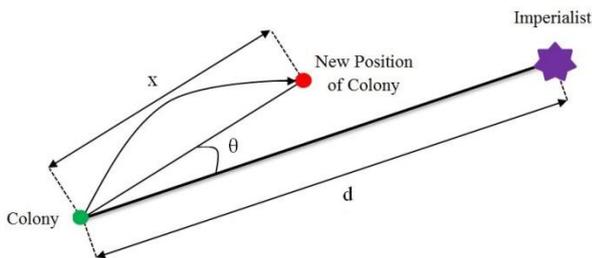


Figure 2. Movement of colonies toward their relevant imperialist in a randomly deviated direction.

In this movement, θ and x are random numbers with uniform distribution as demonstrated in equation (11) and d is the distance between the colony and the imperialist.

$$x \sim U(0, \beta \times d), \theta \sim U(-\gamma, \gamma) \quad (11)$$

Where β and γ are arbitrary numbers that modify the random searching domain of colonies around the imperialist.

The weakest colony of the weakest empire is

selected. This colony is colonized by other empires through competition. The possession probability of each empire is given by the equation (12) and form the vector P as the equation (13) [29]:

$$p_n = \left| \frac{N.T.C._n}{\sum_{i=1}^{N_{imp}} N.T.C._i} \right| \quad (12)$$

$$P = [p_{p_1}, p_{p_2}, p_{p_3}, \dots, p_{p_{N_{imp}}}] \quad (13)$$

A vector R with the same size as P whose elements are uniformly distributed random numbers is produced as the equation (14):

$$R = [r_1, r_2, r_3, \dots, r_{N_{imp}}]$$

where

$$r_1, r_2, r_3, \dots, r_{N_{imp}} \sim U(0,1) \quad (14)$$

Then vector D is created by simply subtracting R from P , as the equation (15):

$$D = P - R = [D_1, D_2, D_3, \dots, D_{N_{imp}}] = [p_{p_1} - r_1, p_{p_2} - r_2, p_{p_3} - r_3, \dots, p_{p_{N_{imp}}} - r_{N_{imp}}] \quad (15)$$

Finally, the empire whose related index in D is maximized will obtain the mentioned colony. The competition continues until the stop condition is met. The stop condition can be one of the following:

- A preset maximum number of iterations is reached
- All the colonies is under the control of unique empire

The weak empires gradually lose their colonies and ultimately they will collapse. The imperialistic competition and the collapse mechanism will cause all the colonies to converge to a state in which there exists just one empire. This remaining empire stands for the solution.

The following workflow is offered to apply the ICA at the computer system [23]:

1. Initialize to generate the initial empires and colonies;
2. Move the colonized countries towards self-empire (assimilation);
3. Change the position of some countries under colonial randomly (revolution);

4. Replace the position of a country with its empire if the cost function of the country in a colonial is greater than its empire;
5. Calculate and compare the total cost of all empires with each other. (total cost for an empire depends on the strength of the emperor and its controlled countries);
6. Transfer the colonies of the weakened empire to the empire with greater power (imperialistic competition);
7. Remove the weakest empire;
8. Continue steps 2 to 7 so that it reaches the stop condition.

5. A Case Study

A sector of one of the Iranian fractured reservoir in the Ahwaz oilfield has been used for this study. This reservoir is part of the Asmari reservoir and is mainly composed of carbonate entities (limestone and dolomite). Oil, water and gas exist as three phase in this reservoir. The schematic diagram of the reservoir is shown in Fig. 3. Other properties of the reservoir are summarized in Table 3. The reservoir is double porosity.

This case study is a highly fractured reservoir such that pressure of all wells is equal in the whole reservoir. Also, the performances of the wells such as well productivity index are very similar. There-

fore, the fractures properties can be assumed homogeneous with good approximation.

In fact, the problem of this type of fractured reservoirs is dependence of the model output such as water and gas production rate and coning phenomenon upon the fractures properties (fracture porosity, shape factor and etc). Due to lack of fracture data such as Formation Micro Scanner (FMS) and Formation Micro Imaging (FMI), fracture modeling has not been established.

Despite the homogeneity of the fracture system, the matrix systems are quite heterogeneous. To show this heterogeneity, matrix porosity distribution is shown in Fig. 4. Also, Fig. 5 shows the histogram of matrix porosity distribution in cells.

There is little or no difference between this fractured reservoir and conventional reservoir in history matching process because this model is a highly fractured reservoir and fracture properties are largely homogeneous. But the main difference between this dual porosity model and conventional model is longer execution time in history matching process. Therefore, the main objective of this paper is to reduce the necessary runtime for matching of history of the field. Another aim of this work is to evaluate the LSSVM performance as a proxy model in this type of fractured reservoirs.

Fig. 6 shows the location of the drilled wells. In this reservoir, 18 wells have been drilled. All the drilled wells have production except well #7. In or-

Table 3. Reservoir properties in full simulation model.

Property	Value	Property	Value
X Dimension	53	Ave Matrix Perm (x & y) (md)	0.18589
Y Dimension	15	Ave Matrix Perm (z) (md)	0.34677
Z Dimension	146	Ave Matrix Porosity	0.05503
Ave Fracture Dx (ft)	1174	Ave Fracture Pressure (psi)	3223.6
Ave Fracture Dy (ft)	1543.6	Ave Matrix Pressure (psi)	3219.4
Ave Fracture Dz (ft)	22.169	Ave Fracture Oil Saturation	0.58231
Ave Matrix Dx (ft)	1207.9	Ave Fracture Water Saturation	0.33771
Ave Matrix Dy (ft)	1559.9	Ave Matrix Oil Saturation	0.36983
Ave Matrix Dz (ft)	22.246	Ave Matrix Water Saturation	0.61439
Active Phases	Live Oil , Water and Gas	Water Density (lb/ft ³)	62.428
Number of Active Cells (Fracture)	58035	Gas Density (lb/ft ³)	0.0608
Number of Active Cells (Matrix)	58035	Oil Density (lb/ft ³)	51.78
Ave Fracture Perm (x & y) (md)	707.09	OWC (ft)	6070
Ave Fracture Perm (z) (md)	445.41	GOC (ft)	2750
Ave Fracture Porosity	0.00786	IOIP (STB)	3553366676

der to maintain the reservoir pressure, well #7 has been drilled for gas injection. All wells produce oil

and gas. In addition to oil and gas production, well #9 also produces water.

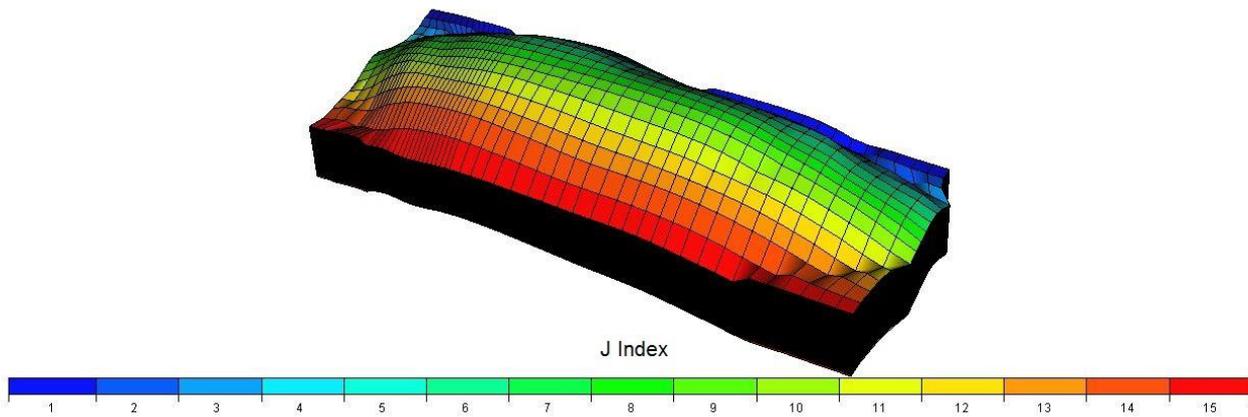


Figure 3. Three-dimensional reservoir model.

