Periodicals of Engineering and Natural Sciences ISSN 2303-4521
Vol. 8, No. 1, March 2020, pp.491-507

Comparing traditional estimators and the estimators of (PSO) algorithm for some growth models of gross domestic product in Iraq

Ibrahim M. Al Marsoomi [1], Emad H. Aboudi [2]
[1] Department of Statistics, University of Baghdad
[2] Department of Statistics, University of Baghdad

ABSTRACT

Growth models are considered to be one of the most important statistical means that is widely used in the study of the behaviors of different phenomena throughout time, and the estimation of the parameters of these models is considered to be the key element that plays a major role in the inference about these models. In this paper, this problem will be discussed briefly. The aim of the paper is to compare the estimators of some traditional methods and the estimators of the Particle Swarm Optimization (PSO) algorithm for estimating the parameters of some growth models as well as building the best growth model for the Gross Domestic Product (GDP) in Iraq. The growth models that are used in this paper will include three linear models which are Polynomials of order (1, 3, and 5) as well as three nonlinear models which are (The Logistic Model, The Gompertz Model, and The Richards Model). It was concluded that the (PSO) algorithm was better than the traditional methods in estimating the parameters of the growth models; also the fifth degree polynomial was the best model to describe the (GDP) data in Iraq.

Keywords: Growth, Growth Models, Logistic Model, Gompertz Model, Richards Model, Particle Swarm Optimization algorithm

Corresponding Author:

Ibrahim M. Al Marsoomi
Department of Statistics, University of Baghdad
Baghdad, Iraq
E-mail: Ibrahim.modhir92@gmail.com

1. Introduction

Growth Models are one of the most important statistical means that are used to study the behavior of a certain phenomenon throughout time. These models are widely used in the study of different economic, social, and health phenomena in the context of the growth of that phenomenon, as the name suggests, or to be more precise the study of the change that occurs to the phenomenon throughout a certain period of time [1]. Growth or the change that occurs to a certain phenomenon throughout time can be represented graphically by a curve known as The Growth Curve. This growth curve can have many forms. Some are linear which are usually used to represent the growth throughout the first year of life of living creatures. Others are nonlinear like the exponential curves and the sigmoid curves (S-shaped curves) which are generally seen in the studies of population growth and animal science, plant science, forestry and biology. Growth curve analysis is considered to be one of the most important statistical tools with wide applications in different fields such as biology, medical sciences, economy, social sciences, and even in the development of technology and business market [1].

The growth model is not highly important but a regression model which explains the relationship between two variables is the most important. One of these variables is called a response variable or dependent variable denoted by (Y), the other is called explanatory variable or independent variable denoted by (X). The only deference between a growth model and an ordinary regression model is that the independent variable in the growth model represents time which is denoted by (t) instead of (X). On these bases, the growth model will investigate the changes that will occur to the phenomenon under study represented by the dependent variable (Y) throughout a certain period of time represented by the independent variable (t). The general form of a growth model is given by [2, 3, 4]:

$$Y_t = f(t, \theta) + \varepsilon_t \ , t = 1,2, \dots, n \qquad (1)$$

where,

Y: The dependent variable (Phenomenon value).

$f(\cdot)$: The growth function.

t: The independent variable (Time).

θ: The parameter vector for the model.

n: The number of observations (Sample size).

ε: The random error term.

Like the ordinary regression models, the growth models can be linear or nonlinear depending on the nature of the phenomenon under study. In the case of linear models, the growth function will be a linear function of the unknown parameters. The growth function in the case of nonlinear models will be a nonlinear function of the unknown parameters. All of these parameters or at least one of them appear in the model expression in a nonlinear fashion. In other words, the growth function depends nonlinearly on one or more of these unknown parameters. Nonlinear models are used to represent the complex relationships between the variables which cannot be represented in a linear form [2, 5].

As for estimating the unknown growth parameters and according to the statistical theory for parameter estimation, it can be said that the procedure of parameter estimation in linear models is almost completely developed unlike parameter estimation in nonlinear models. There are many problems that researchers encounter that are still unsolved. In this paper, this problem will be briefly discussed in addition to conducting a comparison between the estimators of three traditional methods including the Newton-Raphson method, the Gauss-Newton method, Levenberg-Marquardt and the estimators of the Particle Swarm Optimization (PSO) algorithm to estimate the parameters of six growth models. Three of them are linear which will include polynomials of orders (1, 3, and 5), in addition to three nonlinear models that will include the Logistic model, the Gompertz model, and the Richards model [2, 4, 5, 6].

This paper is partitioned into five primary sections. The first section included the introduction and a brief outline on growth models and its properties. In 2nd section, the traditional estimation methods that will be used in this paper are discussed. In third section, the particle swarm optimization (PSO) algorithm will be explained in a brief manner along with all its properties and characteristics. The fourth section will include the numerical computations along with all the procedures that are necessary to conduct the comparison between the traditional estimators and the estimators of the (PSO) algorithm for the previously mentioned growth models for the Gross Domestic Product (GDP) in Iraq. The last section will include the conclusions of the paper.

2. Methods of estimation

In this section, the methods of Ordinary Least Squares (OLS) and Nonlinear Least Squares (NLS) will be discussed briefly. Both of these methods will be used to estimate the parameters of the linear and nonlinear growth models, respectively. The formulas for these growth models are presented in Table 1 [3, 4, 5, 6, 7]. In addition to that, three traditional iterative algorithms or numerical methods will be explained where they will be used in this paper to solve the system of nonlinear equations produced by the method of nonlinear least squares (NLS).

For linear models, the method of ordinary least squares is considered to be one of the most important estimation methods. It is commonly used to estimate the parameters of linear models because of its simplicity and easy to perform calculations. The basic principle of this method is that it tries to find or select the estimator that makes the sum of squares of residuals as small as possible [2, 4, 8].

$$min\left\{S(\theta) = \sum_{i=1}^{n} e_i^2 = \sum_{i=1}^{n}\left(Y_i - f(t_i, \theta)\right)^2\right\}$$

And that is by taking the partial derivatives of the function of sum of squares of residuals for each parameter of the model under study. From these partial derivatives, we get a system of linear equations which can be solved simultaneously using ordinary methods to get the (OLS) estimators that will make the sum of squares of residuals in its minimum value.

Table 1. List of the linear and nonlinear growth models used in this paper

	Model Name	Model Formula
Linear Models	First degree polynomial growth model	$Y_i = \theta_0 + \theta_1 t_i + \varepsilon_i$
	Third degree polynomial growth model	$Y_i = \theta_0 + \theta_1 t_i + \theta_2 t_i^2 + \theta_3 t_i^3 + \varepsilon_i$
	Fifth degree polynomial growth model	$Y_i = \theta_0 + \theta_1 t_i + \theta_2 t_i^2 + \theta_3 t_i^3 + \theta_4 t_i^4 + \theta_5 t_i^5 + \varepsilon_i$
Nonlinear Models	The Logistic Model	$Y_t = \frac{A}{1 + Be^{-Kt}} + \varepsilon_t$
	The Gompertz Model	$Y_t = Ae^{-Be^{-Kt}} + \varepsilon_t$
	The Richards Model	$Y_t = \frac{A}{(1 + Be^{-Kt})^{1/M}} + \varepsilon_t$

For a polynomial growth model of order (m), the model will be as follows [3]:

$$Y_i = \theta_0 + \theta_1 t_i + \theta_2 t_i^2 + \cdots + \theta_m t_i^m + \varepsilon_i \quad ,i = 1,2,\ldots,n \qquad (2)$$

