Neural Comput & Applic (2016) 27:349–374
DOI 10.1007/s00521-015-1849-4

ORIGINAL ARTICLE

Intrusive tumor growth inspired optimization algorithm for data clustering

Deyu Tang · Shoubin Dong · Lifang He · Yi Jiang

Received: 3 August 2014 / Accepted: 11 February 2015 / Published online: 21 March 2015

Abstract Inspired by the invasive tumor growth mechanism, this paper proposes a new meta-heuristic algorithm. A population of tumor cells can be divided into three subpopulations as proliferative cells, quiescent cells, and dying cells according to the nutrient concentration they get. Different cells have different behaviors and interactions among them for competition. In the tumor growing process, an invasive cell is born around a proliferative cell for the higher nutrient concentration and a necrotic cell occurs around a dying cell for the lower nutrient concentration, which presents the balance between life and death. To evaluate the performance of the intrusive tumor growth optimization algorithm (ITGO), we compared it to the many well-known heuristic algorithms by the Wilcoxon's signed-rank test with Bonferroni–Holm correction method and the Friedman's test. At the end, it is applied to solve the data clustering problem, which is a NP-hard problem. The experimental results show that the proposed ITGO algorithm outperforms other traditional heuristic algorithms for several benchmark datasets.

D. Tang · S. Dong (✉) · L. He
School of Computer Science and Engineering, South China University of Technology, Guangzhou 510006, People's Republic of China
e-mail: sbdong@scut.edu.cn

D. Tang
e-mail: scutdy@126.com

D. Tang
School of Medical Information and Engineering, Guangdong Pharmaceutical University, Guangzhou 510006, People's Republic of China

Y. Jiang
Department of Mathematics and Statistics, Georgia State University, Atlanta, GA, USA

Keywords Meta-heuristic algorithm · Intrusive tumor growth optimization · Data clustering

1 Introduction

Nature-inspired meta-heuristic algorithms are powerful and effective in solving optimization problems [1, 2], which have now been used in many fields such as computer science [3], data mining [4], medicine and biology [5], economy [6], and engineering [7]. Over the last few decades, many nature-inspired meta-heuristic algorithms have emerged. For instance, genetic algorithm (GA) [8] which works on the principle of the Darwinian theory of the survival of the fittest by genetic mutation, crossover, and selection operators; differential evolution (DE) [9] which is similar to GA with specialized crossover and selection method; ant colony optimization (ACO) [10] which was inspired by the foraging behavior of the ant for the food; particle swarm optimization (PSO) [11, 12] which was inspired by the foraging behavior of the birds; teaching learning-based optimization (TLBO) which was inspired by the effect of the influence of a teacher on the output of learners in a class [13]; cuckoo search algorithm (CSA) [14] which was inspired by the cuckoo brood parasitism behavior; gravitational search algorithm (GSA) [15], the searcher agents are a collection of masses which interact with each other based on the Newtonian gravity and the laws of motion. Black hole approach (BH), which was inspired by the BH phenomenon [16]; Krill Herd algorithm (KH) [17], which was inspired by the herding behavior of krill individuals; artificial bee colony (ABC) algorithm, which simulates the foraging behavior of bees for multimodal and multi-dimensional numerical optimization problems [18]; biogeography-based optimization (BBO)

was initially developed in [19], which can be seen as an evolutionary algorithm (EA) motivated by the optimality perspective of natural biogeography; artificial cooperative search algorithm (ACS) [20], which was inspired of mutualism and cooperation-based biological interaction of two eusocial superorganisms living in the same habitat; Grey wolf optimizer (GWO) [21], which mimics the leadership hierarchy and hunting mechanism of gray wolves in nature; and invasive weed optimization (IWO) [22], which was inspired from colonizing weeds.

This paper presents a new optimization method and its application to data clustering, which is inspired by the invasive tumor growth mechanism. A tumor is a cluster of abnormal cells in body. The deep investigation and study show that tumor is a complex system. Tumor cells can be divided into the main three types: proliferative cells, quiescent cells, and dying cells, which grow for the nutrient (oxygen and glucose etc.). The adequate nutrient makes the invasive cell born around the proliferative cell, while the lower nutrient make the dying cell convert into necrotic cell. Different cells run with different behaviors by cooperation. A proliferative cell grows with a levy flight behavior to obtain more nutrients, and a quiescent cell grows by the leading of a proliferative cell and the interaction among themselves. For dying cell, they can obtain more nutrients by the leading of a proliferative cell and a quiescent cell. The more interactions and behaviors ensure that our proposed algorithm can easily jump out of the local optimum.

The rest of the paper is organized as follows: In Sect. 2, the clustering problem is discussed. A brief explanation of the tumor growth modeling is given in Sect. 3. In Sect. 4, we introduce our proposed invasive tumor growth model and algorithm. The performance of the proposed algorithm is tested with several benchmark functions such as CEC 2005 and CEC 2008 and applied to the data clustering for the benchmark datasets, which are compared with many well-known algorithms in Sect. 5. Parameter tuning is discussed in Sect. 6. Section 7, we discuss the proposed algorithm (ITGO); finally, Sect. 8 includes a summary and the conclusion of this work.

2 Cluster analysis

Data clustering is an important and popular method in data mining techniques, which has been widely used in many fields such as pattern recognition, document clustering, image processing, marketing, costumer analysis, and biology. Clustering can be seen as a classification process without supervision, in which, a set of data objects is grouped into some clusters. Especially, the data of a cluster must have the great similarity, and the data of different clusters must have high dissimilarity.

Strictly speaking, clustering problem is to determine a partition $G = \{C_1, C_2, \ldots, C_k\}$, $\forall k\ C_k \neq \varnothing$ and $\forall h \neq k, C_h \cap C_k = \varnothing$ such that objects belong to the same cluster are as similar as possible, while objects belong to the different clusters are as dissimilar as possible. One object can only belong to one cluster. To achieve this, a mathematical model can be described as follows [23]:

$$\text{Min} \sum_{i=1}^{n}\sum_{j=1}^{k} w_{ij}||X_i - C_j|| \tag{1}$$

$$0 < \sum_{i=1}^{n} w_{ij} < n, \quad j = 1, 2, \ldots, k \tag{2}$$

$$\sum_{j=1}^{k} w_{ij} = 1, \quad i = 1, 2, \ldots, n \tag{3}$$

$$w_{ij} \in \{0, 1\} \tag{4}$$

$$w_{ij} \in [0, 1] \tag{5}$$

where n is the number of objects and k is the number of clusters, X_i denotes the object i, C_j denotes the cluster center j. $||X_i - C_j||$ is the euclidean distance between the object X_i, and the cluster center C_j. w_{ij} denotes the degree of membership. Using formula (1), (2), (3), and (4), the model was called 'hard partition,' while using formula (1), (2), (3) and (5), it can be called 'soft partition,' or called as fuzzy clustering. It was proven that the clustering problem is NP problem [24].

There are many clustering algorithms in the literature. The classical clustering algorithms are hierarchical algorithm and partitional algorithm [25, 26]. Among all these algorithms, K-means algorithm was the most well known because of its simplicity and efficiency. However, it suffers from some problems. For instance, the number of the initial cluster centers needs to be defined, and the initial cluster center are chosen randomly so that the algorithm is easy to fall into the local optimal solution. To overcome the weakness of the K-means, many heuristic methods have been used, such as DE algorithm [27], GSA [28, 29], PSO algorithm [30], firefly algorithm [31], ACO [32], big bang–big crunch algorithm [33], honey bee mating optimization [34], and TLBO [35].

3 Modeling of invasive tumor growth

Tumor is a cluster of abnormal cells which grows around some tissues. It is a generally accepted view that genome level changes in cells turn a normal cell into a tumor cell. Usually, the cell growth is controlled by the deactivation of growth pathway and activation of programmed cell death pathway in a normal tissue. When both of these regulatory

mechanisms fail with an overproduction of growth factors and inhibition of programmed cell death, tumor occurs [36, 37]. From the point of view of the system science, the tumor can be considered as a complex linear dynamic system. The tumor cells proliferate and invade into surrounding tissue as seen in Fig. 1. The main experimental observations show that tumor cells proliferate when they are in a high nutrient (oxygen and glucose etc.) level, the tumor cells trigger cell death (apoptosis) in low nutrient levels, and in intermediate nutrient levels, the tumor cells stay quiescent [38]. In order to apprehend the mechanism of tumor growth, many mathematical models have been established [38–41]. All these models can be divided into two categories: (1) continuum mathematical models that use space averaging and thus consist of partial differential equations and (2) discrete cell population models that consider processes that occur on the single-cell scale and introduce cell–cell interaction using cellular automata-type computational machinery. For many continuum mathematical models, heat conduction equation was used to present the diffusion of nutrient concentration for the tumor cells. The equation [39] is as follows:

$$\frac{\partial n}{\partial t} = D_n \nabla^2 n + \sigma e - \varphi c - \lambda n \tag{6}$$

where n, D_n, σ, φ, and λ are the nutrient concentration, diffusion coefficient, production, consumption, and natural decay rates of the nutrient, respectively. To survive and grow, a tumor cell consumes nutrients at a certain rate, and if the nutrient concentration around it in the microenvironment is lower than the specified value, it will become a quiescent cell or dying cell or apoptosis (dead). Moreover, for the discrete cell population models, in order to simulate the process of the tumor growth, they consider the influence of the interactions between tumor cells and their microenvironment, including their surrounding cells, as well as the extracellular matrix (ECM), chemical signals, metabolic substrates such as oxygen and glucose. More of these models can really simulate the process of the tumor growth including the space and time, which makes it more easier to understand the mechanism of the tumor growth.

Summarized from the models of tumor growth [39–41], tumor cells have the following characteristics:

1. Tumor can be regarded as a isotropic sphere, and a concentric spatial layered structure consists of an outer rim of proliferative cells, an intermediate layer of viable, but dormant, quiescent cells, and an inner necrotic core.
2. In order to survive and grow, tumor cells must get enough nutrients, and the tumor cells have the invasive ability.
3. Tumor cells will move toward the direction of higher nutrient concentrations.
4. There is a complex interaction between tumor cells.
5. They have a random walk of motivation.
6. When nutrient concentration is lower than a certain value, the proliferative cells may turn into quiescent cells; while nutrient concentration is lower than the minimum value, the quiescent cells turn into dying cells, or even apoptosis.
7. Nutrient concentration corresponds to different types of tumor cells.

4 Intrusive tumor growth optimization algorithm (ITGO)

4.1 Modeling of Intrusive tumor growth

As mentioned above, two kind of mathematical models were used to simulate the tumor growth: continuum

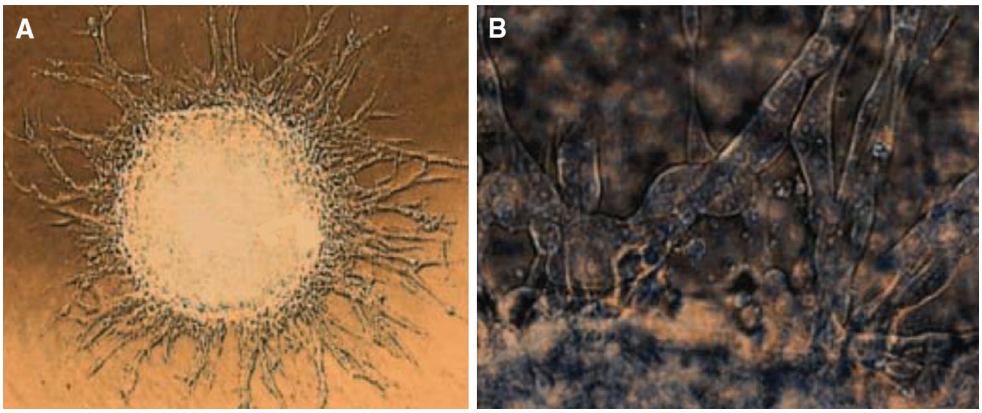

Fig. 1 GBM multi-cellular tumor spheroid (MTS) gel assay showing dendritic invasive branches. **a** The invasive branches centrifugal evolve from the central MTS. The linear size of central MTS is approximately 400 mm. **b** The invasive branches are composed of chains of invasive cells. The images are adapted from Ref. [40]

mathematical models and discrete cell population models. Continuum mathematical models present the diffusion of nutrient concentration for the tumor, and discrete cell population models present the interactions of cells in tumor. However, these models were used only to simulate the processes of tumor growth. How to create a new model to solve the optimization problem should be studied deeply. According to the principle of the Darwinian theory of the survival of the fittest, tumor cells must compete for the nutrient to be survival. If a tumor cell cannot get adequate nutrients, it may be dead. Different tumor cells get the different nutrient by competition, which influences their growing ways. As we all know, tumor is not an individual but a complex system, so competition and interaction for all the tumor cells must be needed. Then a model of intrusive tumor growth optimization can be established. Different from the previous mathematical models, to implement the evolution process of tumor growth for solving optimization problems, five kinds of cells were considered: invasive cell, proliferative cell, quiescent cell, dying cell, and necrotic cell, depending on the nutrient supply they get.