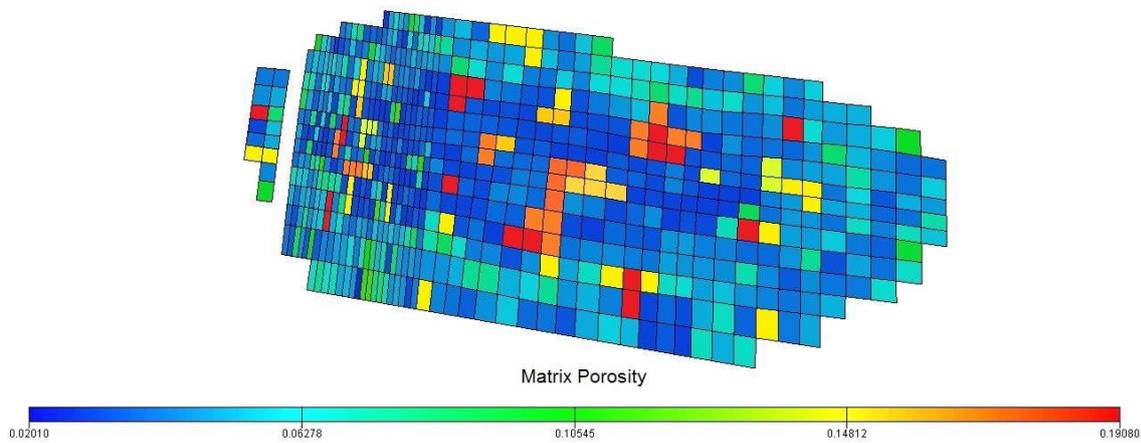


Figure 4. Matrix porosity distribution.

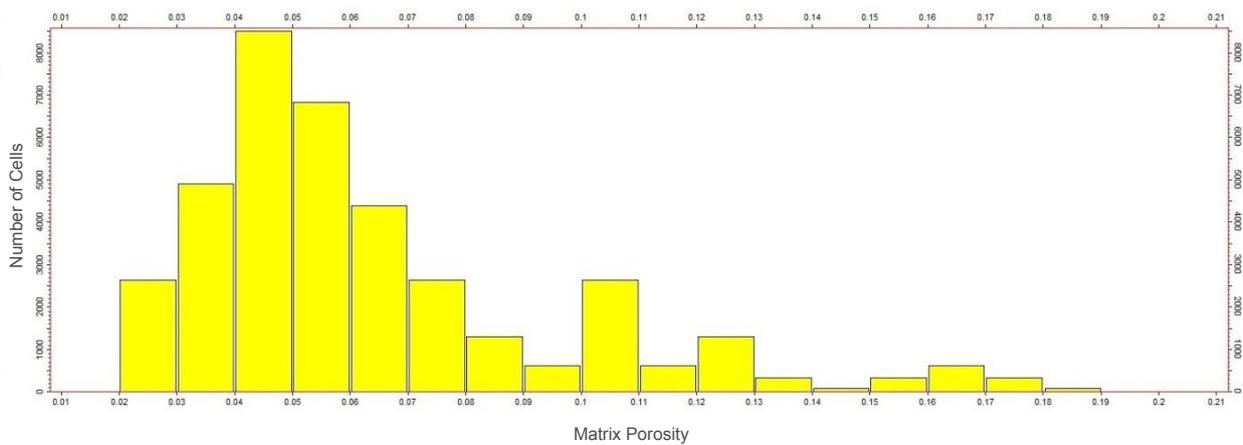


Figure 5. Histogram of matrix porosity distribution in cells.

6. Methodology

6.1. Data Preparation

Due to high level of Iranian fractured reservoirs heterogeneities, these reservoirs are described by wide range of uncertainties. These uncertainties mostly originate from unknown fracture network, aquifer volume and etc. Uncertainties in Iranian fractured reservoirs have many sources and may be created anywhere within the reservoir modeling workflow [31]. All these uncertainties lead to a complex reservoir model. Therefore evaluation of this model produces unacceptable results.

In this work, parameters that have wide range of uncertainty are selected as matching parameters so that fractured reservoirs evaluation will be improved by finding the appropriate value of these parameters in history matching process. Table 4 shows the 13 parameters as input data to build a proxy model for history matching. These parameters will have wide range of uncertainty and the greatest impact on the behavior of fractured reservoirs.

The most important step to build a high-performance proxy model is sampling the input data (design of experiments on data) [5]. Several methods for data sampling and design of experiments are available like full factorial designs in two levels, Plackett-Burman designs and Latin Hypercube Sampling (LHS). Among sampling methods, LHS has more computational efficiency. To design n samples of data, LHS acquires cumulative-proba-

bility distribution for each input parameter. Then LHS offers n uniform distributed points on cumulative-probability distribution for each parameter. Obtained values for the input parameters related to these n points, are combined together to create the sampling randomly. Using the LHS, 480 data sets as input files to run in full simulation model were generated from the 13 parameters. Table 4 shows a summary of the 480 data sets features. The base value column in Table 4 shows the values of these 13 parameters before history matching process. These 480 data sets as input parameters are used to build the proxy model. We are going to match the oil, water and gas rates and well static pressures for all wells in this paper. After running the full simulation model, the oil, water and gas rates and static pressures as the full simulation model outputs are used to create the objective function. The objective function (mismatch function) is defined by the following equation.

$$\text{Objective Function} = \sum_i \frac{(y_i^{\text{calc}} - y_i^{\text{obs}})^2}{\sigma_i^2} \quad (16)$$

$y_i^{\text{calc}}(x)$ as the simulation value is achieved from simulator model. $y_i^{\text{obs}}(x)$ as the observed value is obtained from reservoir production history. σ_i represents the standard deviation of the observed values. In this work, the objective function is defined for the oil, gas and water rates and static pressures separately. The final objective function (final output) is the weighted sum of the objective functions that is defined for the oil, gas and water rates and

Table 4. Input parameters.

Parameters	Data No.	Base Value	Min	Max	Mean	Standard Deviation
Aquifer Porosity	480	0.1	0.03	0.2	0.054222972	0.009505285
Aquifer Permeability (md)	480	20	1	2000	1025.057408	575.5163989
Aquifer Raduis (ft)	480	5000	3000	25000	19041.81708	3214.940298
Aquifer Height (ft)	480	2000	700	3000	1858.985328	636.4149481
Aquifer Compressibility (psi ⁻¹)	480	5.0×10 ⁻⁵	9.89×10 ⁻⁵	2.96×10 ⁻⁴	0.000199214	5.72527×10 ⁻⁵
Matrix Compressibility (psi ⁻¹)	480	3.55×10 ⁻⁶	2×10 ⁻⁶	15×10 ⁻⁶	8.71×10 ⁻⁶	3.72571×10 ⁻⁶
Fracture Compressibility (psi ⁻¹)	480	160×10 ⁻⁶	150×10 ⁻⁶	200×10 ⁻⁶	0.000175379	1.41999×10 ⁻⁵
Shape factor (ft ²)	480	0.002	0.0001	0.1	0.050673449	0.028564032
Matrix block height (ft)	480	20	10	100	56.34262044	26.09562503
Pore volume multiplier	480	0.5	0.2	1	0.634034597	0.213547505
Fracture Permeability (z) (md)	480	100	90	500	350.1113464	87.10665108
Fracture Permeability (x&y)(md)	480	1000	50	2000	1043.143136	556.5866535
Fracture Porosity	480	0.002	0.0001	0.01	0.005592596	0.002747863

static pressures. Fig. 7 shows how to create the final objective function needed to build the proxy model.