It can be seen from (2) that the model in this case has one response variable and only one explanatory variable but it appears differently in each term of the model. According to this model, the OLS method will be inapplicable, but this model can be transformed to the general linear form very easily by using the following transformation:

$$Z_{ji} = t_i^j \quad , \ j = 1,2,\ldots,m \qquad (3)$$

After that, the model will be as follows:

$$Y_i = \theta_0 + \theta_1 Z_{1i} + \theta_2 Z_{2i} + \cdots + \theta_m Z_{mi} + \varepsilon_i \qquad (4)$$

or,

$$Y_i = \theta_0 + \sum_{j=1}^{m} \theta_j Z_{ji} + \varepsilon_i \qquad (5)$$

After that transformation, the OLS method can be used to estimate the parameters of the general linear model. And by using matrices and vectors, the model will be as follows:

$$\underline{Y} = Z\underline{\theta} - \underline{\varepsilon} \qquad (6)$$

$\boldsymbol{Y}$ is the dependent variable. $\boldsymbol{t}$ is the independent variable (time). $\boldsymbol{\theta_0}$ is the constant term of the model. $\boldsymbol{\theta_1}, \boldsymbol{\theta_2}, \boldsymbol{\theta_3}, \boldsymbol{\theta_4}, \boldsymbol{\theta_5}$ are the slops of the model. $\boldsymbol{A}$ is the upper asymptote to the function or it is the upper limit that the growth can reach. $\boldsymbol{B}$ is and integration constant, and its value is related to the initial value of the phenomenon. $\boldsymbol{K}$ is the growth rate. $\boldsymbol{M}$ is a shape parameter. $\boldsymbol{\varepsilon}$ is the error term of the model.

By assuming that the random errors of the model (ε) are independent and identically distributed with mean (0) and a constant variance equal to the matrix ($\sigma^2 I_n$), where (I_n) is an identity matrix of order (n), and according to the (OLS) method, the parameters of the model can be estimated using the following formula:

$$\hat{\theta} = (Z'Z)^{-1}Z'Y \tag{7}$$

For nonlinear models, the method of nonlinear least squares (NLS) is considered to be one of the most important and widely used methods for estimating the parameters of nonlinear models [7]. Just like the (OLS) method, the basic principle of the (NLS) method is that it tries to find or select the estimator that makes the sum of squares of residuals as small as possible [2, 4, 8].

$$min\left\{S(\theta) = \sum_{t=1}^{n} e_t^2 = \sum_{t=1}^{n}\left(Y_t - f(t,\theta)\right)^2\right\}$$

And that is by taking the partial derivatives of the function of sum of squares of residuals for each parameter of the model under study, as a result of these partial derivatives we get a system of nonlinear equations which will be very hard if not impossible to solve using ordinary methods. This problem is one of the most important problems that researchers encounter when they try to estimate the parameters of nonlinear models and because of the importance of the parameter estimation procedure of nonlinear models in the inferences about these models, there had to be a way to solve the system of nonlinear equations resulting from the (NLS) method. The parameter estimation procedure for nonlinear models is considered as an optimization problem that minimizes the objective function, which in this case is the function of sum of squares of the residuals ($S(\theta)$), to its minimum value. On these bases, numerical methods are often used to obtain approximated solutions for these nonlinear equations and as a result obtain the (NLS) estimators which minimize the sum of squares of the residuals to its minimum value. And now, some brief information about the three traditional methods or algorithms which will be used in this paper.

2.1 Newton-Raphson method

This method is one of the most popular iterative numerical methods or iterative algorithms that are used for finding the roots of a real-valued function. It is named after (Isaac Newton) and (Joseph Raphson) and it is sometimes called (Newton's Method). This method produces successively better approximations which represent the roots of the function under study. This procedure is repeated until a sufficiently accurate value is reached and that value would be the root to the function under study. To find the solutions to the system of nonlinear equations that result from the (NLS) method, the nonlinear form of the equations is approximated to the linear form using the first-order Taylor's series expansion as follows [3, 6]:

$$f(t,\theta) \approx f(t,\theta_0) + (\theta - \theta_0)f'(t,\theta_0) \tag{8}$$

Where ($f'(t,\theta_0)$) is the first derivative of the function ($f(t,\theta_0)$) at the point (θ_0) which represents the initial guess of the estimator's value. By letting ($\theta = \theta_1$) and then equating (8) to zero and solving for (θ_1), we obtain:

$$\theta_1 = \theta_0 - \frac{f(t,\theta_0)}{f'(t,\theta_0)} \tag{9}$$

And when generalizing (9) to the $(It)^{th}$iteration or linear approximation, we obtain the general form of the Newton-Raphson method:

$$\theta_{It} = \theta_{It-1} - \frac{f(t, \theta_{It-1})}{f'^{(t,\theta_{It-1})}} \quad , It \in N \tag{10}$$

Where (N) is the set of natural numbers. This process continues until one of these conditions is satisfied:

1) $\theta_{It} - \theta_{It-1} = 0$.

2) $\left|\frac{\theta_{It}-\theta_{It-1}}{\theta_{It-1}}\right| < \delta$, where (δ) is a very small positive number known as the tolerance limit.

And for a system of nonlinear equations with (r) unknowns, (10) becomes:

$$\theta^{(It)} = \theta^{(It-1)} - J\left(\theta^{(It-1)}\right)^{-1} F\left(\theta^{(It-1)}\right) \tag{11}$$

Where $(It = 1,2, ...)$ is the iterations, (θ) is a $(r \times 1)$ vector of unknowns, (F) is a $(n \times 1)$ vector of (n) nonlinear functions, $(J(\theta)^{-1})$ is the inverse of the $(r \times n)$ Jacobian matrix.

2.2. Gauss-Newton method

This method is one of the most common and widely used iterative numerical methods or iterative algorithms for finding solutions for least squares problems and it is considered as a modification to the Newton-Raphson method. The main goal of this method is to find the minimum value of some function. This method is named after (Carl Friedrich Gauss) and (Isaac Newton), and it appeared for the first time in Gauss's (1809) work. This method is specialized in finding solutions for nonlinear least squares problems and unlike the Newton-Raphson method it is only used for finding the minimum of a sum of squares function. Let's assume the model in (1), the objective function in this case is the sum of squares of the residuals $(S(\theta))$. For the purpose of finding the solutions to the system of nonlinear equations produced by the (NLS) method, like in the Newton-Raphson method, the nonlinear form of the equations is approximated to the linear form using the first-order Taylor's series expansion just like in (8), we get [3, 6]:

$$f(t, \theta) = f(t, \theta_0) + z_i(\theta_0)(\theta - \theta_0) \tag{12}$$

Where $(z_i(\theta_0) = f'(t, \theta_0))$ is the first derivative of the function with respect to (θ_0) which is the initial guess for the estimator's value. After that we obtain the linear model:

$$Y_i = f(t_i, \theta_0) + z_i(\theta_0)(\theta - \theta_0) + \varepsilon_i, i = 1,2, \ldots, n \tag{13}$$

And by applying the (OLS) method to find the estimator of the model's parameter (θ) which will minimize $(S(\theta))$ to its minimum value and by assuming $(\theta = \theta_1)$, we obtain:

$$\theta_1 = \theta_0 + \left(Z(\theta_0)'Z(\theta_0)\right)^{-1} Z(\theta_0)'\left(Y - f(t, \theta_0)\right) \tag{14}$$

Where $(Z(\theta_0)' = (z_1(\theta_0), \ldots, z_n(\theta_0)))$, $(Y' = (Y_1, \ldots, Y_n))$, and $(f(t, \theta_0)' = (f(t_1, \theta_0), \ldots, f(t_n, \theta_0)))$. And when generalizing (14) to the $(It + 1)^{th}$iteration or linear approximation, we obtain the general form of the Gauss-Newton method:

$$\theta_{It+1} = \theta_{It} + \left(Z(\theta_{It})'Z(\theta_{It})\right)^{-1} Z(\theta_{It})'\left(Y - f(t, \theta_{It})\right) \tag{15}$$

And this process continues until one of these conditions is satisfied:

1) $\theta_{It+1} - \theta_{It} = 0$, and thus $(Z(\theta_{It})'\left(Y - f(t, \theta_{It})\right) = 0)$ which means that (θ_{It+1}) is the estimator that will minimize $(S(\theta))$ to its minimum value.