Figure 2 shows the tumor cell's competition and interaction. In our model, invasive cells and proliferative cells locate at the outer layer of the tumor, where they can get best nutrients and grow fast, and invasive cells are the daughters of proliferative cells by mutation; quiescent cells locate at the middle layer of the tumor, and they can get general nutrients and grow relatively slow. For competition, they must obtain enough nutrients by leading of proliferative cells; dying cells are located at the inner layer of the tumor, due to the least nutrient concentration, they can grow slowly or may develop into necrotic cells (dead cells). For competition, they obtain nutrient by leading of proliferative cell and quiescent cells; necrotic cells locate at the core of the tumor; they are dead, and they cannot grow or move. Figure 3 shows the diffusion of nutrient, and different colors from dark red to light pink present the diffusion of nutrient from the outer layer to the inner layer of tumor. Different nutrient concentration I, P, Q, D, and N corresponds to different kind of cells: Invasive cell (*Icell*), proliferative cell (*Pcell*), Quiescent cell (*Qcell*), Dying cell (*Dcell*), and Necrotic cell (*Ncell*). In addition, necrotic cells have been dead, so they do not move and there are no interactions for them.

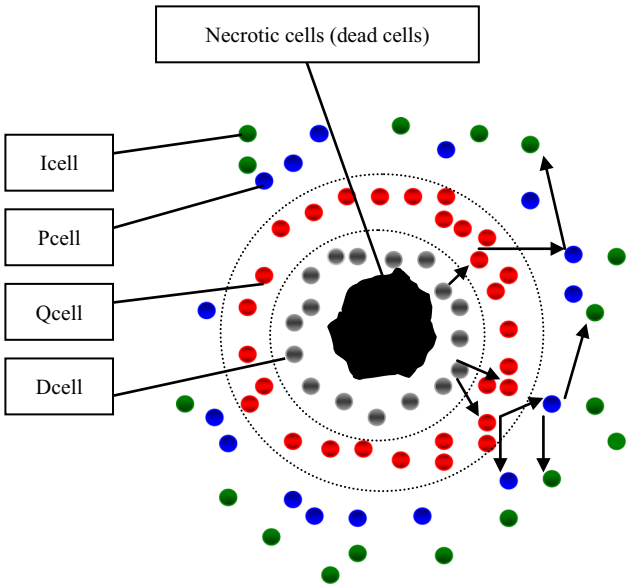

Fig. 2 The optimization model of invasive tumor growth

In our model, tumor growth is a complex system using the interaction among proliferative cells, quiescent cells, dying cells, and invasive cells, which compete for survival. Different cells have different behaviors.

1. The growth of proliferative cells

The proliferative cell locates at the outer layer of the tumor with the highest nutrient concentration, so it is a 'leader role.' For the other kind of cells, all the proliferative cells are leaders. It is to say, tumor has many leaders, which grow by the stronger invasive behaviors (seen as Fig. 1). A remarkable characteristic of proliferative cells is that they move short distance, occasionally moves long distance. To achieve this goal, we use the Levy distribution. Formula is as following:

$$Pcell_{i,j}(t+1) = Pcell_{i,j}(t) + \alpha \cdot Levy(s) \tag{7}$$

$Pcell_{i,j}$ presents the position of proliferative cell, $Levy(s)$ presents the intrusive behavior of the proliferative cell, and α is the control size of step.

$$\alpha = rand \cdot \left(\frac{t}{T}\right) \tag{8}$$

where t is the current iterative number, and T is the total iterative number.

Levy distribution usually appears in a simplified form, as follows:

$$Levy(s) \sim |s|^{-1-\upsilon}, \quad (1 < \upsilon \leq 2) \tag{9}$$

Broadly speaking, Levy flight is a random walk, and the step size obeys Levy distributions and walk direction obeys uniform distribution. To perform the Levy distributions, we use the Levy distribution characterized with Mantegna law [42, 43] to select the step length vector. In Mantegna law, the step size is designed as follows:

$$step = \frac{u}{|v|^{1/\omega}} \tag{10}$$

where u, v obeys normal distribution, i.e.,

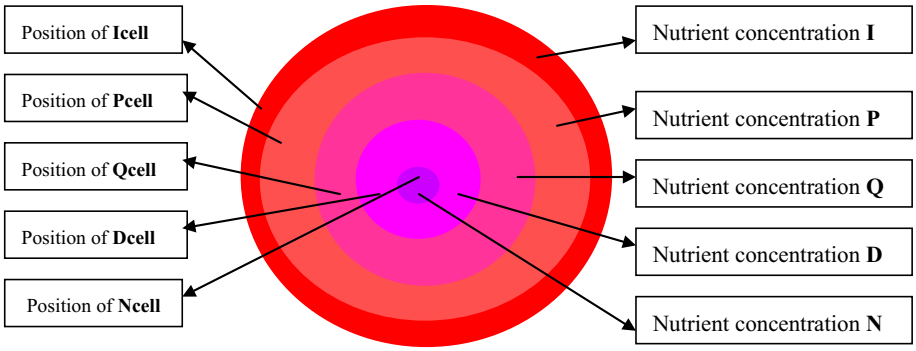

Fig. 3 Diffusion of Nutrient concentration

$$u \sim N(0, \sigma_u^2), \quad v \sim N(0, \sigma_v^2) \tag{11}$$

$$\sigma_v = \left\{ \frac{\Gamma(1+\omega)\sin(\pi\omega/2)}{\Gamma[(1+\omega)/2]\omega 2^{(\omega-1)/2}} \right\}^{1/\omega} \tag{12}$$

ω is a constant.

In the search process of the proliferative cell, $Levy(s)$ presents the search in different regions, and control size of step α achieves the balance of exploration and exploitation. In the early search stage, when $\frac{t}{T} \leq 0.5$, α obtains a more small value, each proliferative cell visits those regions of a search space within the neighborhood of previously visited points, and it can be seen as an exploitation, while in the late search stage, when $\frac{t}{T} > 0.5$, α perhaps obtains a more large value, each proliferative cell visits entirely new regions of a search space, and it can be seen as an exploration.

2. The growth of quiescent cells

The quiescent cells are the general but most members of the tumor, which locate at the middle layer. Due to the low nutrient they get, they grow relatively slow. For competition, the quiescent cells move toward the higher nutrient concentration, guided by proliferative cells, and they interact with each others.

Here, $hPcell_{p,j}(t)$ is the historical best position of proliferative cell, and $cPcell_{p,j}(t)$ is the current position of proliferative cell. $Qcell_{i,j}(t)$ is the old position of quiescent cell, and $sQcell_{i,j}(t+1)$ is the current position of quiescent cell. The step presents the step size of Levy flight such as formula (10); β presents the growing speed; p is an integer, presents a proliferative cell selected randomly from the group of proliferative cell; x and y are integers, present two quiescent cells selected randomly from the group of quiescent cells, and $x \neq y$, x and y are two quiescent cells near the i quiescent cell. Formula (15) presents the mutation of quiescent cell.

In the search process of quiescent cells, we consider the balance of exploration and exploitation, too. Though quiescent cell population is the main body of the tumor, many cells should search including exploration and exploitation. In formula (13), we can see that, each quiescent cell is guided by the historical best position of proliferative cell $hPcell_{p,j}(t)$ and the current position of proliferative cell $cPcell_{p,j}(t)$ alternately, which means that each quiescent cell search in a large region to find a best position, it can be seen as an exploration; while the two neighbor's operation ensures the quiescent cell visits in the neighbor regions, it can be seen as an exploitation.

$$sQcell_{i,j}(t+1) = \begin{cases} Qcell_{i,j}(t) + \beta \cdot (hPcell_{p,j}(t) - Qcell_{i,j}(t)) + \beta \cdot ((Qcell_{x,j}(t) - Qcell_{x,j}(t))), & rand < 0.5 \\ Qcell_{i,j}(t) + \beta \cdot (cPcell_{p,j}(t) - Qcell_{i,j}(t)) + \beta \cdot ((Qcell_{x,j}(t) - Qcell_{y,j}(t))), & rand > 0.5 \end{cases} \tag{13}$$

$$\beta = rand(0,1) \cdot normal(0,1) \cdot step \tag{14}$$

$$Qcell_{i,j}(t+1) = \begin{cases} Qcell_{i,j}(t), & rand < e^{\left(\frac{t}{T}-1\right)} \\ sQcell_{i,j}(t+1), & else \end{cases} \tag{15}$$

3. The growth of dying cells

The dying cells locate at the inner layer of the tumor with the least nutrient, so it is easy to be converted into necrotic cell (dead). Luckily, other cells of them can grow for competition. In dying cell region, nutrient concentration is very low, so these cells move toward the direction of

quiescent cells and proliferative cells with a higher nutrient concentration. Formula is as following:

$$Dcell_{i,j}(t+1) = Dcell_{i,j}(t) + \gamma \cdot (hPcell_{p,j}(t) - Dcell_{i,j}(t)) + \gamma \cdot ((Qcell_{x,j}(t) - Dcell_{i,j}(t))) \quad (16)$$

$Dcell_{i,j}(t)$ presents the position of the old dying cell, $Dcell_{i,j}(t+1)$ presents the position of the current dying cell, $Qcell_{i,j}(t)$ presents the position of quiescent cell, $hPcell_{p,j}(t)$ presents the position of proliferative cell (the historical best position); p is an integer that presents a proliferative cell selected randomly from the group of proliferative cells; x is an integer that presents a quiescent cell selected randomly from the group of quiescent cells. When $\gamma = rand[-1, 1]$., Formula (16) shows that $hPcell_{p,j}(t)$ and $Qcell_{x,j}(t)$ guide $Dcell_{i,j}(t)$ to move for growth.

In the search process of the dying cells, formula (16) means that each dying cell is guided by the proliferative cell and the quiescent cell, and it can be seen as an exploration. Since each dying cell visits a new regions according the position of the proliferative cell and the quiescent cell, two leaders (proliferative cell and the quiescent cell) were chosen in different populations randomly, which can enhance the diversity of our algorithm.

4. The growth of invasive cells

As mentioned above, invasive cells locate at the outer layer of the tumor. They are the most active ones, born by the proliferative cells. When an invasive cell is born, a dying cell with the least nutrient will be dead. The balance of life and death will always be the evolution of the tumor.

Formula (17) presents a new cell which is born randomly in the tumor, and formula (18) shows that a mutant daughter of a historical best proliferative cell is born, which is an invasive cell $Icell_{i,j}(t+1)$. Formula (19) presents that a dying cell is apoptosis and replaced by the invasive cell $Icell_{i,j}(t+1)$.

$$newCell_{i,j}(t) = Random(i, D), D \in [D\ \min, D\ \max] \quad (17)$$

$$Icell_{i,j}(t+1) = hPcell_{p,j}(t) + \eta \cdot (newCell_{i,j}(t) - hPcell_{p,j}(t)) \quad (18)$$

$$Dcell_{i,j}(t+1) = Icell_{i,j}(t+1) \quad (19)$$

Here, $\eta \in rand[0, 1]$ presents the growing speed.

In the search process of invasive cells, an exploration is achieved. Formula (17), (18) means that an invasive cell visits a more larger search region nearby the proliferative cell. Especially, for the multimodal problems, in the late search process, many proliferative cells maybe obtained from different local optimum. So invasive cell ensures that they can jump out the local optimum and find the best position; it is to say, they can enhance the exploration ability of our algorithm. In addition, formula (19) means that some bad solution (dying cell) can be replaced by the best solution (invasive cell).

5. Random walk of cells

Each cell in tumor has a cycle of life. When a cell is not affected by other chemotaxis within the specified cycle of life, the cell performs a random walk as formula (20). Here, $newCell_{i,j}(t)$ presents a new cell using formula (17).

$$Cell(t+1)_{i,j}(t+1) = Cell_{i,j}(t) + \lambda \cdot \left(\frac{newCell_{i,j}(t)}{||newCell_{i,1:D}(t)||} \right) \quad (20)$$

where $||newCell_{i,j:D}(t)||$ is an Euclidean norm. $\lambda \in rand[-1, 1]$.

At here, the random walk can be seen as an exploitation. $\frac{newCell_{i,j}(t)}{||newCell_{i,1:D}(t)||}$ represents a small value in the search region. So the random walk can be seen as a small fluctuation for all the tumor cells. It is to say that each tumor cell can visit its neighbor region.