Required weights of the weighted sum have been used for assimilation the objective functions impact on the final objective function. The weights were adjusted based on the magnitude of the oil, water and gas rates and static pressure. Due to the higher magnitude of the oil rate, the smallest weight for the oil rate was selected. Table 5 shows weights value used in construction of the final objective function.

Table 5. Weights value used in construction of the final objective function.

Weights Value			
w_1	w_2	w_3	w_4
5	10	10	10

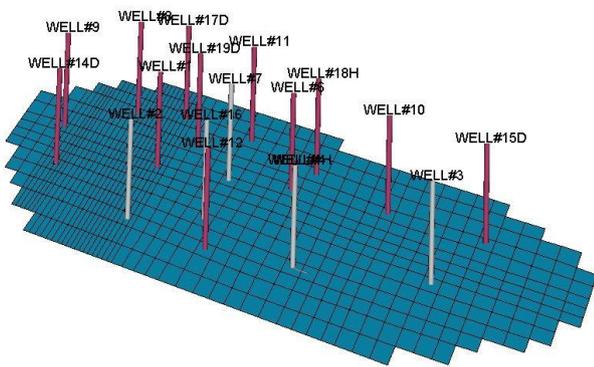


Figure 6. Location of the drilled wells.

6.2. Proxy Construction using LSSVM

The proxy model construction based on LSSVM is summarized as following steps [20]:

1. The entire dataset are divided into three parts:

training, validation and testing. The training and validation dataset are used to build the LSSVM model and the tested data set are used to verify and evaluate the performance and efficiency of this model.

2. Initialize the parameters σ^2 and γ^* using the training dataset. (regularization parameter (γ^*) and kernel width parameter (σ^2) play an important role in the LSSVM performance. The task of γ^* is creating a suitable LSSVM structure based on the minimum training error and is minimizing the model complexity. The input data in original space is transferred into a high-dimensional feature space by σ^2 . The small value of σ^2 causes the over fitting. Also the large value of σ^2 reduces the LSSVM accuracy [32]).
3. Using the grid search technique with cross-validation method, the optimal values of parameters σ^2 and γ^* were obtained. In this work, ten-fold cross-validation is used. In this process, the training datasets are divided into ten equal parts. The grid training data consists of nine equal parts and remaining part devoted to grid validation data. LSSVM will be trained by the grid training data. σ^2 and γ^* are also optimized via the training process of LSSVM. After training the LSSVM, this model is tested by grid validation data and this operation is repeated ten times. The LSSVM training and testing processes continue until a stopping condition is reached. In this condition, optimal value of σ^2 and γ^* can be achieved with the minimized error.
4. After obtaining the optimum value of σ^2 and γ^* , these parameters are used to construct the LSSVM model.
5. After building the LSSVM model, the testing datasets are applied on the model in order to investigate the model performance.

Table 6 shows the optimum value of σ^2 and γ^* to build the LSSVM model in this paper.

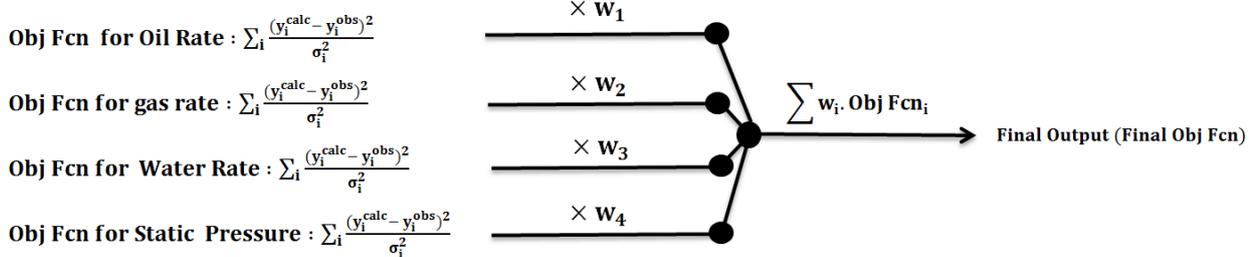


Figure 7. Final objective function construction.

Table 6. The optimal parameter combination (γ, σ^2) to build the LSSVM model.

Tuned Parameters	
γ^*	σ^2
3.1392	1.5982

6.3. Evaluation of Model Performance

Assessment the accuracy of the prediction model is the last and most important step in the modeling process. In this work, quantitative analysis is used to evaluate the model performance. Therefore, correlation coefficient (R^2), absolute relative error (ARE) and root mean square error (RMSE) are used for quantitative analysis. ARE shows closeness of the simulated data with the actual data. ARE is defined as:

$$ARE = \left| \frac{y_i - \hat{y}_i}{y_i} \right| \quad (17)$$

Where, y_i is an actual data and \hat{y}_i is a forecasted value obtained by proxy model. R^2 provides a value that represents the amount of success in reducing the standard deviation by regression analysis. R^2 is defined as:

$$R^2 = 1 - \frac{\sum(y_i - \hat{y}_i)^2}{\sum(y_i - \bar{y})^2} \quad (18)$$

Where \bar{y} is the average of y_i .

RMSE represents the difference between the simulation data and real data. RMSE is defined as:

$$RMSE = \sqrt{\frac{\sum(y_i - \hat{y}_i)^2}{n}} \quad (19)$$

Where, n is the number of observation data.

7. Results

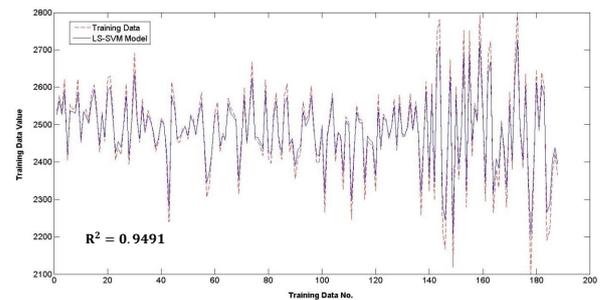
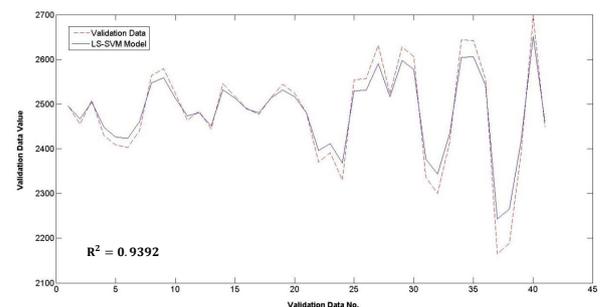
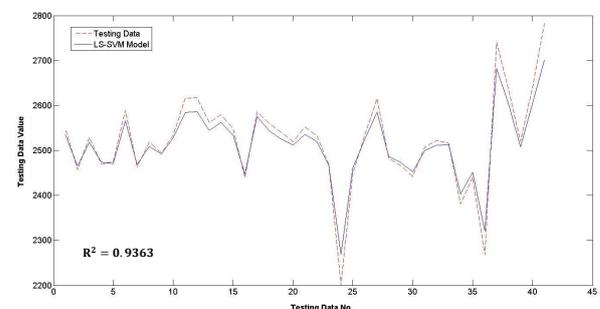
The 480 sets of data were provided as input files to run in full simulation model by LHS. The convergence of reservoir simulation model was obtained in 270 of 480 data sets. These 270 data sets as input parameters are used to build the proxy model. 70%, 15% and 15% of these input dataset are defined as training, validation and testing sets, respectively. To construct the proxy, 230 cases of the numerical simulation model were made and used for the training of the proxy model. 40 cases are selected randomly among the training cases and used to validate the proxy model. 40 cases were also made and used for the testing of the proxy model. The datasets of these cases are

different from the datasets of training cases. Table 7 shows the performance of constructed LSSVM proxy model.

