Computers & Geosciences 46 (2012) 229–247

Contents lists available at SciVerse ScienceDirect

Computers & Geosciences

journal homepage: www.elsevier.com/locate/cageo

Transforming geocentric cartesian coordinates to geodetic coordinates by using differential search algorithm

Pinar Civicioglu *

College of Aviation, Dept. of Aircraft Electrics and Electronics, Erciyes University, Kayseri, Turkey

ARTICLE INFO

Article history:
Received 20 May 2011
Received in revised form
15 December 2011
Accepted 16 December 2011
Available online 27 December 2011

Keywords:
Coordinate transformation
Geocentric cartesian coordinates
Geodetic coordinates
Differential search algorithm

ABSTRACT

In order to solve numerous practical navigational, geodetic and astro-geodetic problems, it is necessary to transform geocentric cartesian coordinates into geodetic coordinates or vice versa. It is very easy to solve the problem of transforming geodetic coordinates into geocentric cartesian coordinates. On the other hand, it is rather difficult to solve the problem of transforming geocentric cartesian coordinates into geodetic coordinates as it is very hard to define a mathematical relationship between the geodetic latitude (φ) and the geocentric cartesian coordinates (X, Y, Z). In this paper, a new algorithm, the Differential Search Algorithm (DS), is presented to solve the problem of transforming the geocentric cartesian coordinates into geodetic coordinates and its performance is compared with the performances of the classical methods (i.e., Borkowski, 1989; Bowring, 1976; Fukushima, 2006; Heikkinen, 1982; Jones, 2002; Zhang, 2005; Borkowski, 1987; Shu, 2010 and Lin, 1995) and *Computational-Intelligence* algorithms (i.e., ABC, JDE, JADE, SADE, EPSDE, GSA, PSO2011, and CMA–ES). The statistical tests realized for the comparison of performances indicate that the problem-solving success of DS algorithm in transforming the geocentric cartesian coordinates into geodetic coordinates is higher than those of all classical methods and *Computational-Intelligence* algorithms used in this paper.

1. Introduction

Transformation of the geocentric cartesian coordinates (X, Y, Z) into geodetic coordinates (φ, λ, h) or vice versa is necessary for solution of many aerospace, geodetic, cartographical, geo-astronomical and horizontal datum problems (Torge, 2001; Vanieek and Krakiwsky, 1986; Vermeille, 2011; Li et al., 2010; Kutoğlu, 2009; Vermeille, 2001; Fukushima, 1999; Seemkooei, 2002; Vermeille, 2004; Bowring, 1976, 1985; Feltens, 2008). Although transforming the geodetic coordinates into geocentric cartesian coordinates is an easily solved problem, it is rather difficult to solve the problem of transforming geocentric cartesian coordinates into geodetic coordinates as it is very hard to define a mathematical relationship between the geodetic latitude (φ) and the geocentric cartesian coordinates (Zhu, 1993, 1994; Borkowski, 1989; Bowring, 1976; Fukushima, 2006; Heikkinen, 1982; Jones, 2002; Zhang et al., 2005; Borkowski, 1987; Shu and Li, 2010; Lin and Wang, 1995). Generally, the methods developed to solve the problem of transforming the geocentric cartesian coordinates into geodetic coordinates can be categorized in two basic groups: iterative solution seeking methods (Borkowski, 1989; Fukushima, 1999; Feltens, 2008; Shu and Li, 2010; Pollard, 2005) and non-iterative solution seeking methods (Zhu, 1993, 1994; Vermeille, 2001, 2004, 2011; Li et al., 2010; Kutoğlu, 2009; Lin and Wang, 1995; Borkowski, 1987; Seemkooei, 2002).

The geodetic coordinates (φ, λ, h) can be transformed into geocentric cartesian coordinates (X, Y, Z) easily using the equations given in Eq.(1);

$$X=(N+h)\cdot\cos\varphi\cdot\cos\lambda$$
$$Y=(N+h)\cdot\cos\varphi\cdot\sin\lambda$$
$$Z=[N\cdot(1-e^2)+h]\cdot\sin\varphi \tag{1}$$

where φ, λ and h show the values of geodetic latitude, geodetic longitude, and geodetic height, respectively. N is defined as,

$$N=\frac{a}{\sqrt{1-e^2\sin^2\varphi}} \tag{2}$$

$$e=\frac{\sqrt{a^2-b^2}}{a} \tag{3}$$

where a and b are the semi-major axis and semi-minor axis of geodetic ellipsoid, respectively and e is the first eccentricity.