4.2 Intrusive tumor growth algorithms

We implement an invasive tumor growth optimization algorithm according to the mechanism of tumor growth. There are the main five steps in our algorithm: proliferative cell growth, quiescent cell growth, dying cell growth, invasive cell growth and the random walk of all the tumor cells. Details shows as Fig. 4. In Fig. 4, step 1–2 implements the initialization like other meta-heuristic algorithms. Step 3–4 creates the three subpopulations: proliferative cell subpopulation, quiescent cell subpopulation, and dying cell subpopulation. Firstly, proliferative cell growth is implemented as the step 5. For $rand < \frac{t}{T}$, t is the number of the current iteration, and T is the number of total iteration. It means that proliferative cell grows using a higher probability with a liner function. This easily helps proliferative cell to jump out of the local optimum. Formula (7)–(12) enhances the searching ability of proliferative cell by Levy flight. Secondly, quiescent cell growth is implemented as the step 7. Many 'leaders' (proliferative cells) are chosen randomly to guide the quiescent cell for the best search direction. Historical position and current position of proliferative cell are used alternately, which enhances the diversity of the algorithm. Third, dying cell growth is implemented as step 9. The proliferative cell and the quiescent cell are chosen randomly to lead the dying cell for the better search. Fourth, invasive cell growth is implemented as step 11. It achieves the balance of the life and death of tumor. The old solution (position of the dying cell) can be replaced by a better solution (the position of the invasive cell), which can enhance the search speed of

Fig. 4 Invasive tumor growth algorithm

1. parameters setting: population size, iteration number etc.
2.initialization: create the initial tumor cells population.
3. Sort tumor cells according to the fitness (nutrient concentration).
4. Create three subpopulations according to the fitness: Proliferative Cells (20%), Quiescent Cells (60%), Dying Cells (20%).
5. for all the proliferative cells:
   if $rand < \frac{t}{T}$
   update position of proliferative cells using formula (7)~(12).
   End
6. If the fitness of new position is not better than the old one for θ times(θ is threshold value)
   update position of proliferative cells using formula (20) .
   end
7.for all the quiescent cells:
   update position of quiescent cells using formula (13)~(15).
8. If the fitness of new position is not better than the old one for θ times(θ is threshold value)
   update position of quiescent cells using formula (20) .
   end
9. for all the dying cells:
   update position of dying cells using formula (16) .
10. If the fitness of new position is not better than the old one for θ times(θ is threshold value)
   update position of dying cells using formula (20) .
   end
11. for all the dying cells :
   If nutrient concentration is lower than the average nutrient concentration in dying cells
   update position of dying cells using fomula (17)(18)(19) .
   end
12. If meeting end of criterion, output the best solution, else go to the step 3.

the algorithm. Fifth, random walk is used for all the tumor cells as step 6, 8, 10. θ is a threshold value, generally, it is an integer. It controls the random walk frequency. The random walk method can help the algorithm jump out of the local optimum.

5 Experiments analysis

In order to solve the data clustering problem, object function shown as formula (1) was used as fitness function of meta-heuristic algorithms. The data clustering problem can be seen to find the best centers for all the clusters. For our algorithm, the position of each tumor cell (solution) is encoded by a link of the centers of all the clusters.

Our experiments are divided into two categories: (1) We choose the benchmark functions for comparison with other algorithms to verify the performance of our proposed algorithm. (2) To verify the performance of our algorithm, we use six benchmark datasets to compare our algorithm with other algorithms for data clustering.

5.1 Comparison of CEC 2005 and CEC 2008

We choose the CEC 2005 [44] and CEC 2008 [45] dataset to verify the performance of the proposed algorithm (ITGO), which are compared with the well-known algorithm such as PSO, DE, GSA, CLPSO, cPSO, rcGA, and TLBO. We choose 15 functions from CEC 2005 with low dimension and five functions from CEC 2008 with high dimension. F1–F4, F16, and F17 are unimodal functions and F5–F15, F18, F19, and F20 are multimodal functions. These functions are shifted or rotated or separable or non-separable or both. So, the 20 functions we choose can reflect the reality problem better. Details are shown in Table 1. Figure 5 shows F11 and F14 function, which are multimodal.

For purposes of avoiding any negative effects of the structure of the initial population during the tests, the benchmark problems are solved 20 times by using a

Table 1 Details of CEC2005 and CEC2008

Fnc	Benchmark problem	Type low	Low	Up	Dim
1	Shifted sphere	U	−100	100	10
2	Shifted Schwefel's problem 1.2	U	−100	100	10
3	Shifted Schwefel's problem 1.2 with noise	U	−100	100	10
4	Schwefel's problem 2.6	U	−100	100	10
5	Shifted rotated Griewank's	M	0	600	10
6	Shifted rotated Ackley's	M	−32	32	10
7	Shifted Rastrigin's	M	−5	5	10
8	Shifted rotated Weierstrass	M	−0.5	0.5	10
9	Schwefel's problem 2.13	M	−100	100	10
10	Expanded rotated extended Scaffes	M	−100	100	10
11	Rotated hybrid comp. Fn 1	M	−5	5	10
12	Rotated hybrid comp. Fn 1 with noise	M	−5	5	10
13	Rotated hybrid comp. Fn 3 with high condition number matrix	M	−5	5	10
14	Rotated hybrid comp. Fn 4	M	−5	5	10
15	Rotated hybrid comp. Fn 4 without bounds	M	−5	5	10
16	Shifted sphere function	U	−100	100	100
17	Shifted Schwefel's problem 2.21	U	−100	100	100
18	Shifted Rosenbrock's function	M	−100	100	100
19	Shifted Griewank's function	M	−600	600	100
20	Shifted Ackley's function	M	−32	32	100

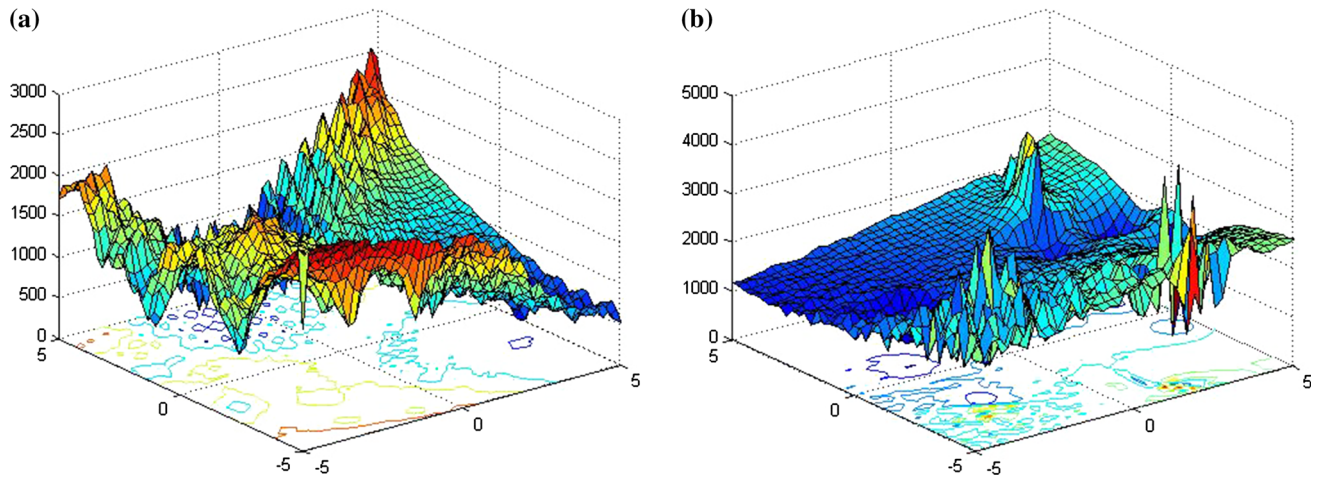

Fig. 5 Benchmark problems. **a** F11 rotated hybrid comp. Fn 1, **b** F14 rotated hybrid comp. Fn 4

different initial population each time. Population size is 30 for all the algorithms. The data such as the mean, standard deviation obtained as a result of the tests have been saved for detailed statistical analysis.

For these algorithms, parameter settings are as followings:

1. PSO, $c1 = c2 = 2$ and inertia factor (w) is decreasing linearly from 0.9 to 0.2 [46].
2. DE, $F = 0.5$, CR = 0.5 [9].
3. GSA, Rpower = 2, Rnorm = 2, ElitistCheck = 1 as in [15].
4. CLPSO [47], $c = 1.49335$, $m = 0$, $p_c = 0.5 \cdot (e^t - e^{t(1)})/(e^{t(p_s)} - e^{t(1)}), t = 0.5 \cdot \left(0 : \frac{1}{p_s - 1} : 1\right)$ as in the Matlab code given in [48].
5. cPSO, $Np = 300$, $\varphi_1 = -0.2$, $\varphi_2 = -0.007$, $\varphi_3 = 3.74$, $\gamma_1 = 1$, $\gamma_2 = 1$ as in [49].
6. rcGA, $Np = 300$ [50].
7. TLBO with no parameter [13]. Matlab Code provided by author in [51].

Table 2 Comparison with ITGO and PSO, DE, and GSA

	PSO	DE	GSA	ITGO
F1 SD	5.7387e+001	**7.6855e−009**	1.7665e+004	8.5149e−009
Means	8.7142e+001	1.8512e−009	5.1465e+003	**9.6323e−010**
F2 SD	5.0758e+001	1.6192e+001	2.2892e+004	**7.5104e−006**
Means	4.9926e+001	4.0329e+001	5.7967e+003	**8.9117e−006**
F3 SD	7.6559e+001	1.3213e+001	2.2919e+004	**1.1999e−003**
Means	8.9703e+001	4.0938e+001	6.0041e+003	**2.5778e−003**
F4 SD	2.7513e+004	**7.2226e−009**	2.0163e+004	4.9194e−004
Means	6.6915e+003	**1.8458e−009**	2.4464e+003	1.6068e−003
F5 SD	3.6044e+003	1.2671e+003	3.6126e+003	**1.2670e+003**
Means	4.6759e+002	4.0256e−002	3.4695e+002	**2.2737e−013**
F6 SD	2.0352e+001	2.0345e+001	**2.0038e+001**	2.0261e+001
Means	9.0376e−002	6.8602e−002	**2.4513e−002**	5.9821e−002
F7 SD	2.0866e+001	**7.6897e−009**	4.6763e+000	1.0655e−001
Means	6.8771e+000	**1.8368e−009**	1.5847e+000	3.0546e−001
F8 SD	4.1819e+000	7.4047e+000	**5.4331e−002**	2.9985e+000
Means	1.1166e+000	6.9087e−001	**8.0375e−003**	1.1665e+000
F9 SD	2.7663e+004	1.2983e+003	9.0255e+004	**1.4029e+002**
Means	6.6915e+003	7.5726e+002	3.0778e+004	**1.5898e+002**
F10 SD	3.1728e+000	3.5887e+000	4.3143e+000	**2.7712e+000**
Means	3.3975e−001	**1.3351e−001**	1.4205e−001	2.8920e−001
F11 SD	1.5236e+002	1.5150e+002	**8.9496e+001**	1.1391e+002
Means	1.7445e+001	2.3391e+001	2.7819e+001	**1.1448e+001**
F12 SD	1.7295e+002	1.7982e+002	**9.2977e+001**	1.2022e+002
Means	2.2291e+001	3.2664e+001	2.6041e+001	**1.5719e+001**
F13 SD	8.4998e+002	7.9262e+002	7.2876e+002	**6.9863e+002**
Means	6.4274e+001	**3.7308e+001**	5.8108e+001	1.7374e+002
F14 SD	7.7302e+002	8.0320e+002	3.0000e+002	**200e+000**
Means	2.2678e+002	3.0946e+002	2.2478e+002	**0**
F15 SD	1.1772e+003	8.1692e+002	1.3139e+003	**8.1497e+002**
Means	1.6188e+002	2.5439e+000	1.1608e+001	2.7029e+000
F16 SD	3.0009e+004	1.3168e+002	4.7878e+005	**9.8599e−009**
Means	1.0092e+004	1.3508e+002	3.5312e+004	**1.1644e−010**
F17 SD	3.9188e+001	6.2165e+001	1.5073e+002	**2.2062e+001**
Means	1.6924e+001	2.1631e+001	5.4376e+000	**3.9608e+000**
F18 SD	1.3449e+009	3.1709e+004	3.7126e+011	**5.4122e+002**
Means	1.0628e+009	3.3904e+004	3.8252e+010	**1.4815e+001**
F19 SD	2.3698e+002	1.8279e+000	4.6133e+003	**8.6268e−004**
Means	8.7682e+001	7.6548e−001	2.7035e+002	**2.6851e−003**
F20 SD	1.1412e+001	1.3318e+000	1.9180e+001	**5.1900e−006**
Means	7.0655e−001	6.2024e−001	6.9656e−002	**5.8889e−006**

8. ITGO, $c = 8$, $d = 8$, $beta = 1.5$. C is the cycle size of the cell; d is the number of neighbors in quiescent cells; *beta* is the parameter of Levy flight.