Table 7. LSSVM performance.

Sample	RMSE	ARE	R^2
Training Data	25.639	0.7302	0.9491
Validation Data	27.8243	0.8691	0.9392
Testing Data	30.275	0.9326	0.9363

Figs. 8, 9 and 10 illustrate difference between the real and the simulated LSSVM output, for training, validation and testing sets, respectively.

**Figure 8.** Training Datasets in LSSVM.**Figure 9.** Validation Datasets in LSSVM.**Figure 10.** Testing Datasets in LSSVM.

For history matching, the proxy model output should be minimized by the optimization algorithms. In fact, optimization algorithms choose a set of input parameters in which the objective function is minimum i.e. the simulation model is history matched.

Prior to optimization, sensitivity of final objective function to each input parameter is evaluated. Figs. 11-23 show the sensitivity of final objective function to each input parameter. In these figures, the sensitivity of the matching parameters to final objective function are shown separately because the new range of the matching parameters (min, max) should be determine in which the final objective function is minimal.

The main purpose of history matching is to reach the minimum objective function. Therefore,

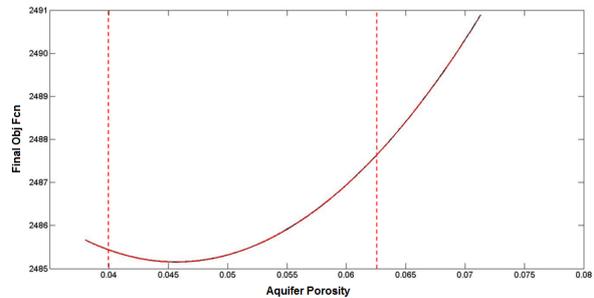


Figure 14. Average Final Output Sensitivity to Aquifer Porosity in LSSVM Model.

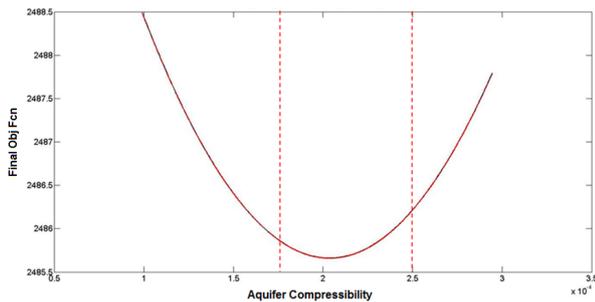


Figure 11. Average Final Output Sensitivity to Aquifer Compressibility in LSSVM Model.

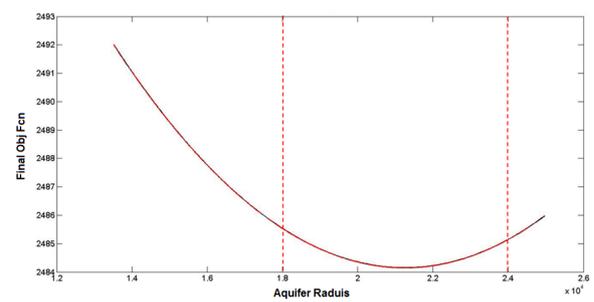


Figure 15. Average Final Output Sensitivity to Aquifer Raduis in LSSVM Model.

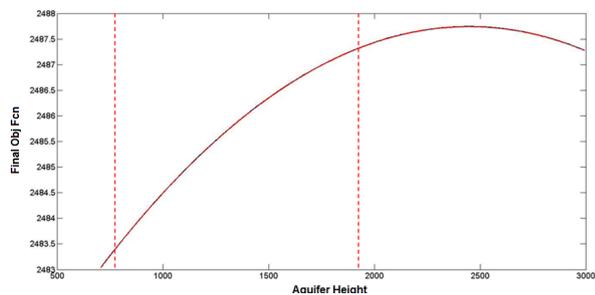


Figure 12. Average Final Output Sensitivity to Aquifer Height in LSSVM Model.

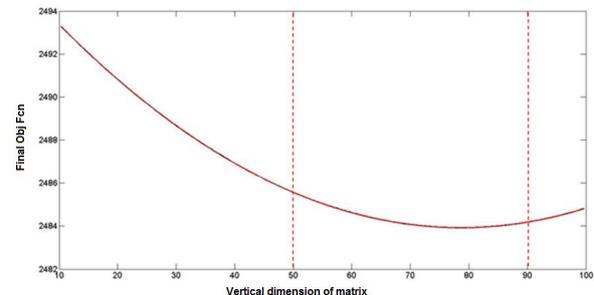


Figure 16. Average Final Output Sensitivity to Vertical dimension of matrix in LSSVM Model.

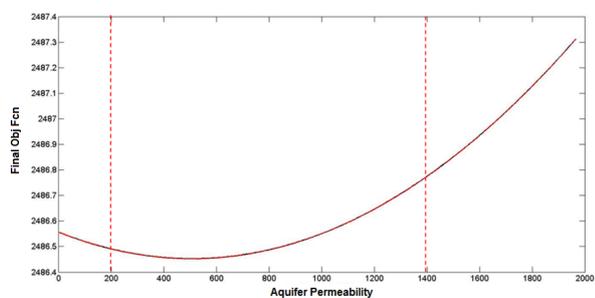


Figure 13. Average Final Output Sensitivity to Aquifer Permeability in LSSVM Model.

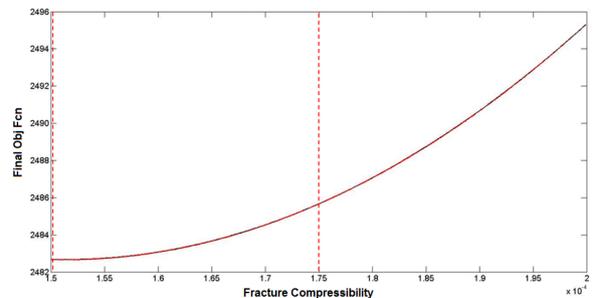


Figure 17. Average Final Output Sensitivity Fracture Compressibility in LSSVM Model.

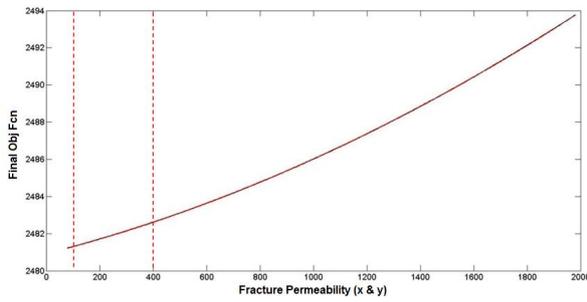


Figure 18. Average Final Output Sensitivity to Fracture Permeability (x & y) in LSSVM Model.

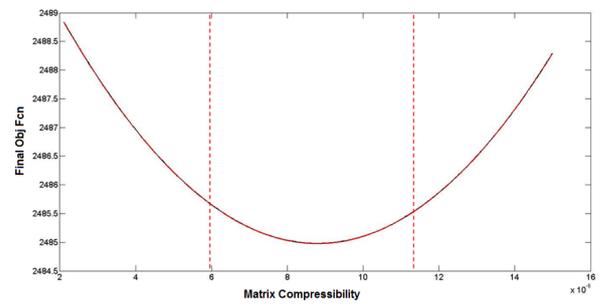


Figure 21. Average Final Output Sensitivity to Matrix Compressibility in LSSVM Model.