2) $\left|\frac{\theta_{It+1}-\theta_{It}}{\theta_{It}}\right| < \delta$, where (δ) is a very small positive number known as the tolerance limit.

And for a system of nonlinear equations with (r) unknowns, (15) becomes as follows:

$$\theta_{It+1} = \theta_{It} + \left(Z(\theta_{It})'Z(\theta_{It})\right)^{-1} Z(\theta_{It})'\left(Y - f(t, \theta_{It})\right) \qquad (16)$$

Where ($It = 1,2, \ldots$) is the iterations, (θ) is a ($r \times 1$) vector of unknowns, ($f(t, \theta_{It})$) is a ($n \times 1$) vector of (n) nonlinear functions, ($Z(\theta_{It})$) is a ($r \times n$) Jacobian matrix, and (Y) is a ($n \times 1$) vector of observations.

2.3. Levenberg-Marquardt method

This method is one of the most popular iterative numerical methods or iterative algorithms which is used for solving least squares problems. This method was first published in 1944 by Levenberg and later it was rediscovered in 1963 by Marquardt. This method is considered to be a mix of Gauss-Newton method and the method of Gradient Descent, which is also one of the traditional iterative numerical methods. The main steps of this method are the same as the Gauss-Newton method but with some differences in the final formula. Levenberg suggested a modification on the Gauss-Newton formula as follows [3, 6]:

$$\theta_{It+1} = \theta_{It} + (Z(\theta_{It})'Z(\theta_{It}) + \lambda I_r)^{-1} Z(\theta_{It})'\left(Y - f(X, \theta_{It})\right) \qquad (17)$$

Where (λ) is a conditioning factor, and (I_r) is an identity matrix of order (r). As an alternative to Levenberg's addition, Marquardt suggested the following modification on the Gauss-Newton formula:

$$\theta_{It+1} = \theta_{It} + (Z(\theta_{It})'Z(\theta_{It}) + \lambda D)^{-1} Z(\theta_{It})'\left(Y - f(X, \theta_{It})\right) \qquad (18)$$

Where (D) is a diagonal matrix with entries consisting of the diagonal elements of ($Z(\theta_{It})'Z(\theta_{It})$). (18) Is what is known as the general form of the Levenberg-Marquardt method. As it was mentioned before, the Levenberg-Marquardt direction interpolates between the Gauss-Newton direction ($\lambda \to 0$) and the direction of the Gradient Descent ($\lambda \to +\infty$) [3].

It is worth mentioning that the Levenberg-Marquardt method is considered to be more robust than the Gauss-Newton method when it comes to finding the estimators. Thus, this method is used if the columns of the Jacobian matrix of ($f(t, \theta)$) with respect to (θ), ($Z(\theta)$), are highly collinear. This means that the Levenberg-Marquardt method will be able to find a solution even if it started with an initial value far from the real minimum of the function under study. Nevertheless, if it starts with a reasonably close initial value, the Gauss-Newton method will find a solution faster than the Levenberg-Marquardt method [3].

3. Particle Swarm optimization algorithm

The PSO algorithm is an iterative computational method that is used to solve different optimization problems. This algorithm was introduced for the first time by James Kennedy and Russell C. Eberhart in 1995 [9, 10, 11]. The algorithm simulates the social behaviors and movements of organisms in a bird flock or fish school. Its main aspects are based on the way that bird flock or fish school search for their food and the cooperation they exhibit in their movements to find the best sources for food. For birds, some of them have the ability to find food better than the others because it has the biological property of recognizing the smell of food in a very active way, in addition to a special way of communicating with each other which helps them to find and exploit the best sources of food. The same thing happens when solving optimization problems where a population (swarm) of candidate solutions (birds or particles inside the swarm) is moved around a limited space called (search space) according to a few simple mathematical formulas about the position and velocity of each particle. All of the movements of each particle (solution) are affected by the personal best position that that particle has found, in addition to that, each particle is guided to the best positions in the search space which are updated when other particles in the swarm find better and better positions. These movements eventually lead the swarm to the best food source or the best solution. In what follows, the main components of the (PSO) algorithm will be explained, its parameters will also be mentioned and after that the steps of the algorithm will be shown.

The (PSO) algorithm consists of a group of (s) candidate solutions for the problem under study, those solutions are called particles or birds, which in sum they will make the swarm. These particles move freely in

a d-dimentional area or space known as the search space which is determined according to the problem being solved. This space represents the set of all possible solutions to the problem under study. The particles in the swarm have a special feature and that is each particle has its own personal experience which it uses while searching in addition to using the experiences of other neighboring particles in the swarm [10, 11].
In the beginning, the (PSO) algorithm is initialized by a number of particles randomly chosen in the search space where each chosen particle has a position in the search space denoted by (p_{i}) that consists of ($p_{\mathrm{i}} = p_{\mathrm{i},1}, p_{\mathrm{i},2}, \dots, p_{\mathrm{i,d}}$). In addition to a velocity denoted by (v_{i}) which consists of ($v_{\mathrm{i}} = v_{\mathrm{i},1}, v_{\mathrm{i},2}, \dots, v_{\mathrm{i,d}}$). The position and velocity of each particle is updated according to the personal best position achieved by that particle up until now denoted by ($P_{\mathrm{best,i,j}}$) as well as the best position achieved by any particle in the whole swarm up until now denoted by ($G_{\mathrm{best,j}}$) in accordance to the dimensions of the problem (d). The velocity and position of each particle is updated using the following two equations, respectively [5, 9, 10, 11, 12]:

$$v_{\mathrm{i,j}}^{It+1} = w v_{\mathrm{i,j}}^{It} + c_1 r_{1,j}^{It}\left(P_{\mathrm{best,i,j}}^{It} - p_{\mathrm{i,j}}^{It}\right) + c_2 r_{2,j}^{It}\left(G_{\mathrm{best,j}}^{It} - p_{\mathrm{i,j}}^{It}\right) \qquad (19)$$
$$p_{\mathrm{i,j}}^{It+1} = p_{\mathrm{i,j}}^{It} + v_{\mathrm{i,j}}^{It+1} \qquad (20)$$

where,

$v_{\mathrm{i,j}}^{It}$: The velocity of the particle (i) in the swarm in dimension (j) at the $(It)^{th}$ iteration (the current velocity of particle (i)).

w: The inertia weight that shows the effect of the previous velocity ($v_{\mathrm{i,j}}^{It}$) on the new velocity ($v_{\mathrm{i,j}}^{It+1}$).

$p_{\mathrm{i,j}}^{It}$: The position of particle (i) in the swarm in dimension (j) at the $(It)^{th}$ iteration (the current position of particle (i)).

c_1, c_2: Two positive constants known as the Acceleration Coefficients (c_1 is the Cognitive Component and c_2 is the Social Component).