The tests of all algorithms are rooted on a uniform standard:

1. For CEC2005, dimension is 10, population size is 30, MAX_FES (total fitness evaluations number) = 1e5 [44]
2. For CEC2008, dimension is 100, population size is 30, MAX_FES = 5e5 [45]

Table 3 Comparison with ITGO and CLPSO, cPSO, QIGSA, rcGA, and TLBO

	CLPSO [58]	CPSO [60]	RcGA [61]	TLBO [61]	ITGO
F1 SD	1.6995e−007	9.7969e−002	**4.4054e−013**	7.8161e−009	8.5149e−009
Means	2.6648e−007	6.5420e−002	**7.5760e−014**	1.3371e−009	9.6323e−010
F2 SD	1.0566e+002	3.3007e+000	1.5015e−001	**8.2962e−009**	7.5104e−006
Means	1.3176e+002	1.7627e+000	5.8963e−001	**1.1618e−009**	8.9117e−006
F3 SD	1.0998e+000	4.1923e+001	8.7665e+001	**8.4358e−009**	1.1999e−003
Means	1.4692e+000	5.2061e+001	1.1250e+002	**1.0647e−009**	2.5778e−003
F4 SD	6.2789e+000	1.1225e+002	1.2712e−004	**8.2777e−009**	4.9194e−004
Means	1.0314e+001	4.6718e+001	4.4181e−005	**1.2328e−009**	1.6068e−003
F5 SD	1.2670e+003	1.2707e+003	1.2670e+003	1.2671e+003	1.2670e+003
Means	2.3328e−013	5.8400e+000	1.8308e−005	4.0256e−002	**2.2737e−013**
F6 SD	2.0682e+001	2.0321e+001	**2.0160e+001**	2.0335e+001	2.0261e+001
Means	3.1638e−001	**5.7115e−002**	9.4724e−002	8.6351e−002	5.9821e−002
F7 SD	4.4064e+001	1.0992e+001	1.9551e+001	8.5946e+000	**1.0655e−001**
Means	4.5880e+001	6.0154e+000	1.1244e+001	3.3635e+000	**3.0546e−001**
F8 SD	4.8000e+000	4.8252e+000	5.4259e+000	5.4080e+000	**2.9985e+000**
Means	**6.6154e−001**	1.1053e+000	2.4877e+000	1.3843e+000	1.1665e+000
F9 SD	1.4656e+003	2.0232e+004	7.4280e+002	2.2109e+004	**1.4029e+002**
Means	6.4697e+002	6.9820e+003	8.6740e+002	7.2890e+003	**1.5898e+002**
F10 SD	3.1218e+000	3.2195e+000	3.5708e+000	**2.7111e+000**	2.7712e+000
Means	2.5833e−001	**2.5072e−001**	3.1913e−001	4.4291e−001	2.8920e−001
F11 SD	**1.1078e+002**	1.6305e+002	1.4775e+002	1.2072e+002	1.1391e+002
Means	1.6331e+001	3.0796e+001	2.9385e+001	2.1296e+001	**1.1448e+001**
F12 SD	1.3077e+002	1.8138e+002	1.7112e+002	1.2075e+002	**1.2022e+002**
Means	1.1988e+001	4.1862e+001	3.2693e+001	**1.1176e+001**	1.5719e+001
F13 SD	7.5627e+002	7.6540e+002	8.2237e+002	7.7748e+002	**6.9863e+002**
Means	7.1650e+001	1.1261e+002	5.7475e+001	**3.4844e+001**	1.7374e+002
F14 SD	2.0000e+002	2.8041e+002	2.6500e+002	2.6000e+002	**200 e+000**
Means	5.7123e−006	2.0700e+002	1.7554e+002	1.2312e+002	**0**
F15 SD	8.1777e+002	8.8992e+002	**9.0791e+001**	8.7119e+002	8.1497e+002
Means	4.3980e+001	2.7205e+001	5.6032e+002	1.5822e+002	**2.7029e+000**
F16 SD	8.0013e+000	3.9197e+003	1.6545e+003	5.4437e−002	**9.8599e−009**
Means	1.8695e+000	8.3951e+002	2.1456e+003	2.4189e−001	**1.1644e−010**
F17 SD	8.5174e+001	7.3141e+001	1.2189e+002	8.7203e+001	**2.2062e+001**
Means	4.0196e+000	4.5574e+000	**2.8556e+000**	4.1943e+000	3.9608e+000
F18 SD	5.9350e+004	4.5974e+005	2.0503e+008	1.1942e+005	**5.4122e+002**
Means	1.7576e+004	1.7270e+005	3.1487e+008	5.3208e+005	**1.4815e+001**
F19 SD	1.0218e+000	2.2173e+001	5.6988e+000	1.6295e+003	**8.6268e−004**
Means	4.4770e−002	4.7487e+000	1.9974e+000	1.9587e−001	**2.6851e−003**
F20 SD	4.2750e+000	1.9491e+001	1.9727e+001	2.1307e+001	**5.1900e−006**
Means	6.4858e−001	9.2799e−002	2.7665e−001	5.9884e−002	**5.8889e−006**

3. Pre-determined stopping criteria: If the absolute error obtained by the algorithm is smaller than err = 1e−8, then stop.

The Wilcoxon's signed-rank test with Bonferroni–Holm correction method and the Friedman's test [52] were used to compare ITGO with other algorithms. When using Wilcoxon's test in our experimental study, the first step is to compute the R+ and R− related to the comparisons between ITGO and the rest of algorithms, then p values can be computed. The null hypothesis H0 was used for the Wilcoxon's signed-rank tests for purposes of this paper. The statistical significance value used to test H0 hypothesis is $\tau = 0.05$. The Bonferroni–Holm procedure adjusts the value of τ in a step-down manner. Let $p_1, p_2, \ldots, p_{k-1}$ be the ordered p values (from smallest to largest), so that $p_1 \leq p_2 \leq, \ldots, \leq p_{k-1}$, and let $H_1, H_2, \ldots, H_{k-1}$ be the

Table 4 Comparison results by Wilcoxon's signed-rank test

	PSO versus ITGO	DE versus ITGO	GSA versus ITGO	CLPSO versus ITGO	CPSO versus ITGO	RcGA versus ITGO	TLBO versus ITGO
F1	+	−	+	+	+	−	−
F2	+	+	+	+	+	+	−
F3	+	+	+	+	+	+	−
F4	+	−	+	+	+	−	−
F5	+	+	+	=	+	+	+
F6	+	+	−	+	+	−	+
F7	+	−	+	+	+	+	+
F8	+	−	−	+	+	+	+
F9	+	+	+	+	+	+	+
F10	+	+	+	+	+	+	−
F11	+	+	−	−	+	+	+
F12	+	+	−	+	+	+	+
F13	+	+	+	+	+	+	+
F14	+	+	+	+	+	+	+
F15	+	+	+	+	+	−	+
F16	+	+	+	+	+	+	+
F17	+	+	+	+	+	+	+
F18	+	+	+	+	+	+	+
F19	+	+	+	+	+	+	+
F20	+	+	+	+	+	+	+
+/=/−	20/0/0	17/0/3	16/0/4	18/1/1	20/0/0	16/0/4	15/0/5

Table 5 Comparison with ITGO and other algorithms using Wilcoxon's signed-rank test and Bonferroni–Holm correction (means)

I	Algorithm	R+	R−	p Value	α/i	Hypothesis
7	ITGO versus PSO	0.0	210.0	8.85e−5	0.0071	REJECT
6	ITGO versus CPSO	0.0	210.0	8.85e−5	0.0083	REJECT
5	ITGO versus DE	7.0	203.0	2.53e−4	0.01	REJECT
4	ITGO versus CLPSO	8.00	182.0	4.63e−4	0.0125	REJECT
3	ITGO versus TLBO	16.0	1940	8.91e−4	0.0167	REJECT
2	ITGO versus GSA	17.0	1930	1.01e−3	0.025	REJECT
1	ITGO versus rcGA	24.0	186.0	2.49e−3	0.05	REJECT

Table 6 Average ranks of eight algorithms by Friedman test (means)

Algorithm	Average ranks
ITGO	1.88
TLBO	3.95
CLPSO	4.03
DE	4.30
rcGA	4.55
cPSO	5.4
GSA	5.75
PSO	6.15

Table 7 Results of Friedman test for eight algorithms by CEC 2005 and CEC 2008

Method	Statistical value	p Value
Friedman test	41.2	5.25e−7

corresponding hypotheses. The Bonferroni–Holm procedure rejects H_1 to H_{i-1} if i is the smallest integer such that $p_i > \alpha/(k-1)$. In addition, Friedman's test was used for a multiple comparisons test that aims to detect significant

Table 8 Comparison with ITGO and other improved PSO version

Fun	(Global)PSO-w	(Global)PSO-cf	UPSO	CLPSO	SG-PSO	SP-PSO	ITGO
F1	1.00e+03	1.82e+03	9.07e+03	**0.00e+00**	**0.00e+00**	1.48e−06	2.1525e−031
F2	1.07e+02	4.60e+02	1.03e+03	4.81e+01	2.08e+02	**2.20e+01**	2.4816e+001
F3	4.46e+03	5.84e+03	7.98e+03	5.46e+03	4.20e+03	4.03e+03	**2.0726e+003**
F4	4.11e+01	1.38e+02	2.00e+02	5.68e+00	2.86e+01	3.28e+00	**1.8064e+000**
F5	1.83e+00	2.70e+01	2.64e+01	2.42e−01	1.60e+00	**0.00e+00**	3.8775e−004
F6	9.87e+03	1.43e+04	3.19e+04	1.36e+04	1.24e+04	1.63e+01	**1.4495e−013**
F7	6.00e+06	5.88e+06	7.85e+06	5.49e+06	5.70e+06	1.73e+05	**9.6187e−002**
F8	2.48e+09	2.43e+09	1.47e+10	1.11e+10	2.99e+09	1.45e+05	**3.5497e+001**
F9	1.38e+02	2.41e+02	2.48e+02	2.16e+02	1.57e+02	1.29e+02	2.3719e+000
F10	2.28e+02	3.96e+02	3.90e+02	1.70e+02	2.31e+02	1.72e+02	1.3099e+002
F11	2.99e+01	3.53e+01	3.98e+01	3.27e+01	2.77e+01	2.89e+01	**2.2163e+001**

The results of improved PSO versions originated from paper [52]

Table 9 Comparison with ITGO and other improved PSO versions using Wilcoxon's signed-rank test and Bonferroni–Holm correction (means)

I	Algorithm	R+	R−	*p* Value	α/i	Hypothesis
3	ITGO versus PSO-w	6.0	60.0	0.003	0.017	REJECT
2	ITGO versus PSO-cf	0.0	55.0	0.003	0.025	REJECT
2	ITGO versus UPSO	0.0	55.0	0.003	0.025	REJECT
1	ITGO versus CLPSO	0.0	66.0	0.005	0.05	REJECT
1	ITGO versus SGPSO	0.0	66.0	0.005	0.05	REJECT
1	ITGO versus SPPSO	0.0	66.0	0.016	0.05	REJECT

Table 10 Average ranks of seven algorithms for data clustering dataset by Friedman test (means)

Algorithm	Average ranks
ITGO	1.27
SPPSO	2.18
SGPSO	3.82
CLPSO	3.82
PSO-w	4.00
PSO-cf	5.82
UPSO	6.82

Table 11 Results of Friedman test for seven algorithms

Method	Statistical value	*p* Value
Friedman test	52.51	1.47e−9

differences between the behaviors of two or more algorithms. At the end, average ranks are used to comparison.

In Tables 2 and 3, we compare ITGO to the well-known algorithm including PSO, DE, GSA, CLPSO, cPSO, rcGA, and TLBO by the means and standard of the error value. Bold font means the winner. Table 4 shows the comparison results by Wilcoxon's signed-rank test; '+' means the algorithm is winner, '=' means the two algorithms are equal, and '−' means the algorithm is lost. The statistical results show the winner number, equation number, and lost number as follows:

ITGO versus PSO:20/0/0
ITGO versus DE: 17/0/3
ITGO versus GSA:16/0/4
ITGO versus CLPSO:18/1/1
ITGO versus cPSO:20/0/0
ITGO versus rcGA:16/0/4
ITGO versus TLBO:15/0/5

We can see that ITGO algorithm has best performance than other algorithms. Table 5 shows the comparison results by Wilcoxon's signed-rank test with Bonferroni–Holm correction. For ITGO vs PSO, cPSO, DE, CLPSO, TLBO, GSA and rcGA, the *p* values are 8.85e−5, 8.85e−5, 2.53e−4, 2.54e−4, 4.63e−4, 8.91e−4, 1.01e−3, and 2.49e−3, which are less than α/i values. So the hypothesis H0 was rejected. In other words, ITGO algorithm has better performance than other algorithm. Tables 6 and 7 show the comparison results by Friedman test. Average ranks show that ITGO is better than other algorithms.