The geodetic longitude defined by Eq. (1) can be acquired using Eq. (4);

$$\lambda=\tan^{-1}\frac{Y}{X} \tag{4}$$

Then, the problem of transforming the geocentric cartesian coordinates into geodetic coordinates can be transformed into a

* Tel.: +90 352 4374901 x41054; fax: +90 352 4375744.
E-mail address: civici@erciyes.edu.tr

0098-3004/$ - see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cageo.2011.12.011

problem expressed with Eq. (5) and (6);

$$\varepsilon = \begin{bmatrix} (N+h)\cdot\cos\varphi\cdot\cos\lambda - X \\ (N+h)\cdot\cos\varphi\cdot\sin\lambda - Y \\ [N\cdot(1-e^2)+h]\cdot\sin\varphi - Z \end{bmatrix} \tag{5}$$

$$\underset{\varphi,h}{\text{argmin}}\ |\varepsilon|_{l_2} \tag{6}$$

In this paper, the problem of transforming the geocentric cartesian coordinates into geodetic coordinates is solved by the proposed *Differential Search Algorithm* (DS), by the classical methods (i.e., Borkowski, 1989; Bowring, 1976; Fukushima, 2006; Heikkinen, 1982; Jones, 2002; Zhang et al., 2005; Borkowski, 1987; Shu and Li, 2010; Lin and Wang, 1995) and by *Computational-Intelligence* algorithms (i.e., Artificial Bee Colony (ABC); Karaboğa and Baştürk, 2007, 2008), Self-Adaptive Differential Evolution Algorithm (JDE) (Feng et al., 2008; Brest et al., 2007, 2009), Adaptive Differential Evolution Algorithm (JADE) (Zhang and Sanderson, 2009), Strategy Adaptation based Differential Evolution Algorithm (SADE) (Qin and Suganthan 2005; Brest et al., 2007), Differential Evolution Algorithm with Ensemble of Parameters (EPSDE) (Mallipeddi et al., 2011), Gravitational Search Algorithm (GSA) (Rashedi et al., 2009), Particle Swarm Optimization Algorithm (PSO2011) (Clerc and Kennedy, 2002; Eberhart and Shi, 2001; Omran and Clerc, 2011), Covariance Matrix Adaptation Evolution Strategy (CMA-ES) (Hansen et al., 2009) through use of a set of test data containing 100,000 test points and the results obtained are examined in detail.

For transformation of the geocentric cartesian coordinates into geodetic coordinates through use of global-search algorithms, the objective function given in Eq. (6) has been used.

This paper is organized as follows. In Section 2, a variety of description is made about *Computational-Intelligence* algorithms mentioned in this paper. In Section 3, Differential Search Algorithm is introduced. Section 4 deals with Statistical Analysis Method and Section 5 with Experiments. In Section 6, the Results are presented.

2. Computational-intelligence algorithms

Computational-Intelligence means search of solution for a problem by natural or artificial agents through use of collective methods. As new and more complex numerical optimization problems to solve arise due to the rapid development of technology, we still need development of new optimization algorithms (Caponio et al., 2007, 2009; Das et al., 2005, 2009; Epitropakis et al., 2011; Fan and Lampinen, 2003a,2003b; Mininno et al., 2011; Neri et al., 2011; Neri and Tirronen, 2010; Neri and Mininno, 2010; Noman and Iba, 2008; Olorunda and Engelbrecht, 2007; Salvatore et al., 2010; Tirronen et al., 2008; Weber et al., 2010; Wang et al., 2011).

Inspired by the individual or social behaviors the living beings develop to solve a certain problem, *Computational-Intelligence* based optimization algorithms are quite successful in solving numerical optimization problems (e.g., *Ant-Colony Optimization* Algorithm (Ant Colony) (Dorigo et al., 1996), ABC, PSO, *Cuckoo-Search* Algorithm (Yang and Deb, 2010; Civicioglu and Besdok, 2011). Most *Computational-Intelligence* based global search techniques are technically based on a *social swarm* model. In nature, many species of living beings benefit from the ability of collective problem-solving of the groups they form through socialization to solve hard problems such as accommodation, reproduction, care of offspring, finding food and defense.

Socialization behaviors observed with the living beings vary radically; although the presocial living beings have basic social functions such as collective living and reproduction mechanism, they do not bring up the offspring together. Most of the living beings have adopted presocial style of life. In nature, only subsocial types of living beings take care of their offspring. And parasocial living beings live alone most of their life.