$r_{1,\mathrm{j}}^{It}, r_{2,\mathrm{j}}^{It}$: Two randomly generated numbers uniformly distributed between (0) and (1) in dimension (j) at the $(It)^{th}$ iteration.

$P_{\mathrm{best,i,j}}^{It}$: The personal best position achieved by particle (i) in dimension (j) up until the $(It)^{th}$ iteration, also called the Local Best Position.

$G_{\mathrm{best,j}}^{It}$: The best position achieved by any particle in the whole swarm in dimension (j) up until the $(It)^{th}$ iteration, also called the Global Best Position.

Basically, the (PSO) algorithm has five main parameters that are determined by the researcher according to the nature of the problem being solved [5, 10, 11]. These parameters are: The Swarm Size which represents the number of particles or birds in the swarm and it is denoted by (s). This size must not be too small that it affects the solving procedure or too big that it complicates the computations in each iteration which will result in a loss of time and effort. The Iteration Number represents the number of times that the work of the algorithm is repeated and it is denoted by (It_{max}). However, this number must not be too small that the algorithm stops before it reaches the optimum solution and it also must not be too big that it consumes a lot of time and effort. The Acceleration Coefficients are two positive constants that help the algorithm to maintain two things, the first is the memory of the particle itself which is represented by the cognitive component (c_1) also known as the personal learning factor. The second is the social structure between the particles in the swarm which is represented by the social component (c_2) and also called the social learning factor. when determining the values of (c_1) and (c_2) one must take into account that small values allows the swarm to roam a wide area in the search space, while large values result in the abrupt movement of the swarm in the search space. It was also noticed from the literature that selecting the value of (c_1) equal to (c_2) and ranging between

(0) and (4) will give reasonably good results. The Inertia Weight is a parameter that was first introduced by (Shi) and (Eberhart) in (1998) [12]. Its purpose was to enhance the work of the algorithm by adding a constant value to the velocity update equation. This parameter is denoted by (w) and its importance lies in maintaining the balance between the global (exploration) and local (exploitation) search of the particle. There are many ways for selecting the inertia weight. In this paper, we'll use a linearly decreasing inertia weight which is calculated as follows [5, 12]:

$$w = w_{max} - \left(\frac{w_{max} - w_{min}}{It_{max}}\right) It \qquad (21)$$

Where (w_{max}) is the upper limit or the initial value of the inertia weight, (w_{min}) is the lower or the final value of the inertia weight, (It) is the current iteration, and (It_{max}) is the upper limit of the number of iterations [5, 12].

In addition to everything previously mentioned, the most important component of the (PSO) algorithm is the Fitness Function, also known as the Objective Function. This function is considered to be one of the most important elements of the (PSO) algorithm and it has a direct effect on the work of the algorithm. Using this function, the (PSO) algorithm evaluates the particles in the swarm and compares them to each other in order to determine the best positions which in return will represent the optimum solution to the problem [5, 9, 10, 11]. The fitness function is determined according to the type of the problem under study where this function differs from one problem to another. In this paper, the function of sum of squares of the residuals $\left\{S(\theta) = \sum_{t=0}^{n} e_t^2 = \sum_{t=0}^{n}\left(Y_t - f(t,\theta)\right)^2\right\}$ will be chosen as a fitness function for the algorithm and depending on the used model [5]. The flow chart of the (PSO) algorithm is shown in Figure 1.

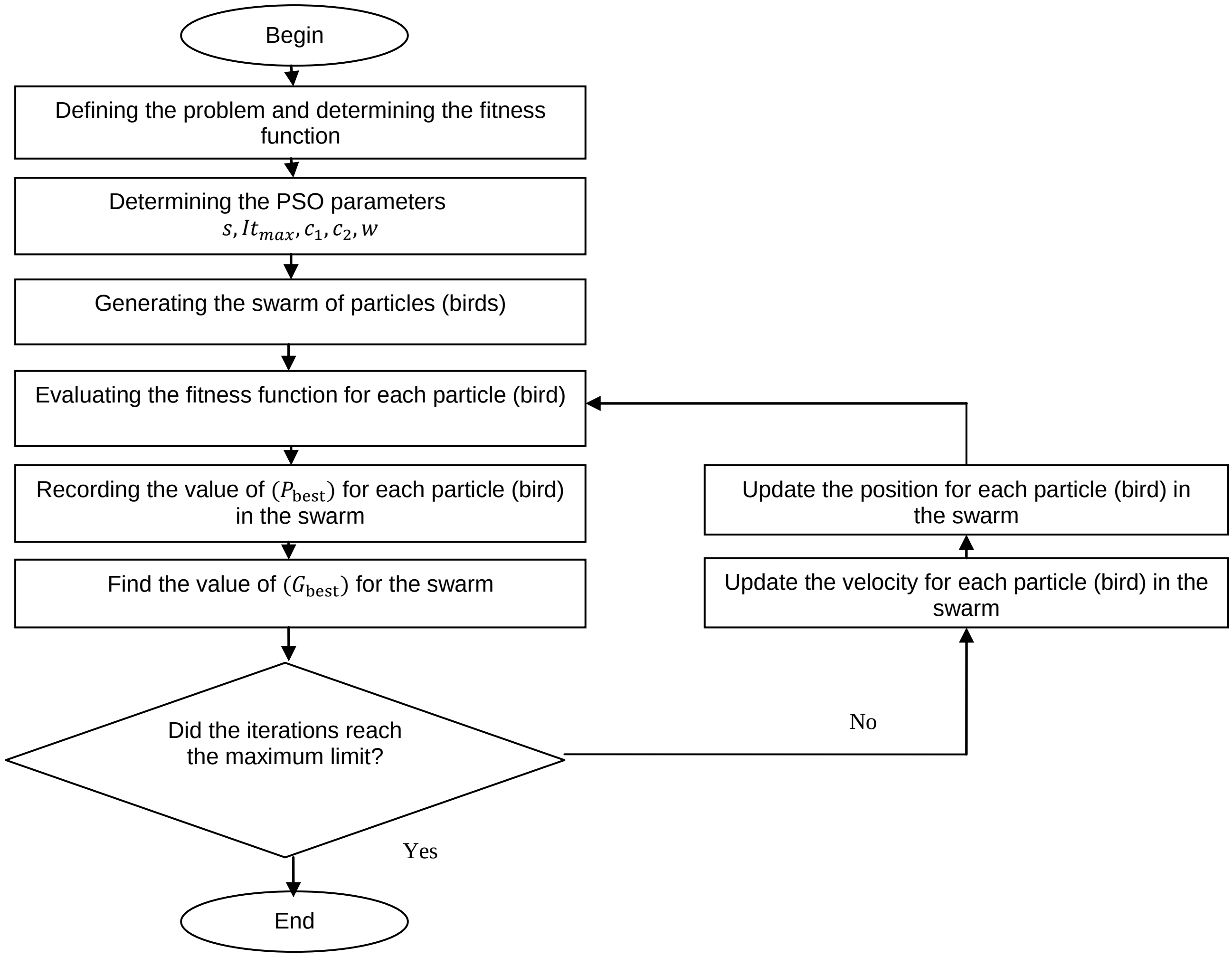


Figure 1. Flow chart of the PSO algorithm

4. Results and discussion

In this section of the paper, the parameters of the growth models mentioned in Table 1 will be estimated using the (OLS) method for the linear models and the (NLS) method for the nonlinear models. After that a comparison will be conducted between the estimates of the traditional methods and the estimates of the (PSO) algorithm for the nonlinear models and that is by using the Mean Squared Error (MSE) and then the Coefficient of Determination (R^2) will be used to compare the six growth models for the purpose of finding which one of these models is the best to describe the Gross Domestic Product data in Iraq. To calculate the (MSE) the following formula will be used [2, 3, 8]:

$$MSE(\hat{\theta}) = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2 \qquad (22)$$

where:

Y_i is the real value for observation (i).