In order to understand the characteristic of our algorithm, we analyze the experimental results deeply. For F1–F4 functions, they are unimodal functions. These functions are used to verify the local search ability of algorithms. From Tables 2 and 3, we can see that the performance of rcGA is better than others for F1 function, and TLBO has better performance than others for F2, F3, and F4. Though ITGO is not the best one, its performance is better than PSO, GSA, CLPSO, and cPSO for F1 and better than PSO, DE, GSA, CLPSO, cPSO, and rcGA for F2 and F3. Sometimes, one algorithm has so fast convergence speed for unimodal problems, and it is not a

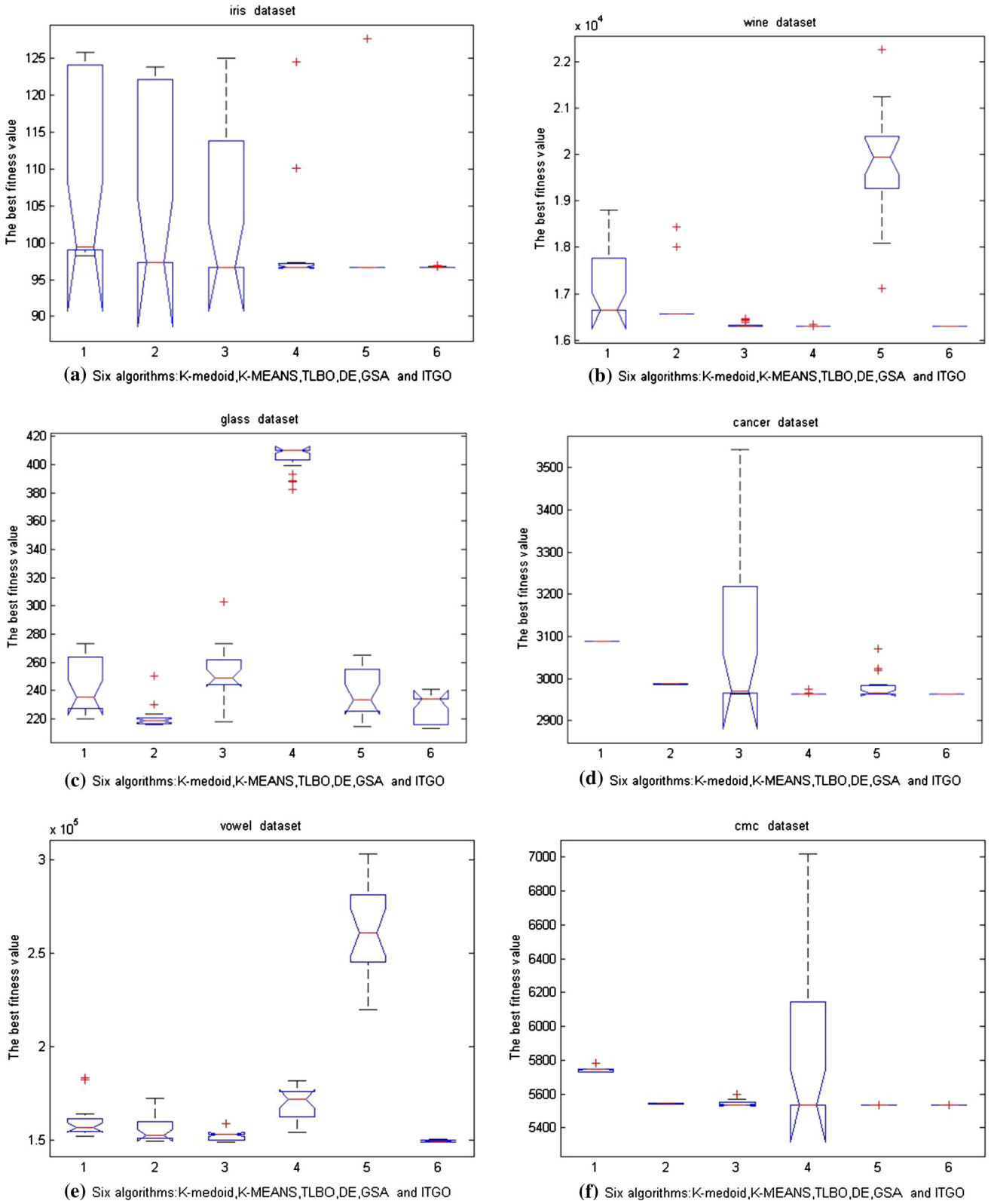

Fig. 6 Comparison results of five algorithms by box plot

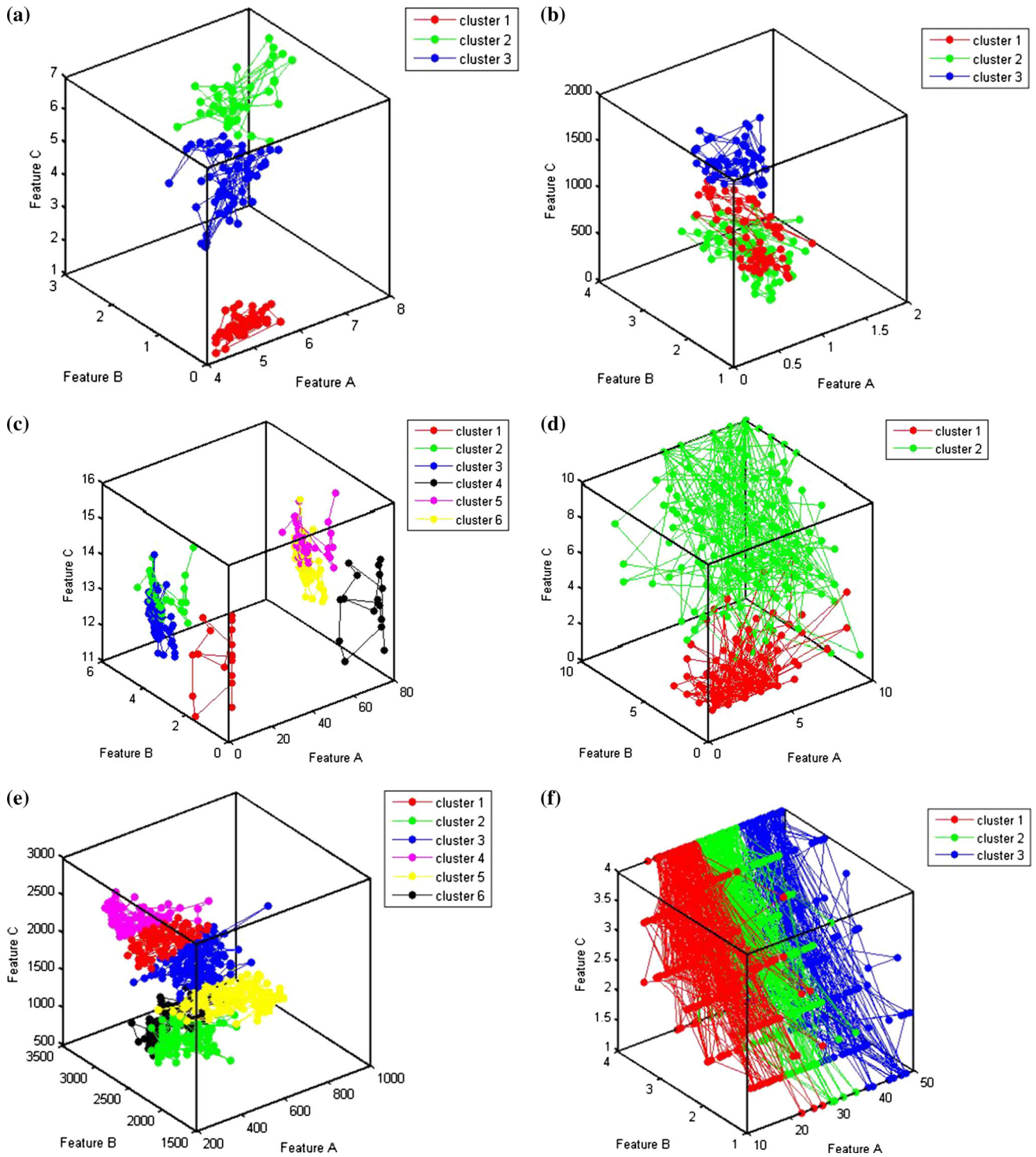

Fig. 7 Results of data clustering by ITGO. **a** Iris, **b** wine, **c** glass, **d** cancer, **e** vowel, **f** CMC

good thing, because it is perhaps easy to fall into the local optimum for multimodal problems. So good algorithms must consider it. F5–F15 functions are multimodal problems. These functions are used to verify the global search ability of algorithms. From Table 2, we can see that the performance of ITGO is better than PSO, DE and GSA for F5, F9, F10, F13, F14, and F15, while the performance of ITGO is better than CLPSO, cPSO, rcGA, and TLBO for F7, F8, F9, F12, F13, and F14. The results show that ITGO has powerful global search ability than many other

Table 12 Details of six datasets

Datasets	Number of clusters	Dimension	Distribution (sample numbers of each class)
Iris	3	4	150 (50, 50, 50)
Wine	3	13	178 (59, 71, 48)
Glass	6	9	214 (70, 76, 17, 13, 9, 29)
Cancer	2	9	683 (444, 239)
Vowel	6	3	871 (72, 89, 172, 151, 207, 180)
CMC	3	9	1473 (629, 334, 510)

Table 13 The sum of intra-cluster distances obtained by algorithms on different datasets

Datasets	Criterion	K-medoid	K-means	TLBO	DE	GSA	ITGO
Iris	Min	9.8214e+001	97.3259	96.6554	96.6554	96.6554	96.6554
	Max	1.2584e+002	123.8497	125.0844	124.5916	127.6676	**96.9682**
	MeanS	1.0674e+002	104.8645	103.6115	100.2417	98.2060	**96.7117**
	Std	1.1890e+001	11.8063	11.0483	8.8373	6.9345	**0.1065**
	Meadian	9.9482e+001	97.3462	96.6686	96.6913	96.6554	96.6554
Wine	Min	1.6637e+004	16,555.6794	16,295.1447	**16,292.1888**	17,123.2430	16,292.1904
	Max	1.8811e+004	19,436.9520	16,461.2366	16,333.0285	22,259.2894	**16,294.1875**
	MeanS	1.7188e+004	16,962.0423	16,329.2662	16,296.0207	19,767.6886	**16,293.1343**
	Std	9.6150e+002	733.8995	52.4949	9.0831	1194.7086	**0.7129**
	Meadian	1.6637e+004	16,555.6794	16,306.1358	16,293.8157	20,029.2376	**16,292.6767**
Glass	Min	2.2014e+002	215.7316	218.3590	382.3724	214.4027	**213.4988**
	Max	2.7309e+002	250.0580	303.1350	410.2468	265.2572	**241.1775**
	MeanS	2.4174e+002	228.7861	253.1780	405.0525	239.0503	**228.5708**
	Std	1.8643e+001	**7.6414**	18.3250	9.2617	16.8698	10.2850
	Meadian	2.3741e+002	**218.6841**	249.1153	410.2468	236.1694	234.1775
Cancer	Min	3.0891e+003	2986.9613	2964.3870	2964.3869	2964.3869	2964.3869
	Max	3.0891e+003	5115.4278	3542.0871	2974.8663	3071.7828	**2964.3874**
	MeanS	3.0891e+003	3031.7678	3097.1951	2965.0402	2980.4701	**2964.3871**
	Std	9.3312e−013	0.7485	195.5960	2.3327	27.7893	**0.0001**
	Meadian	3.0891e+003	2998.4278	2970.8433	2964.3870	2968.6077	**2964.3871**
Vowel	Min	1.5173e+005	149,413.1753	149,056.1753	154,016.1650	219,794.0771	**149,013.3922**
	Max	1.8338e+005	172,056.4440	158,615.9159	181,701.7314	302,756.5419	**150,172.0665**
	MeanS	1.5965e+005	169,190.1505	152,698.5111	169,612.0376	261,555.3147	**149,516.4777**
	Std	8.5940e+003	7002.5058	3372.3409	8317.1979	21692.0729	**441.3491**
	Meadian	1.5953e+005	153,992.5367	153,052.7057	172,055.3090	261,013.4525	**149,370.6384**
CMC	Min	5.7308e+003	**5542.1821**	5532.2446	5532.1847	5532.1847	5532.1881
	Max	5.7835e+003	5545.2928	5598.0817	7018.0412	5532.9950	**5532.2435**
	MeanS	5.7487e+003	5543.7253	5542.5356	5885.6379	5532.2642	**5532.2008**
	Std	1.8850e+001	1.5841	18.5901	627.5897	0.2123	**0.0127**
	Meadian	5.7450e+003	5545.0497	5532.7779	5532.1848	**5532.1847**	5532.1964

Bold values indicate the best fitness values for all the algorithms

algorithms for multimodal problems. Especially, F1–F15 functions only verify the performance of ITGO in low dimension (dimension = 10), so we verify the performance of ITGO for F16–F20 functions in high dimension (dimension = 100). From Tables 2 and 3, we can see that the performance of ITGO algorithm outperforms PSO, DE, GSA, CLPSO, CPSO, rcGA, and TLBO in high dimension including unimodal and multimodal functions. It is to say that our algorithm can solve more large complex problems.

Table 14 Comparison with ITGO and other algorithms for data clustering dataset using Wilcoxon signed-rank test and Bonferroni–Holm correction (means)

I	Algorithm	R+	R−	p Value	α/i	Hypothesis
1	ITGO versus DE	0.0	21.0	0.028	0.05	REJECT
1	ITGO versus GSA	0.0	21.0	0.028	0.05	REJECT
1	ITGO versus TLBO	0.0	21.0	0.028	0.05	REJECT
1	ITGO versus K-means	0.0	21.0	0.028	0.05	REJECT
1		0.0	21.0	0.028	0.05	REJECT

5.2 Comparison with ITGO and improved PSO versions

In this test, we choose the 11 benchmark functions from the paper [44, 53]. Population size and maximum iteration number are set to 30 and 10,000 according to the suggestion of paper [53]. The benchmark functions are multimodal functions except for F1 and F6 function. In addition, F1–F5 are standard benchmark functions, F6–F9 are shifted functions, and F10–F11 are shifted rotated functions. Details can be seen in paper [53].