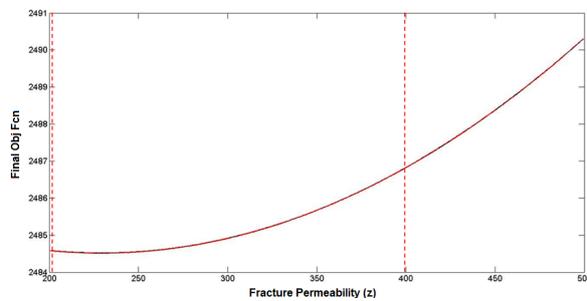


Figure 19. Average Final Output Sensitivity to Fracture Permeability (z) in LSSVM Model.

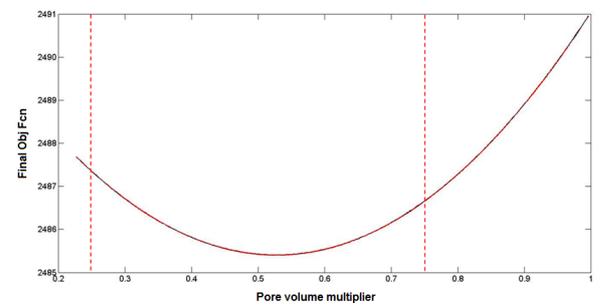


Figure 22. Average Final Output Sensitivity to Pore volume multiplier in LSSVM Model.

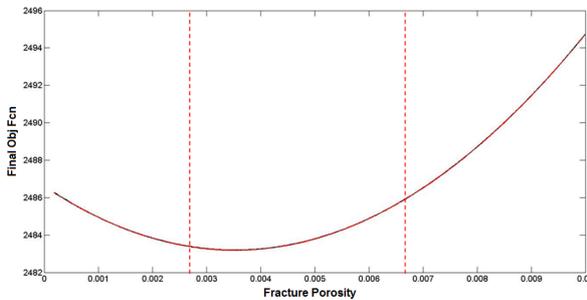


Figure 20. Average Final Output Sensitivity to Fracture Porosity in LSSVM Model.

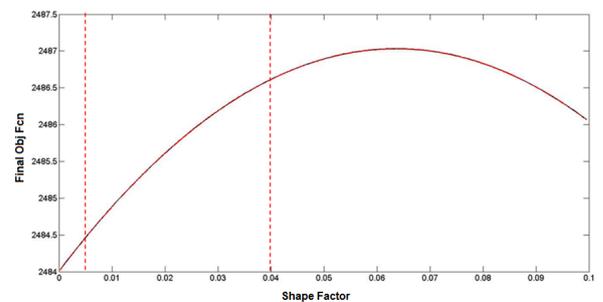


Figure 23. Average Final Output Sensitivity to Shape Factor in LSSVM Model.

determining these ranges is useful to improve the history matching efficiency. Red dash lines in these figures show the modified range of input parameters in which the final objective function is minimal. Fig. 24 shows a comparison between the sensitivity of the final objective function to 13 parameters. Fracture permeability in x and y direction and pore volume multiplier as shown in Fig. 24 have the greatest impact on the LSSVM model output. So, the modified range of these two parameters is more effective in minimizing the LSSVM model output. Table 8 shows the modified ranges of input data.

Next step is to apply the modified ranges in two optimization algorithms (ICA and PSO). These two algorithms have some adjustable parameters as mentioned before. These parameters are used to tune the optimization process.

The main tuning parameters of the PSO model are C_1 , C_2 and the swarm size. The settings of these parameters determine how it optimizes the search-space. According to the type of problem, these tuning parameters should be changed. Therefore, the proper tuning parameters value is needed to be get. The way to solve this problem is by trial and error. In this work, 15 alterations to the PSO were inves-

tigated and their performance (mean square error) determined. Due to the high number of matching parameters (13 parameters), PSO requires a high number of particles (3000, 4000 and 5000) to improve its performance.

Fig. 25 shows impact of the tuning parameters on performance of the PSO.

It is observed that there is a slight improvement of performance of the PSO with increasing swarm size; a larger swarm increases the number of calculations to converge to an error limit. The research presented in this paper found out that setting the two weight factors C_1 and C_2 at 1.5 and 2.5, respectively provides the best performance of the PSO for all runs. Other combinations of values lead to lower performance of the PSO. Table 9 shows performance of the alterations to the PSO.

It is clear that the performance of the ICA is affected by tuning parameters such as number of countries, revolution rate, assimilation coefficient (β) and assimilation angle coefficient (γ). So, good values for the tuning parameters were obtained by trial and error. To investigate the effect of the population size on the performance of the ICA the number of countries was selected from the set {50, 70, 90, 100, 120, 150, 170 and 200} and then executed ICA. Fig. 26 shows the results. Increasing the number of countries reduces MSE.

MSE decreased rapidly for up to 100 countries. So, 100 countries have been used in this work. Based on previous studies on a number of optimi-

zation problems [29, 33 and 34], the best range of variation is 0.1-0.2 for the revolution rate, 0.5-2.5 for β and 0.3-1 (radian) for γ . In this paper, many tests are executed by changing the range of variation for these parameters (the revolution rate between 0.1-0.2, β between 0.5-2.5 and γ from 0.3-1 radian). Table 10 shows the results of some of these tests. The best performance of ICA (the lowest MSE) occurred in the revolution rate=0.1, $\beta=2$ and $\gamma=0.5$.

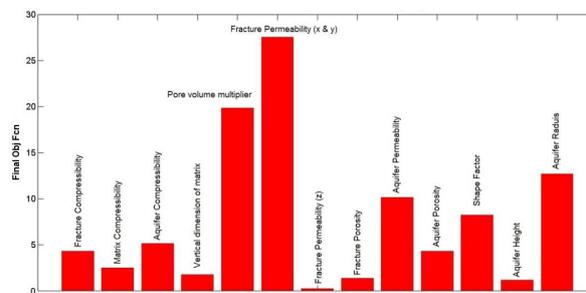


Figure 24. The Sensitivity of the LSSVM Model Output to each Input Argument.

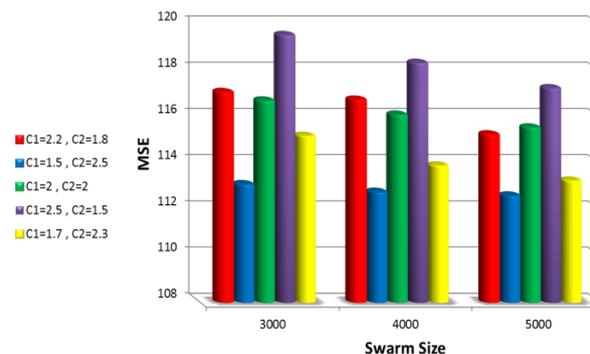


Figure 25. Impact of the tuning parameters on performance of the PSO.

Table 8. Modified range of input parameters in LSSVM model.