$\hat{Y}_i$ is the estimated value for observation (i).

As for the (R^2), it will be calculated using the following formula:

$$R^2 = 1 - \frac{\text{SSE}}{\text{SST}}. \qquad (23)$$

where:

SSE is the Sum of Squared Errors, ($\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2 = \sum_{i=1}^{n} e_i^2$).

SST is the Total Sum of Squares, ($\sum_{i=1}^{n}(Y_i - \bar{Y})^2$).

Concerning the data of the research, a sample was taken consisting of the seasonal data for the Gross Domestic Product in Iraq in nominal values for ten years (2008 – 2017). The data was obtained from the Central Statistical Organization in Iraq and it is shown in Table 2.

In the beginning, it must be mentioned that the statistical analysis was done using the MATLAB R2019a program. In regard to the linear growth models, the polynomial growth models was transformed to the linear form by using the transformation explained in (3) and then the parameters of the three linear models will be estimated using the (OLS) method in addition to calculating the (MSE) and the (R^2) for each model. The results are listed in Table 3 and the growth curve for each model can be seen in Figure 2.

Regarding the nonlinear growth models, the (NLS) method was used to estimate the parameters of the three nonlinear models as a primary method. As to solving the system of nonlinear equations resulting from this method, three traditional numerical methods in addition to the (PSO) algorithm were all used to find approximated solutions for these equations and then compare the results of these four methods for the purpose of knowing which one of these methods is the best for estimating the parameters of the nonlinear models. The traditional methods were used to estimate the parameters of the three nonlinear growth models. Furthermore, the (MSE) and the (R^2) for each model were also calculated and the results for the Newton-Raphson method, the Gauss-Newton method, and the Levenberg-Marquardt method are listed in Tables 4, 5, and 6, respectively.

In regard to the parameters of the PSO algorithm, the values for these parameters were determined based on the literature concerning this subject as follows:

In this paper, the swarm size (s) was adjusted to 100. The iteration number (It_{max}) was set to 500 iterations. Regarding the inertia weight, (w_{max}) and (w_{min}) were selected as 0.9 and 0.4 respectively. The values for the acceleration coefficients (c_1) and (c_2) were both chosen equal to 2. As for the termination criterion, it was selected as the iteration limit which means that the algorithm will stop after 500 iterations and the results of the analysis are listed in Table 7.

Table 2. The seasonal data for the Gross Domestic Product in Iraq in nominal values for ten years (2008 – 2017) that are used in the paper

Years	Seasons (Time) t	Gross Domestic Product (GDP) Y	The natural Log for (Y) ln(Y)
2008	1	37653079.3	17.4439
	2	46961825.2	17.6648
	3	44072423.4	17.6013
	4	28338733.7	17.1597
2009	5	25743384.7	17.0637
	6	32127325.1	17.2852
	7	35384541.5	17.3818
	8	37387949.1	17.4369
2010	9	37712855.1	17.4455
	10	39555111.2	17.4932
	11	39930096.8	17.5026
	12	44866502.4	17.6192
2011	13	47933786.1	17.6853
	14	56180513.5	17.8441
	15	55838001.9	17.8380
	16	57374805.9	17.8651
2012	17	57635096.6	17.8696
	18	63028554.2	17.9591
	19	65952875.3	18.0045
	20	67608964.6	18.0293
2013	21	65173143.2	17.9926
	22	68881169.8	18.0479
	23	69674467.4	18.0593
	24	69858748.8	18.0620
2014	25	68005401.5	18.0351
	26	71237512.5	18.0815
	27	65603229.2	17.9991
	28	61574241.3	17.9358
2015	29	43501008.4	17.5883
	30	54533786.5	17.8143
	31	51710622.2	17.7612
	32	49970282.8	17.7269
2016	33	44830037.0	17.6184
	34	50130996.6	17.7302
	35	51835211.0	17.7636
	36	57073587.6	17.8599
2017	37	51724334.1	17.7614
	38	53154906.5	17.7887
	39	57491418.9	17.8671
	40	63624519.6	17.9685

The natural Log was used for the data for easier calculations.

Table 3. The estimates of the linear growth models along with the values of (MSE) and (R^2) for each model

Model Name	Parameters	Estimates	MSE	R^2
First degree polynomial	θ_0	17.4793	0.0436	0.3333
	θ_1	0.0128		
Third degree polynomial	θ_0	17.21638	0.0277	0.5765
	θ_1	0.045117		
	θ_2	-0.0004		
	θ_3	-0.00001		
Fifth degree polynomial	θ_0	17.81138	0.0088	0.8656
	θ_1	-0.22877		
	θ_2	0.03117		
	θ_3	-0.00138		
	θ_4	0.00002		
	θ_5	-0.00000		

These results were obtained using the natural Log of the data ($\ln(Y)$) for easier calculations.

Table 4. The estimates of the nonlinear growth models along with the values of (MSE) and (R^2) for each model using Newton-Raphson method

Model name	Parameters	Estimates	MSE	R^2	Iteration number	Total run time
Logistic	A	17.9211	0.0332	0.4917	7	16 s.
	B	0.042648				
	K	0.097134				
Gompertz	A	17.9219	0.0333	0.4908	9	18.9 s.
	B	0.041791				
	K	0.095687				
Richards	A	17.8993	0.0302	0.5374	18	73.19 s.
	B	11.6741				
	K	0.19793				
	M	58.5538				

These results were obtained using the natural Log of the data ($\ln(Y)$) for easier calculations.

Table 5. The estimates of the nonlinear growth models along with the values of (MSE) and (R^2) for each model using Gauss-Newton method

Model name	Parameters	Estimates	MSE	R^2	Iteration number	Total run time
Logistic	A	17.9201	0.0332	0.4917	50	24.6 s.
	B	0.042687				
	K	0.097801				
Gompertz	A	17.9307	0.0333	0.4900	50	24.55 s.

Model name	Parameters	Estimates	MSE	R^2	Iteration number	Total run time
	B	0.04129				
	K	0.089002				
Richards	A	-	-	-	-	-
	B	-				
	K	-				
	M	-				

These results were obtained using the natural Log of the data ($\ln(Y)$) for easier calculations.

Table 6. The estimates of the nonlinear growth models along with the values of (MSE) and (R^2) for each model using Levenberg-Marquardt method

Model name	Parameters	Estimates	λ	MSE	R^2	Iteration number	Total run time
Logistic	A	17.9224	0.7	0.0332	0.4915	50	24.7 s.
	B	0.043494					
	K	0.098565					
Gompertz	A	17.9223	0.6	0.0333	0.4906	50	26.2 s.
	B	0.042599					
	K	0.097549					
Richards	A	17.9193	0.7	0.0330	0.4951	50	31.3 s.
	B	0.22301					
	K	0.10083					
	M	4.8783					

These results were obtained using the natural Log of the data ($\ln(Y)$) for easier calculations.