In the eusocial groups, there is, generally, a very powerful and complex hierarchy determining position and duty of the individuals in the society. Most eusocial living beings produce a phenomenon triggering reproduction or communication. In eusocial living beings, the reproduction is charged with only for this purpose and it is most of the time realized by the isolated (*sterilized*) individuals (i.e. *queen* of the colony). Ant Colony numerical optimization algorithm has been established on the phenomenon the ants use for communication purpose.

Ants, bees and termites, the most known eusocial–subsocial living beings, are reproduced from a single *queen* their colonies have. However, each individual thus reproduced has its own specific genetic characteristics. Genetic diversity in a society is the result of basic genetic processes such as *crossover*, *mutation* and *selection*. Genetic Algorithm and its derivatives solve numerical optimization problems based on some basic genetic rules such as *crossover*, *mutation*, *adaptation* and *selection*. There are a number of metaheuristic optimization algorithms based on basic genetic rules (e.g., Differential Evolution Algorithm (DE) (Storn, 1999; Storn and Price, 1997; Price and Storn, 1997; Qin et al., 2009a,2009b; Civicioglu, 2009) and its derivatives; JDE , JADE, EPSDE, SADE).

In general, eusocial living beings live in social colonies (i.e. *superorganism*) containing great number of individuals. This situation significantly improves ability of the *superorganism* to solve problems and overcome the difficulties.

With a population consisting of random solutions of the numerical optimization problem, the Computational-Swarm Intelligence techniques simulate the *superorganism* behavior observed with the eusocial living beings. In general, the concept *superorganism* is used for description of the concept *distributed intelligence*, which is associated with the swarm intelligence.

In order to develop artificial social evolution models, many researchers have benefited from the models, excessively simplified by abstraction of the natural behaviors such as food search (e.g. ABC), communication (e.g. Ant Colony), common care of the offspring (e.g. Cuckoo-Search) and formation of superorganism (e.g. PSO2011). For this reason, there may be radical differences between the theoretical behaviors of the living beings underlying a computational-swarm algorithm and the real behaviors of the living beings in the nature.

Computational-swarm models used in the *Computational-Intelligence* based optimization algorithms are very useful tools to understand and describe functionality of some global search methods, essentially based on *random-walk* algorithms (Civicioglu and Besdok, 2011).

Level of genetic *diversity* in the groups of living beings is very important for birth of individuals with different characteristics. Genetic Algorithm essentially simulates the evolution mechanism based on exchange of genes in the living beings. While wandering between the food source and colony, the ants mark the way they follow with pheromone, a type of chemical agent. As it is, we may think that the way containing pheromone more intensively is the way going to the food source containing more food. Ant Colony optimization algorithm is inspired by the strategies of the ants to find the shortest way between the food source and colony. DE optimization algorithm follows a strategy based on essential genetic rules such as *mutation*, *crossover* and *selection*. In general, it is considered that the living beings have an individual memory mechanism, which ensures the individuals to benefit from personal experience. *Social memory* is necessary in order that the best solution obtainable by the social community out of the random solutions of a problem can be benefitted by those individuals that have not reached to that solution yet. Standard-PSO (Eberhart and Shi, 2001) is based on search of the solution for a problem by taking advantage of the individual and social memories of the agents that constitute the artificial *superorganism*. In eusocial bee colonies, each bee has a hierarchical duty; only the *queen* bee reproduces and the worker bees have the duty to search food in the nature and bring the food to the colony. In general, it is considered that the honey bees can communicate other bees the direction and distance of the food source they discover. ABC algorithm is based on the assumption that the bees share the information about the places of the nectar sources they find with other bees on possibility basis.

In scientific literature, there are many studies using the algorithms ABC, JDE, JADE, SADE, EPSDE, GSA, PSO2011 and CMA-ES. For this reason, the success of the DS algorithm in solving numerical optimization problems is compared with those of the ABC, JDE, JADE, SADE, EPSDE, GSA, PSO2011 and CMA-ES algorithms.

In the following descriptions, concentration is given only on the general characteristics of the related algorithms and the idea of computational-swarm on which they are based (as in Wang et al., 2011 and Zhang and Sanderson, 2009).