In Table 8, we compare ITGO to the currently improved algorithms including PSO-w [53], PSO-cf [53, 54, 55], UPSO [56], CLPSO [47], SG-PSO [53], and SP-PSO [53] by the means of the error value. Bold font means the winner. Table 9 shows the comparison results by Wilcoxon's signed-rank test with Bonferroni–Holm correction. For ITGO versus PSO-w, PSO-cf, UPSO, CLPSO, SG-PSO, and SP-PSO, the p values are 0.003, 0.003, 0.003, 0.005, 0.005, and 0.016, which are less than α/i values. So the hypothesis H0 was rejected. In other words, ITGO algorithm has better performance than other algorithm. Tables 10 and 11 show the comparison results by Friedman test. Average ranks show that ITGO is better than other algorithms. Especially, from the results of Table 8, we can also see that the performance of ITGO is better than PSO-w, PSO-cf, UPSO, CLPSO, SG-PSO, and SP-PSO for F3, F4, F6, F7, F8, and F11 including standard or shifted or shifted rotated functions. It is to say that our algorithms can solve many different complex problems.

5.3 Comparison for data clustering

We choose six benchmark datasets with different dimensions from the repository of the machine learning databases [57], namely, iris, wine, glass, Wisconsin breast cancer, vowel, and contraceptive method choice (CMC). This dataset has been used to solve the data clustering problem widely in other algorithms [16, 33–35].

The iris dataset contains three categories, where each category refers to a type of iris plant. In the iris dataset, there are four attributes, which are sepal length in cm, sepal width in cm, petal length in cm, and petal width in cm.

The wine dataset contains chemical analyses of wines derived from three different cultivars, with 13 attributes, namely, alcohol, malic acid, ash, alkalinity of ash, magnesium, total phenols, flavonoids, non-flavonoid phenols, proanthocyanins, color intensity, hue, OD280/OD315 of diluted wines, and praline.

The glass identification dataset contains six different types of glass, which are, float-processed building windows, non-float-processed building windows, float-processed vehicle windows, containers, tableware, and headlamps. There are nine attributes, namely, refractive index, sodium, magnesium, aluminum, silicon, potassium, calcium, barium, and iron.

The Wisconsin breast cancer dataset consists of two categories in the data: malignant and benign. There are nine features: clump thickness, cell size uniformity, cell shape uniformity, marginal adhesion, single epithelial cell size, bare nuclei, bland chromatin, normal nucleoli, and mitoses.

The vowel dataset consists of 871 patterns (rows). There are six overlapping vowel classes (Indian Telugu vowel) and three input features (formant frequencies).

The CMC dataset is a subset of the 1987 National Indonesia Contraceptive Prevalence Survey. The objects are married women who either were not pregnant or did not know if they were at the time of interview. The problem involves predicting the choice of the current contraceptive method of a woman based on her demographic and socioeconomic characteristics.

Table 12 summarizes the main characteristics of the used six datasets. The performance of the ITGO algorithm is compared against well-known and the most recent algorithms reported in the literature, including K-means [26], K-medoid, DE [9], GSA [28], and the TLBO [35]. In order

Table 15 Average ranks of six algorithms for data clustering dataset by Friedman test (means)

Algorithm	Average ranks
ITGO	1.00
GSA	3.67
TLBO	3.83
K-means	3.83
DE	4.00
K-medoids	4.67

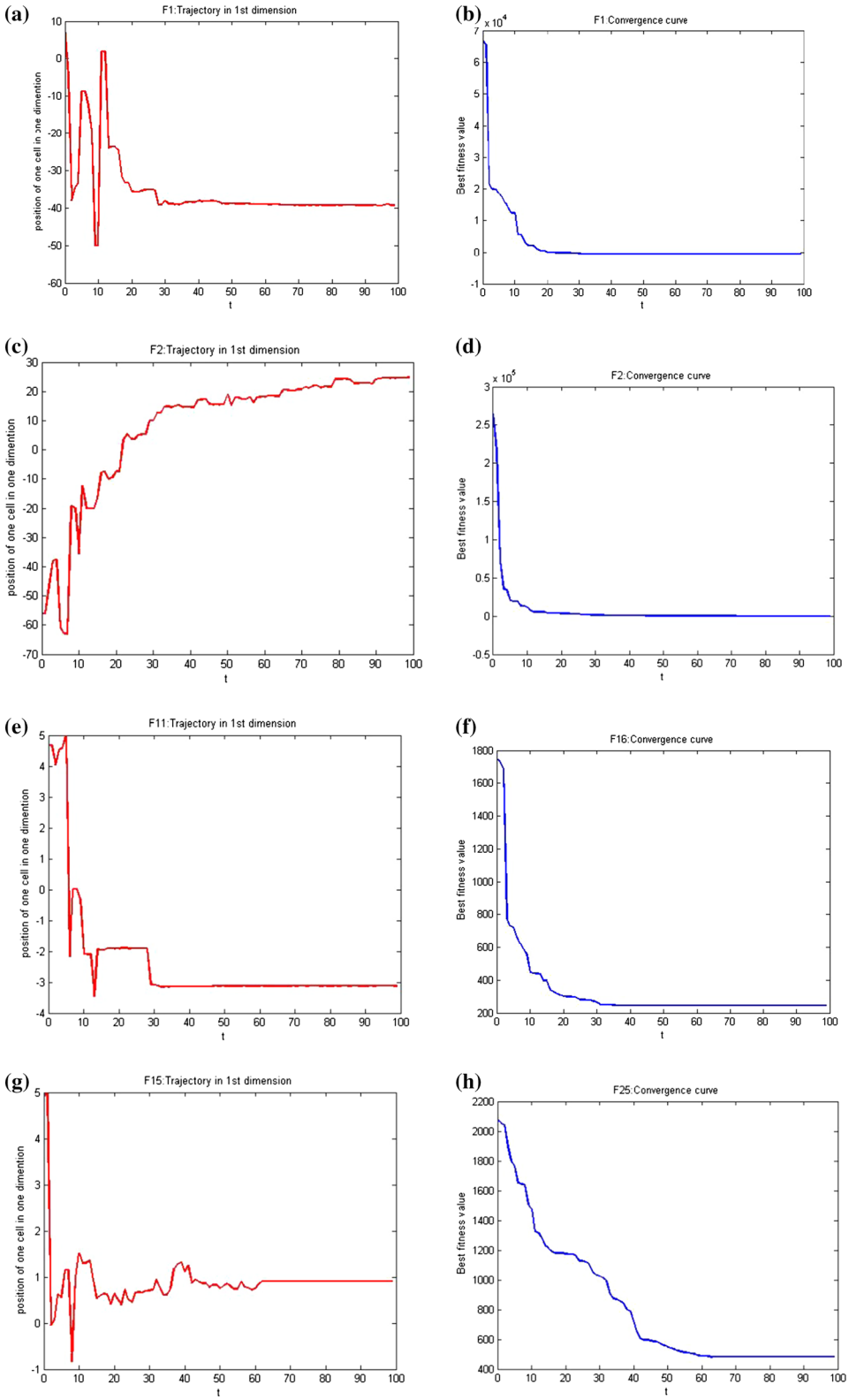

Fig. 8 Trajectory of one of the search agents in one of the dimension as well as the convergence curves

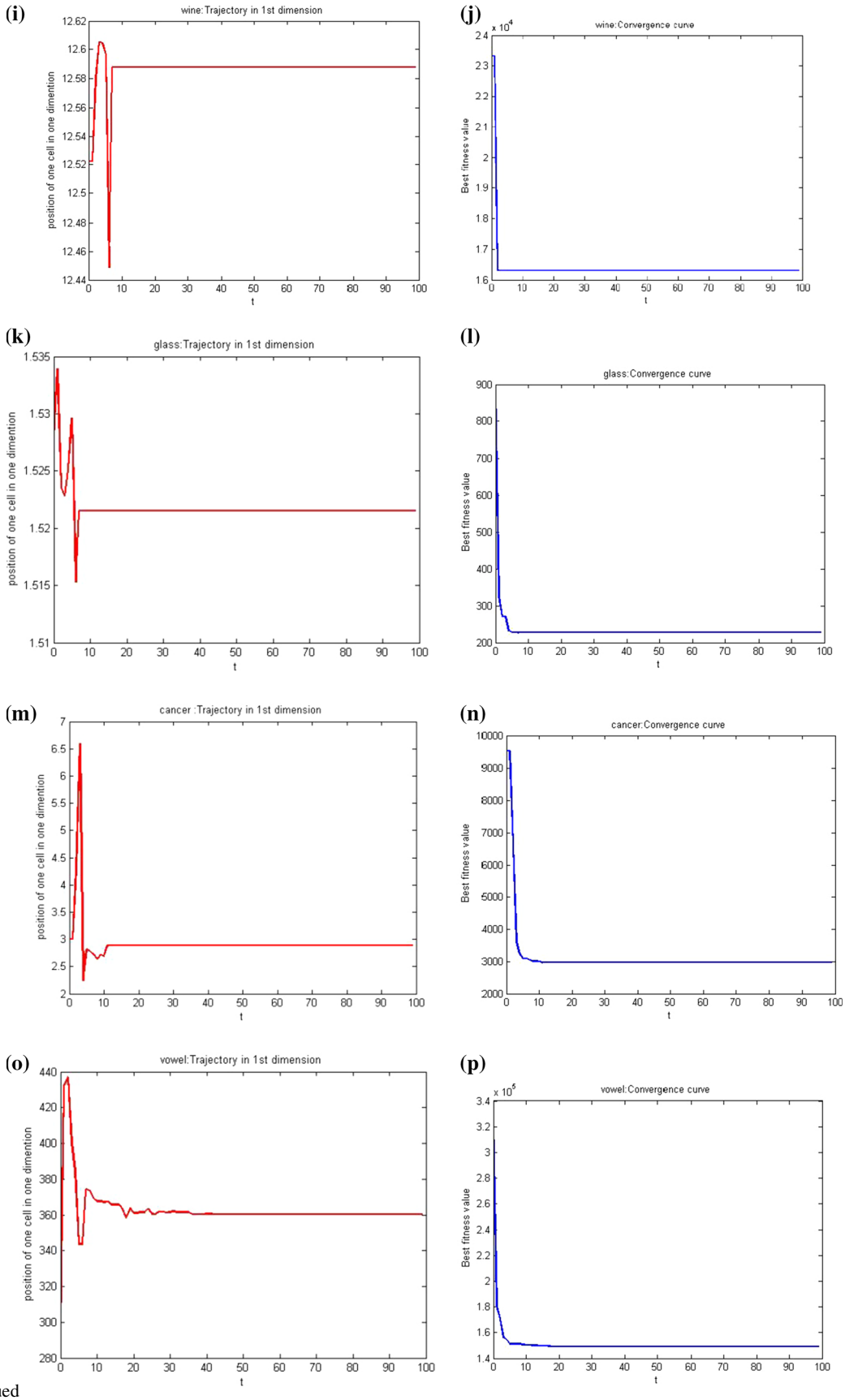

Fig. 8 continued

Table 16 Results of Friedman test for six algorithms

Method	Statistical value	p Value
Friedman test	13.91	0.016

Table 17 The best cluster center of Iris dataset obtained by ITGO

Iris		
Center 1	Center 2	Center 3
5.9343618044	6.7333769288	5.0121702151
2.7978170955	3.0679155552	3.4031157613
4.417933082	5.6301165765	1.4716522725
1.4173204962	2.1068108464	0.2353811945

Table 18 The best cluster center of wine dataset obtained by ITGO

Wine		
Center 1	Center 2	Center 3
12.8155322802	12.5108007404	13.6993223399
2.5332961783	2.3151926224	1.8705547120
2.3703969396	2.3384118602	2.3996847854
19.5109267095	21.3027113363	16.8885868716
98.9411828106	92.5260451024	105.2378750705
2.0654697960	2.0413978786	2.8574204986
1.4875042173	1.7832298798	3.0570760079
0.4290907179	0.4241488015	0.2870843030
1.42638999441	1.4267508945	2.0208455221
5.7840510615	4.3715599107	5.6431297110
0.9180374962	0.9448212694	1.057125834
2.2139961273	2.4537720286	3.028053459
686.9558829111	463.6155503031	1137.3766459926

to evaluate and compare the performance of the algorithms, criteria can been defined as follows: The distance between each data object and the center of each cluster can be found according to the formula (1). The smaller the sum of intra-cluster distances, the higher the quality of the clustering. So, the sum of intra-cluster distances is the evaluation values in our work, namely, fitness value for meta-heuristic algorithms. Each algorithm is run for 20 times independently, and the min, max value, means, standard, median for the fitness are used to comparison. In addition, box plots are used to verify the performance of the algorithms. Box plots can be very convenient to compare the performance of algorithms in graphical form. In a typical case, endpoint I of the box is on the quarter point, the length of the box is the distance of the quarter point 3 plus quarter point 1, median number marks with line, two dotted lines in the box (called a beard) extends to the maximum and minimum values, and discrete points can also be displayed in box.