From the results in **Tables (4)**, **(5)**, **(6)**, and **(7)** we notice the following:
As far as the comparison of the estimation methods go and regarding the traditional methods, we notice that the Newton-Raphson method was the best in almost all of the aspects. The (MSE) values for the three traditional methods were all equal in the Logistic model and the Gompertz model, but they were different in the Richards model in favor of the Newton-Raphson method, where the (MSE) value was larger in the Levenberg-Marquardt method than in the Newton-Raphson method. On the other hand the Gauss-Newton method was unable to find good estimates for the parameters of the Richards model and that is due to the Jacobian matrix becoming singular or close to singular. In addition to the value of the (MSE), the convergence of the Newton-Raphson method was also better than the other two methods since it converged after only (7) iterations for the Logistic model, (9) iterations for the Gompertz model, and (18) iterations for the Richards model while the Gauss-Newton method and the Levenberg-Marquardt method converged after (50) iterations. The only downside to the Newton-Raphson method was the run time as this method was the slowest when it

comes to execution speed compared to the Gauss-Newton method and the Levenberg-Marquardt method. The growth curves for each nonlinear model estimated using the traditional methods can be seen in Figures 3, 4, 5, and 6.

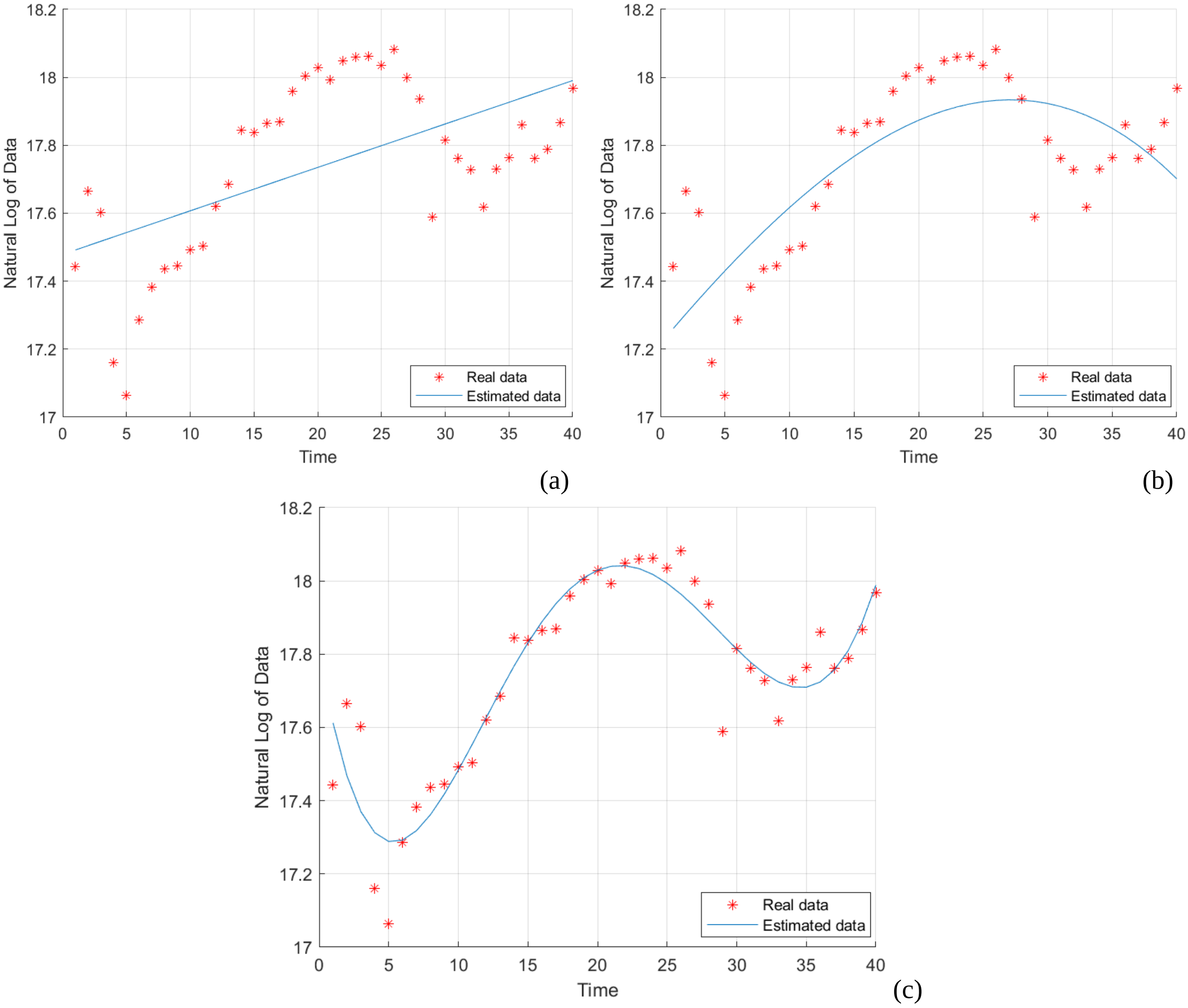
Table 7. The estimates of the nonlinear growth models along with the values of (MSE) and (R^2) for each model using the (PSO) algorithm

Model name	Parameters	Estimates	MSE	R^2	Iteration number	Total run time
Logistic	A	17.9211	0.0332	0.4917	500	4.3 s.
	B	0.042648				
	K	0.097134				
Gompertz	A	17.9219	0.0333	0.4914	500	4.3 s.
	B	0.041791				
	K	0.095687				
Richards	A	17.898	0.0300	0.5413	500	4.5 s.
	B	10.652				
	K	0.19352				
	M	60				

These results were obtained using the natural Log of the data ($\ln(Y)$) for easier calculations.

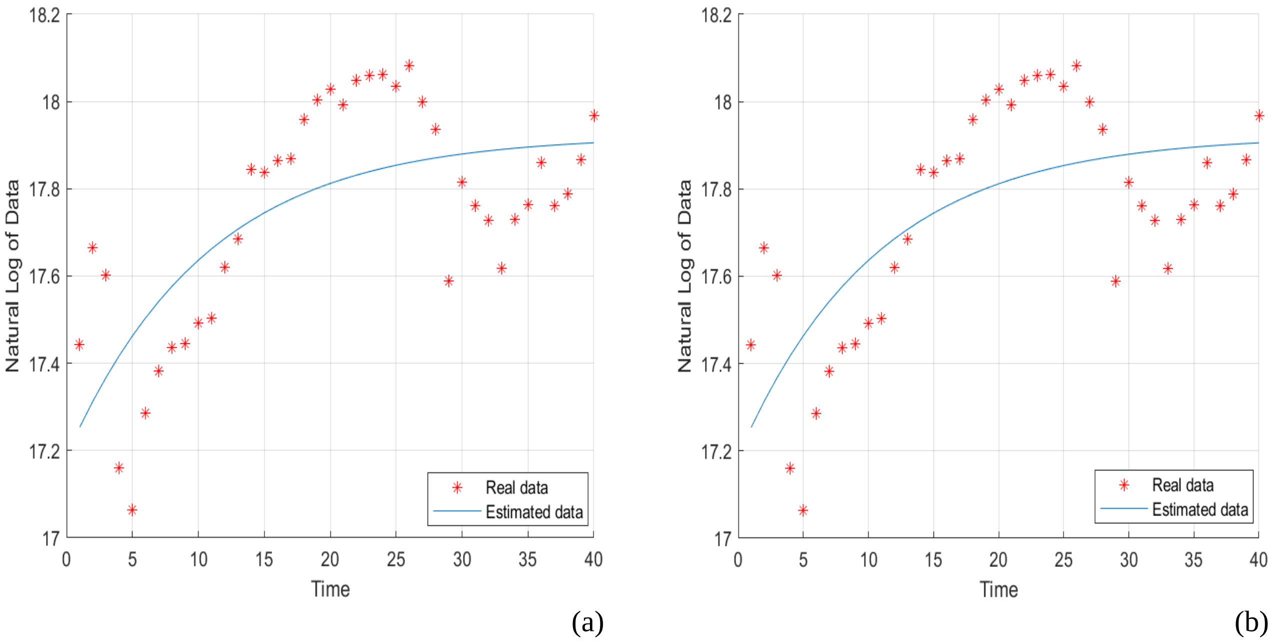
As for the (PSO) algorithm, it was the best in all aspects compared to the traditional methods. The results of the (PSO) algorithm were close to that of the Newton-Raphson method with some differences in favor of the (PSO) algorithm. It was also better than all the traditional methods when it comes to execution speed for this algorithm ran (500) iterations and managed to find better results than the other methods in no more than (5) seconds and that is due to its simple calculations because it doesn't need to calculate the partial derivatives for the functions and that is what makes this algorithm faster and more efficient than the traditional methods.