Parameters	Modified Range
Aquifer Porosity	0.04 – 0.0625
Aquifer Permeability (md)	200 – 1400
Aquifer Raduis (ft)	1.8×10^4 – 2.4×10^4
Aquifer Height (ft)	750 - 1900
Aquifer Compressibility (psi ⁻¹)	1.75×10^{-4} – 2.5×10^{-4}
Matrix Compressibility (psi ⁻¹)	6×10^{-6} – 11×10^{-6}
Fracture Compressibility (psi ⁻¹)	1.5×10^{-4} – 1.75×10^{-4}
Shape factor (ft ²)	0.005 – 0.04
Matrix block height (ft)	50 – 90
Pore volume multiplier	0.25 – 0.75
Fracture Permeability (z) (md)	200 – 400
Fracture Permeability (x & y) (md)	100 – 400
Fracture Porosity	0.0025 – 0.0065

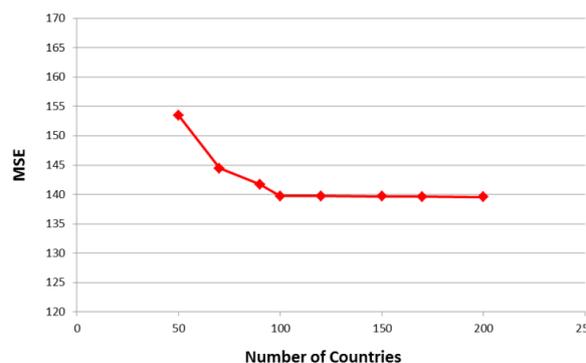


Figure 26. The impact of the population size on the performance of the ICA.

Table 9. Performance of the PSO on LSSVM model.

Run No	Revolution Rate	Assimilation Coefficient (β)	Assimilation Angle Coefficient (γ)	MSE	Run No	Revolution Rate	Assimilation Coefficient (β)	Assimilation Angle Coefficient (γ)	MSE
Run # 1	0.2	0.5	0.3	143.356	Run # 11	0.11	0.7	0.4	141.236
Run # 2	0.2	2	0.7	142.965	Run # 12	0.2	2.3	0.8	143.754
Run # 3	0.12	1	0.5	142.780	Run # 13	0.14	1.7	0.5	139.987
Run # 4	0.1	2.5	0.5	141.324	Run # 14	0.15	1	0.5	140.327
Run # 5	0.1	2	0.5	139.721	Run # 15	0.1	1.5	0.4	143.540
Run # 6	0.2	2.5	0.4	140.013	Run # 16	0.15	2.5	0.3	141.783
Run # 7	0.1	2	1	140.732	Run # 17	0.18	2.5	1	143.481
Run # 8	0.2	1.5	0.4	140.12	Run # 18	0.16	1.5	0.4	141.331
Run # 9	0.15	2.3	0.5	139.823	Run # 19	0.2	1.3	0.5	142.471
Run # 10	0.2	2.5	0.6	140.654	Run # 20	0.1	2	0.7	139.908

Table 10. Performance of the ICA on LSSVM model.

Run No	Swarm Size	Self Confidence (C_1)	Swarm Confidence (C_2)	MSE	Run No	Swarm Size	Self Confidence (C_1)	Swarm Confidence (C_2)	MSE
Run # 1	3000	1.5	2.5	113.132	Run # 8	4000	2	2	116.142
Run # 2	3000	1.7	2.3	115.223	Run # 9	4000	2.2	1.8	116.789
Run # 3	3000	2	2	116.754	Run # 10	4000	2.5	1.5	118.381
Run # 4	3000	2.2	1.8	117.121	Run # 11	5000	1.5	2.5	112.642
Run # 5	3000	2.5	1.5	119.587	Run # 12	5000	1.7	2.3	113.285
Run # 6	4000	1.5	2.5	112.802	Run # 13	5000	2	2	115.585
Run # 7	4000	1.7	2.3	113.937	Run # 14	5000	2.2	1.8	115.285
					Run # 15	5000	2.5	1.5	117.285

Table 11 shows the best performance of optimization algorithms to minimize the LSSVM model output (final objective function).

MSE shown in Table 11 indicates the minimum value of the LSSVM model output obtained by the optimization algorithms. According to the achieved MSE, PSO performance in minimization of the LSSVM model output is better than ICA.

Table 11. The best performance of the optimization algorithms on LSSVM model.

Performance	LSSVM+PSO	LSSVM+ICA
MSE	112.642	139.729

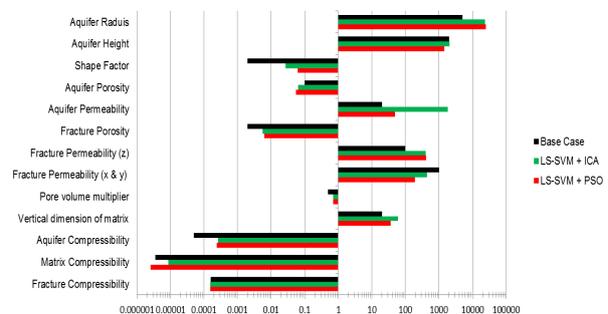
Table 12. Matching parameters achieved from the optimization algorithms.

Parameters	LSSVM+PSO	LSSVM+ICA
Aquifer Porosity	0.044502455	0.0499802
Aquifer Permeability (md)	490.59655812	815.1510
Aquifer Raduis (ft)	24734.9675	23740.975
Aquifer Height (ft)	1438.803772	1886.7283
Aquifer Compressibility (psi ⁻¹)	0.000238368	0.0002408
Matrix Compressibility (psi ⁻¹)	9.582×10 ⁻⁶	8.67×10 ⁻⁶
Fracture Compressibility (psi ⁻¹)	0.000153298	0.000154381
Shape factor (ft ⁻²)	0.03266354	0.026635967
Matrix block height (ft)	78.82779504	60.9488937
Pore volume multiplier	0.709300894	0.699201758
Fracture Permeability (z) (md)	213.5918638	362.0976531
Fracture Permeability (x & y) (md)	191.542375	436.8632893
Fracture Porosity	0.006230512	0.005529285

Table 12 shows the outputs of the optimization algorithms. These outputs are the matching parameters. Fig. 27 shows comparison of matching parameters obtained from the optimization algorithms. Also, table 13 shows the CPU time for using the techniques described in this work.

In automatic history matching, sensitive analysis is often performed on full simulation model. In this work, to get new range of the uncertain parameters (matching parameters) in which the objective function has a minimum value, sensitivity analysis is also performed on the proxy model.

By applying the modified ranges to the optimization methods, optimization of the objective function will be faster and more accurate. So, outputs of the optimization methods (matching pa-

**Figure 27.** Comparison of matching parameters obtained from the optimization algorithms.

rameters) are produced in less time and with high precision.

Now matching parameters obtained by each of the optimization methods are applied in the reservoir simulator. Simulator outputs which include oil, gas and water rates and well static pressure are compared with the real data.

Figs. 28-32 show history matching between simulation data and real data achieved by two different methods. The base case in these figures, states the simulation data before history matching operation.

Table 14 shows the performance of the investigated optimization techniques in history matching. RMSE in this table expresses the difference between the actual data and simulation data. LSSVM+PSO has a better performance than LSSVM+ICA in the history matching process.

Table 13. Comparison of CPU time in different models.

Model	Optimization technique	CPU time (second)
LSSVM (linear)	PSO	179
	ICA	213
LSSVM (radial)	PSO	78
	ICA	92
LSSVM (polynomial)	PSO	122
	ICA	157

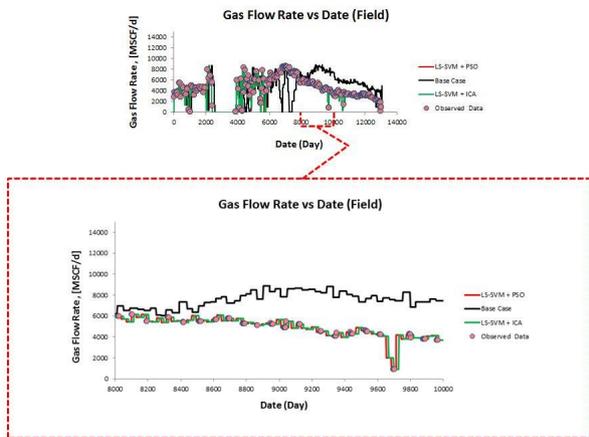


Figure 28. Gas Flow Rate vs Date (Field).