Table 13 shows the fitness values obtained by algorithms. For iris datasets, ITGO is better than K-medoid, K-means, DE, GSA, and TLBO according to the max, means and standard value. ITGO equals GSA by min and median value, and smaller than our algorithms. For wine datasets, ITGO is better than K-medoid, K-means, DE, GSA, and TLBO according to the max, means, standard value, and medians. For glass datasets, ITGO is better than other algorithms by the min and max value except for the means, SD, and median. For cancer and vowel datasets, ITGO is better than other algorithms by the min, max,

Table 20 The best cluster center of Wisconsin breast cancer dataset obtained by ITGO

Cancer	
Center 1	Center 2
2.8892052753	7.1170611327
1.1277263106	6.6410829399
1.2005263587	6.6256351703
1.1641527738	5.61446411312
1.9933892969	5.2406332698
1.1209414044	8.1008978365
2.0055231423	6.0788141111
1.1013696495	6.0219507718
1.0316155511	2.3257067501

Table 19 The best cluster center of glass dataset obtained by ITGO

Glass					
Center 1	Center 2	Center 3	Center 4	Center 5	Center 6
1.5208199521	1.5323359710	1.5235646736	1.5195830179	1.5227745060	1.5195076554
13.2886409616	13.7842639421	13.0794266036	13.8716820038	11.8980458215	14.6484918869
0.4066367468	3.5091029793	3.5267122885	2.2553041540	0.1415268861	0.0957039581
1.4989757453	1.0351739927	1.3537539771	2.6947371581	1.0254221336	2.2457734782
73.0381403173	71.8934956113	72.8628305544	71.3379456629	71.9542173382	73.2916682433
0.3813479545	0.2102331969	0.5733710000	2.5980806592	0.2352618299	0.0214452200
11.1973938945	9.4220745593	8.3710078600	6.0183574563	14.4575952997	8.7322876219
0.0684541398	0.0410425993	0.0118450934	1.4446776413	0.2403338127	0.9688689870
0.0823057996	0.0692142244	0.0703303122	0.2145966979	0.1418565074	0.0381674712

Table 21 The best cluster center of vowel dataset obtained by ITGO

Vowel					
Center 1	Center 2	Center 3	Center 4	Center 5	Center 6
377.8087719786	405.1074838756	508.1473570044	357.2783180237	629.4973795450	445.3023937825
2144.8307319333	1017.2479668195	1828.9373876649	2291.4942826890	1304.6896612483	992.42058353785
2677.6725686875	2326.6606192387	2550.9674148433	2977.4104518448	2326.2986386547	2675.0376202512

means SD, and median value. For CMC dataset, ITGO is better than other algorithms, too. To verify the robustness, box plots are used in Fig. 6. The number 1, 2, 3, 4, 5, and 6 of the horizontal ordinate corresponds to the K-medoid, K-means, TLBO, DE, GSA, and ITGO algorithms. We can see that ITGO algorithm is superior to K-medoid, K-means, TLBO, DE, and GSA for the five datasets, except for the glass dataset. For glass dataset, ITGO algorithm is still better than TLBO, DE, and GSA algorithms. At the end, Fig. 7 shows the best results of ITGO algorithm for data clustering of the six datasets in three-dimensional space. Three features are chosen randomly from the six datasets. Tables 17, 18, 19, 20, 21, and 22 show the best center of clusters for the six datasets. Tables 14, 15, and 16 show the comparison results of ITGO for K-medoid, K-means, TLBO, DE and GSA using Wilcoxon signed-rank test and Bonferroni-Holm correction and Friedman test, which indicates that the performance of ITGO is better than other algorithms. In addition, the Fig. 8 shows the trajectory and the convergence curve of the first particle, in which changes of the first search agent in its first dimension can be observed. It can be seen that there are abrupt changes in the initial steps of iterations which are decreased gradually over the course of iterations. According to van den Bergh and Engelbrecht [58], this behavior can guarantee that a SI algorithm eventually convergences to a point in search space.

Table 22 The best cluster center of CMC dataset obtained by ITGO

CMC		
Center 1	Center 2	Center 3
24.4173448044	33.4928129132	43.6373284955
3.0412508930	3.1355623404	3.0025594997
3.5135065520	3.5533496585	3.4550662188
1.7934692739	3.6495463152	4.5853368309
0.9290658593	0.7888998679	0.7948559375
0.7945167688	0.6974427717	0.7628553012
2.3058817181	2.0998383876	1.8224881412
2.9716517275	3.2871792908	3.4340904083
0.0381008990	0.0593486457	0.0913074179

5.4 The computation complexity of ITGO algorithms

In this section, we use the method proposed in [59] for the computational complexity. The computational complexity of the proposed method depends on the number of iterations, the number of cells (particles), the number of fitness evaluations, the repartition of tumor, the mechanism of proliferative cell growth, the mechanism of quiescent cell growth, the mechanism of dying cell growth, the mechanism of invasive cell growth, and the random walk mechanism. So, the overall computational complexity is as follows:

$$\begin{aligned} O(ITGO) = O(T * (&O(repartition) + O(proliferative\,cell) \\ &+ O(quiescent\,cell) + O(dying\,cell) \\ &+ O(invasive\,cell) + O(random\,walk))) \end{aligned}$$

where T is the maximum number of iterations. The computational complexity of the repartition of tumor cells is of $O(p * p)$ in the worst. p is the number of population size. We utilized quicksort to the repartition of tumor cells, and the computational complexity of the repartition is of $O(p * \log p)$ in the best case. The computational complexity of proliferative cell growth is of $O(npFes)$ in the implementation where $npFes$ is the number of fitness evaluations for the proliferative cell. Note that the $npFes$ belong to [0, Max_num], Max_num is the number of the proliferative cell. Because the proliferative cell operator is used due to the probability $rand < \frac{t}{T}$, t is the current number of iteration and T is the max number of iteration. The computational complexity of quiescent cell growth is of $O(qpFes+qp*qp)$ in the implementation where qp is the number of the implemented quiescent cell and $qpFes$ is the number of fitness evaluations for quiescent cell. To choose the best two neighbors of quiescent cell, sort operator should be used. We utilized quicksort to the quiescent cell growth, the computational complexity is of $O(qpFes+qp * log(qp))$ in the best case. The computational complexity of dying cell growth is of $O(dpFes)$ in the implementation where $dpFes$ is the number of fitness evaluations for dying cell. The computational complexity of invasive cell growth is of $O(qpFes/2)$ in the implementation where $qpFes/2$ is the number of fitness evaluations for invasive cell. Since half of the dying cells are replaced by the invasive cells, the computational

Table 23 Results of beta parameter for shifted rotated Ackley's Function in CEC2005

Beta	0.8	0.9	1.0	1.1	1.2	1.3	1.4	1.5	1.6	1.7	1.8
Means	2.0289e+001	2.0278e+001	2.0282e+001	2.0284e+001	2.0286e+001	2.0268e+001	2.0273e+001	2.0242e+001	**2.0238e+001**	**2.0233e+001**	2.0240e+001
SD	6.4375e−002	6.0512e−002	7.1161e−002	6.9060e−002	5.4298e−002	5.4752e−002	6.3572e−002	5.3723e−002	**5.0657e−002**	**5.2495e−002**	5.4462e−002

Bold values indicate the best fitness value for ITGO algorithm with different parameter values

Table 24 Results of beta parameter for hybrid composition function 1 in CEC2005

Beta	0.8	0.9	1.0	1.1	1.2	1.3	1.4	1.5	1.6	1.7	1.8
Means	2.0318e+002	1.5039e+002	2.0535e+002	1.8018e+002	2.2451e+002	1.9597e+002	1.9191e+002	**1.1896e+002**	2.0035e+002	1.8912e+002	1.5952e+002
SD	1.8730e+002	1.9538e+002	2.1077e+002	1.8993e+002	2.0770e+002	2.0058e+002	1.8738e+002	**1.5609e+002**	1.7681e+002	1.8171e+002	1.6690e+002

Bold values indicate the best fitness value for ITGO algorithm with different parameter values

Table 25 Results of c parameter for shifted rotated Ackley's Function in CEC2005

c	1	2	3	4	5	6	7	8	9	10
Means	2.0245e+001	2.0250e+001	2.0248e+001	**2.0242e+001**	2.0264e+001	2.0265e+001	2.0263e+001	2.0258e+001	2.0256e+001	2.0272e+001
SD	6.3239e−002	6.7589e−002	6.0569e−002	**5.3795e−002**	4.6663e−002	5.9379e−002	6.4900e−002	6.0927e−002	6.6024e−002	6.9321e−002

Bold values indicate the best fitness value for ITGO algorithm with different parameter values

Table 26 Results of c parameter for hybrid composition function 1 in CEC2005

c	1	2	3	4	5	6	7	8	9	10
Means	1.5177e+002	2.2019e+002	1.9629e+002	1.6513e+002	1.1896e+002	1.1467e+002	**8.7157e+001**	1.2505e+002	1.4026e+002	1.8665e+002
SD	1.6058e+002	1.9192e+002	1.8864e+002	1.8164e+002	1.5609e+002	1.7477e+002	**1.4555e+002**	1.7860e+002	1.8609e+002	2.0616e+002

Bold values indicate the best fitness value for ITGO algorithm with different parameter values

Table 27 Results of *d* parameter for shifted rotated Ackley's function in CEC2005

d	3 (8)	4	5	6	7	8	9	10	11	12
Means	2.0261e+001	2.0247e+001	2.0231e+001	2.0277e+001	2.0261e+001	**2.0223e+001**	2.0270e+001	2.0269e+001	2.0276e+001	2.0241e+001
SD	5.6371e−002	8.4024e−002	9.2793e−002	6.5377e−002	6.1417e−002	**6.3164e−002**	3.7604e−002	5.6178e−002	4.1832e−002	6.2713e−002

Bold values indicate the best fitness value for ITGO algorithm with different parameter values

Table 28 Results of *d* parameter for hybrid composition function 1 in CEC2005

d	3 (15)	4	5	6	7	8	9	10	11	12
Means	1.6265e+002	**8.7662e+001**	2.1070e+002	2.2516e+002	1.4613e+002	1.6128e+002	2.1313e+002	2.5815e+002	2.0007e+002	1.7833e+002
SD	1.8973e+002	**1.4498e+002**	1.9995e+002	2.0151e+002	1.8601e+002	1.9143e+002	1.9263e+002	1.9113e+002	1.9878e+002	1.8317e+002

Bold values indicate the best fitness value for ITGO algorithm with different parameter values

complexity of random walk is of $O(rwFes)$ in the implementation where $rwFes$ is the number of fitness evaluations by random walk. Therefore, the final computational complexity of the proposed algorithm is as follows:

$$\begin{aligned} O(ITGO) = O(T * (O(p * logp + O(npFes) \\ + O(qpFes + qp * log(qp))) + O(dpFes) \\ + O(dpFes/2)) + O(rwFes))) \end{aligned}$$

where T is the maximum number of iterations, p is the population size, $npFes$ is the number of fitness evaluations for the proliferative cell, $qpFes$ is the number of fitness evaluations for the quiescent cell, qp is the number of quiescent cell, $dpFes$ is the number of fitness evaluations for the dying cell, and $rwFes$ is the number of fitness evaluations by the random walk.

6 Parameter tuning

Three parameters were used in ITGO, namely, *beta, c*, and *d*. Nevertheless, a complete evaluation on all possible combinations of these parameters is impractical. The parameter tuning strategy, as reported in [60, 61], is used to find an appropriate parameter combination of ITGO for the good performances. Two multimodal functions (F8 and F15) are chosen and *beta* ranges in [0.8, 1.8], *c* ranges in [1, 10], whereas *d* ranges in [3, 12]. We first initialize the parameter combination of [*beta*, *c*, *d*] as the mean values of their respective lower and upper boundary values, i.e., [1.3, 5, 8], *c* and *d* are integer. Based on this initial combination, we adjust the parameters one at a time. We first vary parameter *beta* from 0.8 to 1.8 with *c* and *d* are fixed. Then we choose the best *beta* value due to the best fitness value. In the next step, we fix the *beta* and *d* value, adjust *c* from 1 to 10. At the end, we vary parameter *d* value from 3 to 12 as so on.