In regard to the comparison of the six growth models and which one of them is the best to represent the (GDP) data in Iraq. The results are listed in Tables 3, 4, 5, 6, and 7. It can be noticed that all the nonlinear growth models were better than the first degree polynomial in representing the data because it had larger (R^2) values, but these values were less than the values of the third and fifth degree polynomial models. The best of the nonlinear models was the Richards model with (R^2) values ranging between (0.5413) and (0.4951) for all the estimation methods. The second best was the Logistic model with (R^2) values ranging between (0.4917) and (0.4915). The last was the Gompertz model with (R^2) values ranging between (0.4914) and (0.4900). But all of these values are not good or insufficient statistically and that is because these models do not explain or interpret or account for almost half of the total variations in the data. With that being said, the best growth model which represented the data was the fifth degree polynomial growth model with a value for (R^2) equal to (0.8656) which means that this model explains (86.56%) of the total variations in the data and that is a good result in general. It should be mentioned that the Richards model is considered to be more flexible than the other nonlinear models where it was noticed during the experiment that it is possible to increase the value of (R^2) for this model by increasing the value of the shape parameter (M) that controls the position of the infliction point in the growth curve of the model.

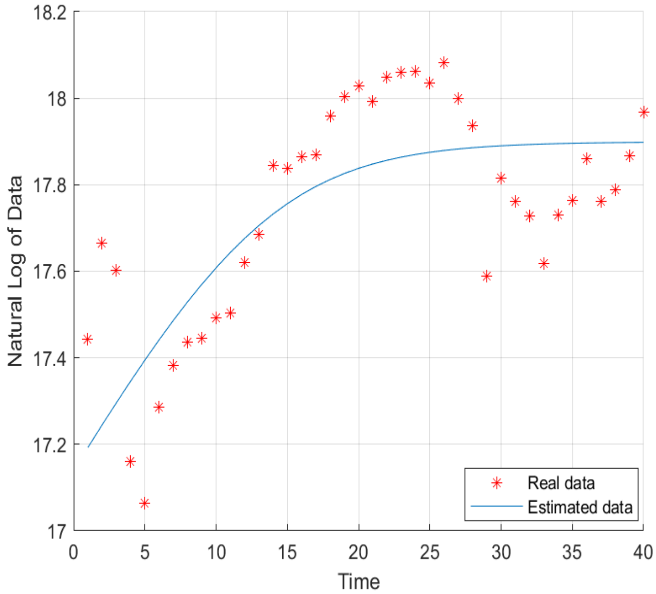


(a) (b) (c)

Figure 2. The growth curves for the linear models, (a) 1st degree polynomial, (b) 3rd degree polynomial, and (c) 5th degree polynomial

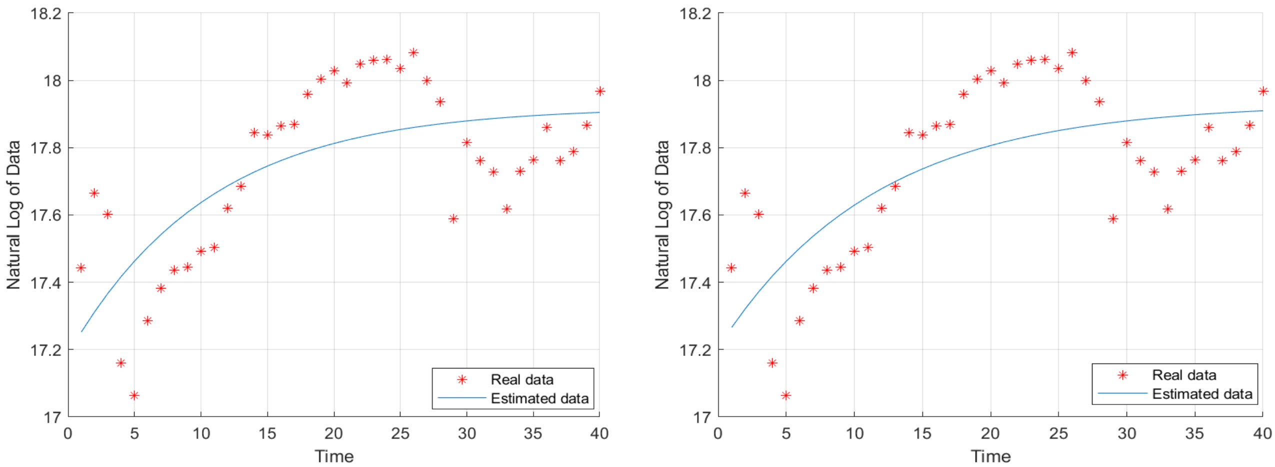


(a) (b)



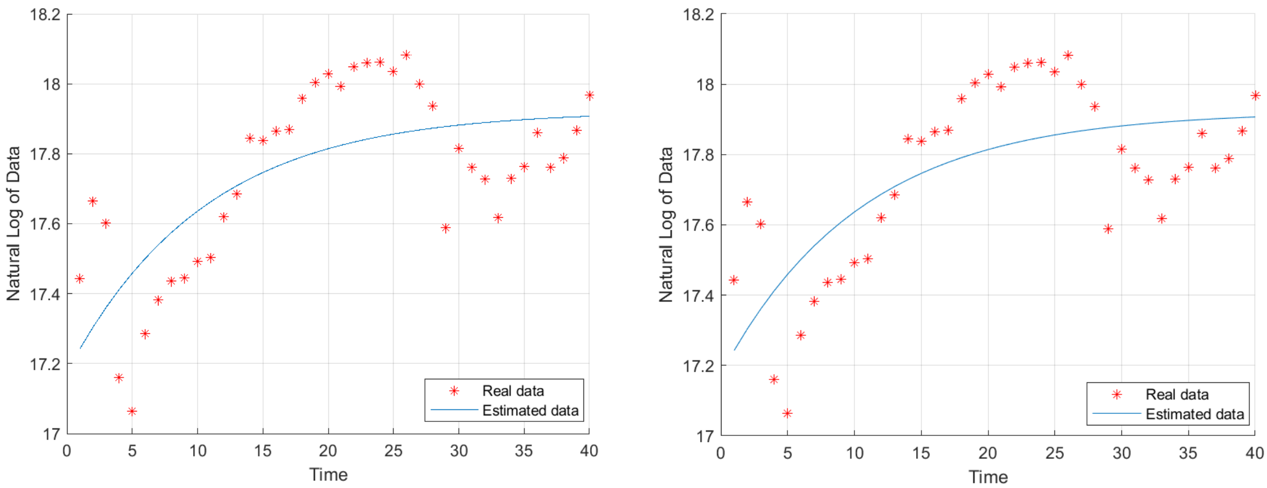
(c)

Figure 3. The growth curves for the nonlinear models using Newton-Raphson method, (a) Logistic model, (b) Gompertz model, and (c) Richards model