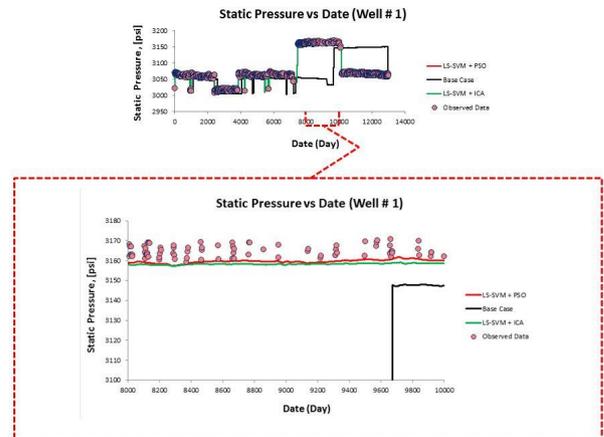


Figure 31. Static Pressure vs Date (Well # 1).

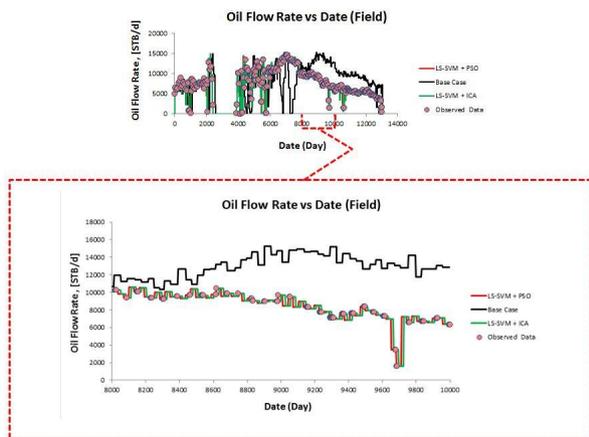


Figure 29. Oil Flow Rate vs Date (Field).

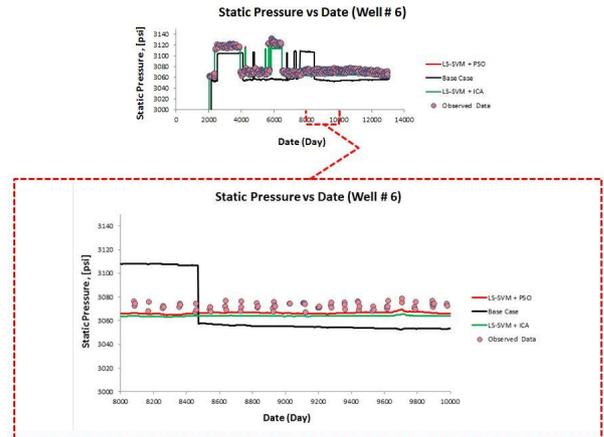


Figure 32. Static Pressure vs Date (Well # 6).

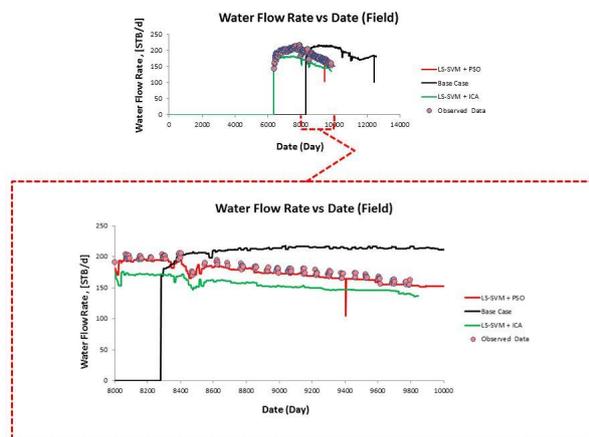


Figure 30. Water Flow Rate vs Date (Field).

Figs. 28 and 29 show the gas rate and oil rate history matching at the field scale. In these figures, both proposed methods have provided acceptable

match. Fig. 30 shows the water rate history matching in the field scale. In this figure, lack of historical water rate data reduces the effectiveness of ICA in the water rate history matching. To investigate the static pressure history matching, two wells (Well #1 and Well #6) were selected from this reservoir randomly. Figs. 31 and 32 show the static pressure history matching in Well #1 and Well #6 respectively. During production, pressure data registered by the well head gauges is often noisy [35]. The noise in the pressure data reduces the efficiency of the investigated optimization methods and proxy model.

8. Conclusions

After evaluating the results of previous section, the concluded items are as follow:

- Due to high speed and need for little data sets, LSSVM is the best tool to build a proxy model

- PSO as an optimization algorithm has a better performance than ICA. High- speed operation and small number of tuning parameters improve the efficiency of PSO.
- The population size has hardly any effect on the performance of the PSO method.
- By performing the sensitivity analysis on the proxy model, fracture permeability in x and y direction and pore volume multiplier are determined as the most important matching parameters in the history matching. These parameters have the greatest impact on the oil, gas, water rates and the static pressure.
- The LSSVM as a proxy model reduces the number of required runs to history matching. It also increases the speed, precision and ease of the history matching process.

Nomenclature

ARE	Absolute relative error	md	Millidarcy
Ave	Average	Obj Fcn	Objective function
R ²	Correlation coefficient	OWC	Oil water contact
DE	Differential evolution	PSO	Particle swarm optimization
GOC	Gas oil contact	Perm	Permeability
GA	Genetic algorithm	pbest	Personal best position
gbest	Global best position	psi	Pounds per square inch
ICA	Imperialist competitive algorithm	RMSE	Root mean square error
IOIP	Initial oil in place	c ₁ , c ₂	Self and Swarm confidence
LHS	Latin hypercube sampling	STB	Stock tank barrel
LSSVM	Least square support vector machine	SVM	Support vector machine
MSE	Mean square error	t _{Day}	Time in Day

Greek symbols

γ	Assimilation angle coefficient
β	Assimilation coefficient
σ^2	Kernel width parameter
α_1^*, α_1	Lagrangian multipliers
γ^*	Regularization parameter
Σ	Shape factor
ϵ_i, ϵ_i^*	Slack variables
σ_i	Standard deviation
σ_i^2	Variance

References

1. Zhang, X., Hou, H., Wang, D., Mu, T., Wu, J. and Lu, X., (2012), "An Automatic History Matching Method of Reservoir Numerical Simulation Based on Improved Genetic Algorithm", International Workshop on Information and Electronics Engineering (IWIEE), Procedia Engineering, Vol. 29, pp. 3924-3928.
2. Bjordalen, N., Kuru, E. and Schiozer, D.J., (2008), "Application of Neural Network and Global Optimization in History Matching", Journal of Canadian Petroleum Technology, PET-SOC-08-11-22-TN, Vol. 47, Issue. 11.
3. Jurecka, F., (2007), "Robust Design Optimization Based on Metamodeling Techniques", Ph.D. thesis, Technische Universität München, München.
4. Lophaven, S.N., Nielsen, H.B. and Sondergaard, J., (2002), "DACE: A Matlab Kriging Toolbox Version 2.0. Technical Report IMM-TR-2002-12", Technical University of Denmark, Lyngby, Denmark, 1st August.
5. Zubarev, D.I., (2009), "Pros and cons of applying proxy models as a substitute for full reservoir simulations", SPE 124815, in Proceedings of the SPE Annual Technical Conference and Exhibition held in New Orleans, Louisiana, USA.
6. Cullick, A.S., Johnson, D. and Shi, G., (2006),