From Tables 23 and 24, the simulation results show that *beta* value should be chosen as 1.5, or 1.6 or 1.7 for function 8 and 1.5 for function 15. Therefore, we set *beta* value as 1.5. We use new *beta* value and fixed *d* value to adjust *c* value from 1 to 10. From Tables 25 and 26, we can see that *c* should be chosen as 4 or 8 or 9 for function 8 and 7 or 8 for function 15. So, we set *c* value as 8, because when $c = 8$, the function 8 and function 15 can obtain better fitness value, too. At the end, we set the *beta* value as 1.5 and *c* value as 8, adjust *d* value from 3 to 12. From Tables 27 and 28, we can see that *d* value should be chosen as 8 for function 8 and 4 or 8 or 9 for function 15. So we set *d* value as 8. Then the parameter tuning process is completed. The experimental findings suggest that we can set the parameter combination [*beat, c, d*] as [1.5, 8, 8] in the performance evaluation.

7 Discussions

In this section, we discuss some aspects of the proposed algorithm (ITGO). Though, some operations are similar to the operations of particle swarm optimization, such as the operation of the quiescent cell and dying cell and some operations are similar to the DE or rcGA, because the cross operation is used; it is different from them. Firstly, there are many 'leaders' (chosen randomly from the proliferative cell group) which guide other agents; secondly, the cross operation is different from DE or rcGA. In addition, there are many new operations such as the invasive behavior of the proliferative cell with the Levy distribution, the random walk operations, and the search of invasive cell. Especially, we achieved the interaction of the different subpopulations, which can enhance the diversity of the proposed algorithm. In addition, the analysis of the experimental results shows that the proposed algorithms has fast convergence speed for many multimodal or high dimension problems. Standard value shows the stability of our algorithm. It is shown that our algorithm is convenient and efficient to solve the data clustering problems. Then we analyze the computation complexity, the proposed algorithm is ranked third, maybe due to other computation such as the levy distribution and the neighbor numbers of quiescent cell. Though, there are some disadvantages of the proposed algorithm such as the complexity of realization and parameter tuning, there are many advantages such as, fast convergence speed, better stability, and better search ability.

8 Conclusion

In this paper, we proposed a new meta-heuristic algorithm called 'ITGO' for data cluttering. According to the mechanism of tumor growth, tumor cells can be divided into five types, namely: invasive cell, proliferative cell, quiescent cell, dying cell, and necrotic cell. In order to implement the invasive tumor growth optimization, five different behaviors occur in tumor for different cells: invasive cell growth, proliferative cell growth, quiescent cell growth, dying cell growth, and random walk. Necrotic cell has been dead, so it does not grow. Many interactions among invasive cell, proliferative cell, quiescent cell and dying cell enhance the search ability of our algorithm. Then, we use CEC 2005 and CEC 2008 benchmark functions to verify the performance of our algorithm, compared with the well-known algorithms including PSO, DE, GSA, CLPSO, cPSO, PSO-w, PSO-cf, UPSO, SG-PSO, SP-PSO rcGA, and TLBO by the Wilcoxon's signed-rank test with Bonferroni–Holm correction method and the Friedman's test. At the end, six benchmark datasets are chosen to verify the performance of our algorithm for data clustering,

compared with K-means, DE, GSA, and TLBO. Experimental results show that ITGO algorithm can not only be used to solve data clustering problem better, but also be used to solve other optimization problems, which can be widely used in other engineering and science fields. In future, many improvements in ITGO can be started due to the disadvantages of it, such as the better operations and better parameter tuning method.

Acknowledgments This work is supported by the National Natural Science Foundation Project (No. 61070092/F020504); the building of strong Guangdong Province for Chinese Medicine Scientific Research (20141165); the Humanities and social science fund project for Guangdong Pharmaceutical University (RWSK201409).

References

1. Kang F, Li J, Ma Z (2011) Rosenbrock artificial bee colony algorithm for accurate global optimization of numerical functions. Inf Sci 181:3508–3531
2. Kundu D, Suresh K, Ghosh S, Das S, Panigrahi BK, Das S (2011) Multi-objective optimization with artificial weed colonies. Inf Sci 181:2441–2454
3. Akay B, Karaboga D (2012) A modified artificial bee colony algorithm for real-parameter optimization. Inf Sci 192:120–142
4. Yeh W-C (2012) Novel swarm optimization for mining classification rules on thyroid gland data. Inf Sci 197:65–76
5. Christmas J, Keedwell E, Frayling TM, Perry JRB (2011) Ant colony optimisation to identify genetic variant association with type 2 diabetes. Inf Sci 181:1609–1622
6. Zhang Y, Gong D-W, Ding Z (2012) A bare-bones multi-objective particle swarm optimization algorithm for environmental/economic dispatch. Inf Sci 192:213–227
7. Manoj VJ, Elias E (2012) Artificial bee colony algorithm for the design of multiplier-less nonuniform filter bank transmultiplexer. Inf Sci 192:193–203
8. Holland J (1975) Adaptation in natural and artificial systems. University of Michigan Press, Ann Arbor
9. Storn R, Price K (1997) Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces. J Glob Optim 11:341–359
10. Dorigo M (1992) Optimization, learning and natural algorithms, Ph.D. Thesis, Politecnico di Milano, Italy
11. Kennedy J, Eberhart RC (1995) Particle swarm optimization. In: Proceedings of IEEE international conference on neural networks, Piscataway, NJ, pp 1942–1948
12. Tang D, Cai Y, Zhao J, Xue Y (2014) A quantum-behaved particle swarm optimization with memetic algorithm and memory for continuous non-linear large scale problems. Inf Sci 289:162–189
13. Rao RV, Savsani VJ, Vakharia DP (2012) Teaching–learning-based optimization: an optimization method for continuous non-linear large scale problems. Inf Sci 183:1–15
14. Yang XS, Deb S (2009) Cuckoo search via Levy flights. In: Proceedings of World congress on nature & biologically inspired computing. IEEE Publications, USA, pp 210–214
15. Rashedi E, Nezamabadi-Pour H, Saryazdi S (2009) GSA: a gravitational search algorithm. Inf Sci 179:2232–2248
16. Hatamlou A (2013) Black hole: a new heuristic optimization approach for data clustering. Inf Sci 222:175–184
17. Gandomi AH, Alavi AH (2012) Krill Herd: a new bio-inspired optimization algorithm. Commun Nonlinear Sci Numer Simul 17(12):4831–4845
18. Karaboga D (2005) An idea based on honey bee swarm for numerical optimization. Technical Report TR06, Erciyes University, Engineering Faculty, Computer Engineering Department
19. Simon D (2008) Biogeography-based optimization. IEEE Trans Evol Comput 12(December):702–713
20. Civicioglu P (2013) Artificial cooperative search algorithm for numerical optimization problems. Inf Sci 229:58–76
21. Mirjalili S, Mirjalili SM, Lewis A (2014) Grey wolf optimizer. Adv Eng Softw 69:46–61
22. Mehrabiana AR, Lucas C (2006) A novel numerical optimization algorithm inspired from weed colonization. Ecol Inf 1:1355–1366
23. Shelokar PS, Jayaraman VK, Kulkarni BD (2004) An ant colony approach for clustering. Anal Chim Acta 509:187–195
24. Adib AB (2005) NP-hardness of the cluster minimization problem revisited. J Phys A Math Gen 40:8487–8492
25. Jain AK, Murty MN, Flynn PJ (1999) Data clustering: a review. In: Computing surveys. ACM, pp 264–323
26. Jain AK (2010) Data clustering: 50 years beyond K-means. Pattern Recogn Lett 31:651–666
27. Das S, Abraham A, Konar A (2009) Automatic hard clustering using improved differential evolution algorithm. In: Studies in computational intelligence, pp 137–174
28. Hatamlou A, Abdullah S, Nezamabadi-Pour H (2011) Application of gravitational search algorithm on data clustering. In: Rough sets and knowledge technology. Springer, Berlin, pp 337–346
29. Hatamlou A, Abdullah S, Nezamabadi-Pour H (2012) A combined approach for clustering based on K-means and gravitational search algorithms. Swarm Evol Comput 6:47–52
30. Izakian H, Abraham A (2011) Fuzzy C-means and fuzzy swarm for fuzzy clustering problem. Expert Syst Appl 38:1835–1838
31. Senthilnath J, Omkar SN, Mani V (2011) Clustering using firefly algorithm: performance study. Swarm Evol Comput 1:164–171
32. Ghosh A, Halder A, Kothari M, Ghosh S (2008) Aggregation pheromone density based data clustering. Inf Sci 178:2816–2831
33. Hatamlou A, Abdullah S, Hatamlou M (2011) Data clustering using big bang–big crunch algorithm. In: Communications in computer and information science, pp 383–388
34. Fathian M, Amiri B, Maroosi A (2007) Application of honey-bee mating optimization algorithm on clustering. Appl Math Comput 190:1502–1513
35. Satapathy SC, Naik A (2011) Data clustering based on teaching–learning-based optimization. In: Panigrahi BK, Suganthan PN, Das S, Satapathy SC (eds) Swarm, evolutionary, and memetic computing. Lecture notes in computer science, vol 7077. Springer, Berlin, Heidelberg, pp 148–156
36. Unnikrishnan GU, Unnikrishnan VU, Reddy JN et al (2010) Review on the constitutive models of tumor tissue for computational analysis. Appl Mech Rev 63(4):040801
37. Deisboeck TS, Berens ME, Kansal AR, Torquato S, Rachamimov A et al (2001) Patterns of self-organization in tumor systems: complex growth dynamics in a novel brain tumor spheroid model. Cell Prolif 34:115–134
38. Mahmood MS, Mahmood S, Dobrota D (2011) Formulation and numerical simulations of a continuum model of avascular tumor growth. Math Biosci 231(2):159–171
39. Jeon J, Quaranta V, Cummings PT (2010) An off-lattice hybrid discrete-continuum model of tumor growth and invasion. Biophys J 98(1):37–47
40. Jiao Y, Torquato S (2011) Emergent behaviors from a cellular automaton model for invasive tumor growth in heterogeneous microenvironments. PLoS Comput Biol 7(12):e1002314

41. Roose T, Chapman SJ, Maini PK (2007) Mathematical models of avascular tumor growth. SIAM Rev 49(2):179–208
42. Mantegna RN (1991) Levy walks and enhanced diffusion in Milan stock exchange. Phys A 179:232–242
43. Mantegna RN (1994) Fast, accurate algorithm for numerical simulation of Levy stable stochastic process. Phys Rev E 5(49):4677–4683
44. Suganthan PN, Hansen N, Liang JJ, Deb K, Chen Y-P, Auger A, Tiwari S (2005) Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization. Technical Report, Nanyang Technological University, Singapore and KanGAL Report Number 2005005 (Kanpur Genetic Algorithms Laboratory, IIT Kanpur)
45. Tang K, Yao X, Suganthan PN, MacNish C, Chen Y-P, Chen C-M, Yang Z (2007) Benchmark functions for the CEC'2008 special session and competition on large scale global optimization, Technical Report. University of Science and Technology of China (USTC), School of Computer Science and Technology, Nature Inspired Computation and Applications Laboratory (NICAL), China
46. Kennedy J, Eberhart RC (1995) Particle swarm optimization. In: Proceeding IEEE international conference neural network, Perth, Western Australia, pp 1942–1948
47. Liang JJ, Qin AK, Suganthan PN, Baskar S (2006) Comprehensive learning particle swarm optimizer for global optimization of multimodal functions. IEEE Trans Evol Comput 10(3):281–295
48. Zhang Q (2011) http://dces.essex.ac.uk/staff/qzhang/. Accessed 6 Oct 13
49. Neri F, Mininno E, Iacca G (2013) Compact particle swarm optimization. Inf Sci 239:96–121
50. Mininno E, Cupertino F, Naso D (2008) Real-valued compact genetic algorithms for embedded microcontroller optimization. IEEE Trans Evol Comput 12:203–219
51. https://sites.google.com/site/tlbocodes
52. Derrac J, García S, Molina D, Herrera F (2011) A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. Swarm Evol Comput 1:3–18
53. Shin YB, Kita E (2014) Search performance improvement of particle swarm optimization by second best particle information. Appl Math Comput 246:346–354
54. Shi Y, Eberhart R (1998) A modified particle swarm optimizer. In: Proceedings of the IEEE international conference evolutionary computation, pp 69–73
55. Clerc M, Kennedy J (2002) The particle swarm-explosion, stability, and convergence in a multi-dimensional complex space. IEEE Trans Evol Comput 6:58–73
56. Parsopoulos KE, Vrahatis MN (2004) UPSO: a unified particle swarm optimization scheme, lecture series on computer and computation science, vol 1. Springer, Berlin, pp 868–873
57. Merz CJ, Blake CL (1996) UCI repository of machine learning databases. http://www.ics.uci.edu/-mlearn/MLRepository.html
58. van den Bergh F, Engelbrecht A (2006) A study of particle swarm optimization particle trajectories. Inf Sci 176:937–971
59. Mirjalili S, Mirjalili SM, Lewis A (2014) Let a biogeography-based optimizer train your multi-layer perceptron. Inf Sci 269:188–209
60. Lim WH, Isa NAM (2014) Bidirectional teaching and peer-learning particle swarm optimization. Inf Sci 280:111–134
61. Lam AYS, Li VOK, Yu JJQ (2012) Real-coded chemical reaction optimization. IEEE Trans Evol Comput 16:339–353