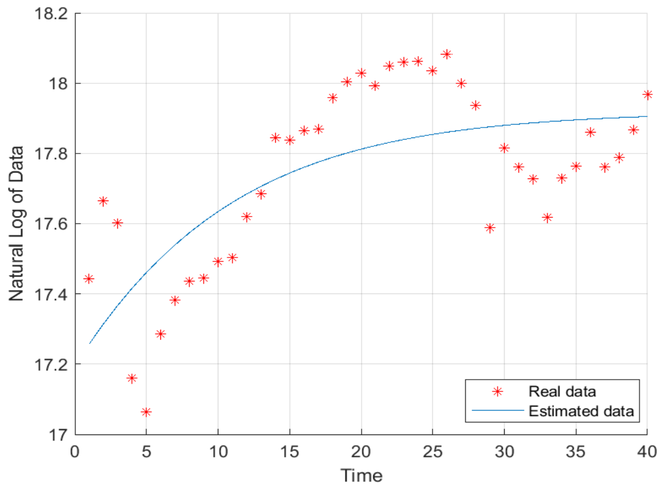


(a) (b)

Figure 4. The growth curves for the nonlinear models using Gauss-Newton method, (a) Logistic model, and (b) Gompertz model

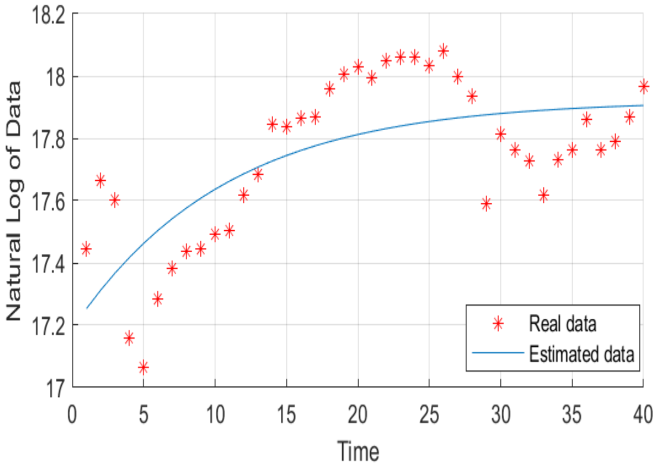


(a) (b)

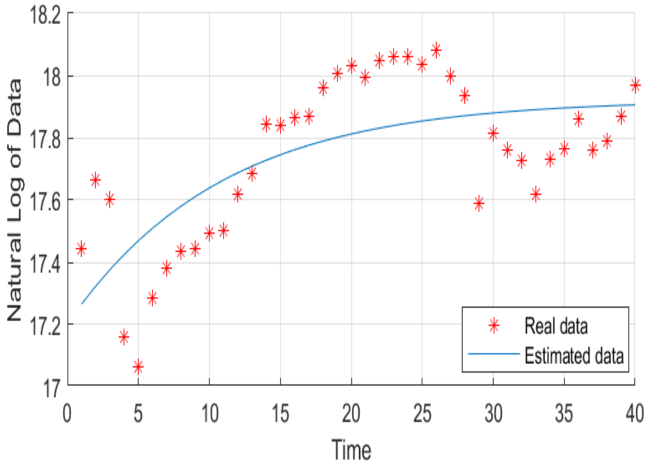


(c)

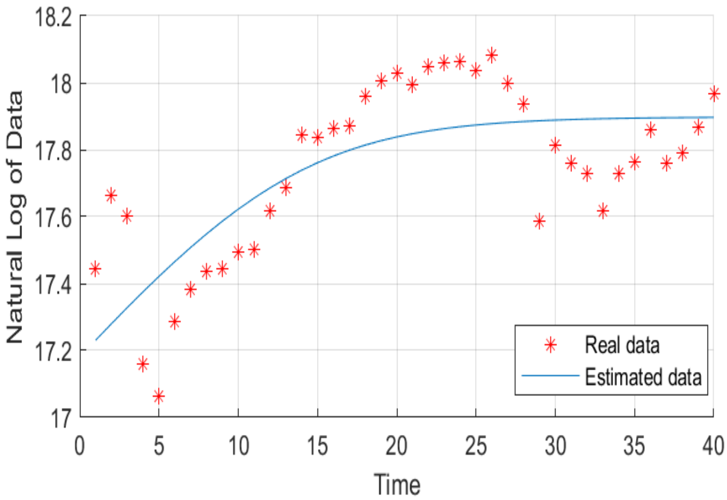
Figure 5. The growth curves for the nonlinear models using Levenberg-Marquardt method, (a) Logistic model, (b) Gompertz model, and (c) Richards model



(a)



(b)



(c)

Figure 6. The growth curves for the nonlinear models using the (PSO) algorithm, (a) Logistic model, (b) Gompertz model, and (c) Richards model

5. Conclusions

In this paper, some common growth models were studied in order to compare parameter estimation methods for these models using three traditional methods and the Particle Swarm Optimization (PSO) algorithm. It was seen that the (PSO) algorithm gives better results than the traditional methods for all models and that is because the (PSO) algorithm does not require any additional complicated calculations to be performed which makes its execution simpler, faster and more efficient. Furthermore, a comparison between three linear growth models and three nonlinear growth models was conducted for the purpose of building the best growth model for the data of the Growth Domestic Product (GDP) in Iraq. The results show that the fifth degree polynomial growth model was the best to represent the (GDP) data and with the possibility of using the Richards model with the condition of selecting a relatively large value for the shape parameter of the model in order to increase the flexibility of the Richards growth curve to better represent the (GDP) data in Iraq.

References

[1] K. E. Castellano and A. D. Ho, A Practitioner’s Guide to Growth Models, Wash., DC: Council of Chief State School Officers, 2013.

[2] D. N. Gujarati and D. C. Porter, Basic Econometrics, 5th edition, Mass: McGraw-Hill, Boston, 2009.

[3] M. J. Panik, Growth curve modeling: theory and applications, Wiley, 2014.

[4] G. A. F. Seber and C. J. Wild, Nonlinear regression, Wiley, Hoboken, New Jersey, 2003.

[5] V. S. Ozsoy and H. Orkcu, "Estimating the parameters of nonlinear regression models through particle swarm optimization", *Gazi University Journal of Science*, vol. 29, no. 1, pp. 187–199, 2016.

[6] E. M. T. Hendrix and B. G.-Tóth, Introduction to nonlinear and global optimization, vol. 37, Springer, 2010.

[7] N. G. Hossein-Zadeh, "Modeling the growth curve of Iranian Shall sheep using non-linear growth models", *Small Ruminant Research*. 130: 60-66, 2015.

[8] J. L. Devore and K. N. Berk, Modern mathematical statistics with applications, 2nd edition, Springer, London, 2012.

[9] R. Eberhart and J. Kennedy, "A New Optimizer Using Particle Swarm Theory", *Proceedings of 6th International Symposium on Micro Machine and Human Science*, Nagoya, Japan, IEEE Service Center Piscataway NJ, 39-43, 1995.

[10] J. Kennedy and R. Eberhart, "Particle swarm optimization", *Neural Networks, Proceedings., IEEE International Conference on Neural Networks (Path, Australia)*, IEEE Service Center, Piscataway, NJ, Vol. 4, 1942-1948, 1995.

[11] J. Kennedy and R. Eberhart, Swarm Intelligence, San Mateo, CA: Morgan Kaufmann, 2001.

[12] Y. Shi, and R. Eberhart, "A modified particle swarm optimizer", *Proceedings of IEEE International Conference on Evolutionary Computation, IEEE World Congress on Computational Intelligence*, 69-73, 1998.