- "Improved and More-Rapid History Matching With a Nonlinear Proxy and Global Optimization", SPE 101933, SPE Annual Technical Conference and Exhibition held in San Antonio, Texas, U.S.A., 24-27 September, <http://dx.doi.org/10.2118/101933-MS>.
7. Yu, T., Wilkinson, D. and Castellini, A., (2008), "Constructing Reservoir Flow Simulator Proxies Using Genetic Programming for History Matching and Production Forecast Uncertainty Analysis", *Journal of Artificial Evolution and Applications*, Vol. 2008, Article ID 263108.
 8. Rammay, M.H. and Abdulraheem, A., (2014), "Automated History Matching Using Combination of Adaptive Neuro Fuzzy System (ANFIS) and Differential Evolution Algorithm", *Society of Petroleum Engineers, SPE Large Scale Computing and Big Data Challenges in Reservoir Simulation Conference and Exhibition, Istanbul, Turkey, SPE-172992-MS*, 15-17 September.
 9. Maschio, C. and Schiozer, D. J., (2014), "Bayesian history matching using artificial neural network and Markov Chain Monte Carlo", *Engineering*, Vol. 123, pp. 62-71.
 10. Goodwin, N., (2015), "Bridging the Gap Between Deterministic and Probabilistic Uncertainty Quantification Using Advanced Proxy Based Methods", *SPE Reservoir Simulation Symposium*, 23-25 February, Houston, Texas, USA, SPE-173301-MS.
 11. He, J., Xie, J., Wen, X. and Chen, W., (2015), "Improved Proxy For History Matching Using Proxy-for-data Approach And Reduced Order Modeling", *SPE Western Regional Meeting*, 27-30 April, Garden Grove, California, USA, SPE-174055-MS.
 12. Boser, B. E., Guyon, I. M. and Vapnik, V., (1992), "A training algorithm for optimal margin". In: Haussler, D. (Ed.), *Proceedings of the Annual Workshop on Computational Learning Theory*. Pittsburgh, PA, ACM, New York, NY, pp. 144-152.
 13. Cortes, C. and Vapnik, V., (1995), "Support vector networks. *Machine Learning*", Vol. 20, pp. 273-297.
 14. Guyon, I., Boser, B. and Vapnik, V., (1993), "Automatic capacity tuning of very large VC-dimension classifiers". In: Hanson, S.J., Cowan, J.D., Giles, C.L. (Eds.), *Advances in Neural Information Processing Systems*. Morgan Kaufmann Publishers, San Mateo, CA, Vol. 5, pp. 147-155.
 15. Scho'lkopf, B., Burges, C. and Vapnik, V., (1995), "Extracting support data for a given task", In: Fayyad, U.M., Uthurusamy, R. (Eds.), *Proceedings of the First International Conference on Knowledge Discovery & Data Mining*, Menlo Park, CA, pp. 252-257.
 16. Vapnik, V., Golowich, S. and Smola, A., (1997), "Support vector method for function approximation, regression estimation, and signal processing", In: Mozer, M.C., Jordan, M.I., Petsche, T. (Eds.), *Advances in Neural Information Processing Systems*. MIT Press, Cambridge, MA, Vol. 9, pp. 281-287.
 17. Smolatand Bernhard Scholkof, A. J., (2003), "A tutorial on support vector regression".
 18. Suykens, J.A.K. and Vandewalle, J., (1999), "Neural Processing Letters", Vol. 9, pp. 293-300.
 19. Samui, p., (2011), "Application of Least Square Support Vector Machine (LSSVM) for Determination of Evaporation Losses in Reservoirs", *Engineering*, Vol. 3, pp. 431-434.
 20. Elmabrouk, S. Kh., (2012), "Application of function approximations to reservoir engineering", Ph.D. thesis, Department of Petroleum Engineering, University of Regina.
 21. Shi, D.F. and Nabil, N.G., (2007), "Tool wear predictive model based on least squares support vector machines", *Mechanical Systems and Signal Processing*, Vol. 21, pp. 1799-1814.
 22. Kennedy, J. and Eberhart, R., (1995), "Particle Swarm Optimization", *Proceedings of the IEEE International Conference on Neural Networks*, Piscataway, NJ, USA, Vol. 4, pp. 1942-1948.
 23. Riazi, S.H., Heydari, H., Ahmadpour, E., Gholami, A. and Parvizi, S., (2014), "Development of novel correlation for prediction of hydrate formation temperature based on intelligent optimization algorithms", *Journal of Natural Gas Science and Engineering*, Vol. 18, pp. 377-384.
 24. Kennedy, J. and Mendes, R., (2002), "Popu-

- lation structure and particle swarm performance”, Proc. of IEEE Conference on Evolutionary Computation, Vol. 2, pp. 1671-1676.
25. Hassan, R., Cohanin, B., Weck, O. and Venter, G., (2005), “A comparison of particle swarm optimization and the genetic algorithm”, Proceedings of the 1st AIAA multidisciplinary design optimization specialist conference, 18-21 April, Austin, Texas, pp. 1-13.
 26. Martinez-Soto, R., Rodriguez, A., Castillo, O. and Aguilar, T., (2012), “Gain Optimization for Inertia Wheel Pendulum Stabilization using Particle Swarm Optimization and Genetic Algorithms” Computing, Information and Control, Vol. 8, pp. 4421-4430.
 27. Mohamed, L., Christie, M. and Demyanov, V., (2010), “Reservoir Model History Matching With Particle Swarms”, Institute of Petroleum Engineering, Heriot-Watt University, Edinburgh, UK, SPE 129152.
 28. Mokhtari, Gh., Ghanizadeh, A.J. and Ebrahimi, E., (2012), “Application of Imperialist Competitive Algorithm to Solve Constrained Economic Dispatch”, Electrical Engineering Department, Amirkabir University of Technology, Tehran, Iran, International Journal on Electrical Engineering and Informatics, Vol. 4, No. 4.
 29. Atashpaz-Gargari, E. and Lucas, C., (2007), “Imperialist Competitive Algorithm: An Algorithm for Optimization Inspired by Imperialistic Competition”, IEEE Congress on Evolutionary Computation, Singapore, pp. 4661- 4667.
 30. Towsyfyan, H., Adnani-Salehi, S. A., Ghayyem, M. and Mosaedi, F., (2013), “The Comparison of Imperialist Competitive Algorithm Applied and Genetic Algorithm for Machining Allocation of Clutch Assembly”, International Journal of Engineering, Vol. 26, No. 12, pp. 1485-1494.
 31. Zabalza-Mezghani, I., Manceau, E., Feraille, M. and Jourdan, A., (2004), “Uncertainty management: From geological scenarios to production scheme optimization”, Journal of Petroleum Science and Engineering, Vol. 44, pp. 11-25.
 32. Espinoza, M., Suykens, J.A.K. and De Moor, B., (2003), “Least squares support vector machines and primal space estimation”, Proceedings of the 42nd IEEE Conference on Decision and Control Maui Hawaii, USA, Vol. 4, pp. 3451-3456.
 33. Bijami, E., Abshari, R., Askari, J., Hoseinnia, S. and Farsangi, M. M., (2011), “Optimal Design of Damping Controllers for Multi-machine Power Systems Using Metaheuristic Techniques”, International Review of Electrical Engineering (IREE), Vol. 6, No. 4, pp. 1883-1894.
 34. Rajabioun, R., Hashemzadeh, F., Atashpaz-Gargari, E., Mesgari, B. and Rajaiee Salmasi, F., (2008), “Identification of a MIMO Evaporator and its Decentralized PID Controller Tuning Using Colonial Competitive Algorithm”, Proceedings of the 17th World Congress, The International Federation of Automatic Control, pp. 9952-9957.
 35. Nardone, P., (2009), “Well Testing Project Management: Onshore and Offshore Operations”, book, Gulf professional press, pp. 61.