2.1. Artificial Bee Colony algorithm (ABC)

ABC algorithm is a *computational-swarm* algorithm proposed for solution of the numerical optimization problems (Karaboğa and Baştürk, 2007, 2008), which has been successful in solving many different types of problems. In the ABC algorithm, it is assumed that a random solution for a problem corresponds to a nectar source and the global minimizer of the respective problem is searched by the artificial-bees, assumedly flying to find food in the region remaining between the sources of nectar. The calculation process used in the ABC algorithm is consisted of two basic stages: stage of *Employed-Bee* and stage of *Scout-Bee*. ABC is inclined to search more the sites providing better solution in the search space of the respective problem and this situation allows the ABC algorithm to quite effectively search the search-space. Crossing the search space limits during such searches, the *artificial-bees* are carried to the search-space limits nearest to them. If a nectar source is not sufficiently rich, it is abandoned after a certain attempt and a new nectar source, randomly produced, is added to the population instead of that location. Control parameters of the ABC algorithm are *limit* value and number of *employed-bees* (Karaboğa and Baştürk, 2007,2008). Although the mutation strategy used in the ABC algorithm likens the mutation strategy used in the DE algorithm, no crossover strategy is used in the ABC algorithm. Structure and problem-solving success of the ABC algorithm has been studied by Karaboğa and Baştürk (2007, 2008) and Civicioglu and Besdok (2011) in detail.

2.2. Self-adaptive differential evolution algorithm (JDE)

Values of the *mutation coefficient* (F) and *crossover coefficient* (CR), the basic control parameters of the DE algorithm, usually vary depending on the structure of the problem to be solved. In practice, determining the most-suitable initial values of F and CR is a time-consuming work based on trial and error. The JDE algorithm is based on determination of F and CR values adaptively (Feng et al., 2008; Brest et al., 2007, 2009). The JDE algorithm has been obtained by development of *DE/rand/1/bin* mutation strategy of the *DE* algorithm (Storn, 1999; Storn and Price, 1997; Price and Storn, 1997). JDE works with a fixed size of population and uses separate F and CR values for each element of the random solutions constituting the population. In JDE, the initial values of F and CR have been selected as 0.5 and 0.90. In the calculation process of JDE algorithm, the F and CR values are constantly updated according to a decision rule related to τ_1 and τ_2 threshold values; new values are constantly produced in the range of [0.1 1] for F and in the range of [0 1] for CR, depending on the uniform distribution.

Although the structure of the JDE algorithm is very simple; problem-solving success is higher than the DE/rand/1/bin algorithm. Problem-solving success of the JDE algorithm has been studied in detail by Feng et al. (2008) and Wang et al. (2011).

2.3. Adaptive differential evolution algorithm (JADE)

JADE algorithm contains a new mutation strategy (i.e., *DE/current-to-pbest*), which has been developed to use together with the DE algorithm (Zhang and Sanderson, 2009). The mutation operator used in JADE algorithm forces a randomly selected individual to evolve toward one of 100p% individuals providing the best solution in the population at that moment. In JADE algorithm, mutation coefficient is produced with $F \sim N(\mu_c, 0.1)$ and crossover coefficient with $CR \sim Cauchy(\mu_F, 0.1)$. Here $N(\mu_c, 0.1)$ is a normal distribution function with the average value of μ_c and standard deviation value of 0.1. Similarly, $Cauchy(\mu_F, 0.1)$ is a cauchy distribution function with local parameter value of μ_F and scale parameter value of 0.1. JADE algorithm can solve numerical optimization problems much more successfully than *DE/current-to-best/1* and *DE/best/1* strategies used in the standard differential evolution algorithm.

Problem-solving success of JADE algorithm has been studied by Zhang and Sanderson (2009) and Wang et al. (2011) in detail.

2.4. Strategy adaptation based differential evolution algorithm (SADE)

SADE algorithm is based on adaptive use of the mutation strategies, which are used in the DE algorithm (Qin and Suganthan, 2005; Brest et al., 2007). Mutation coefficient is obtained by $F \sim N(0.5, 0.3)$ with F value forced to remain in the range of (0 2]. Which mutation strategy shall be used at any moment is determined by a probability value calculated depending on the historical success of the respective mutation strategies in the iteration steps. In SADE algorithm, the crossover coefficient is produced with $CR \sim N(CR_m, 0.1)$. While CR value obtained is used throughout the next five iterations without change, F value constantly changes. For CR, initial value has been selected as 0.50. Local and global ability of SADE algorithm is highly developed.

Detailed analyses on problem-solving success of SADE algorithm are given in Qin and Suganthan (2005), Brest et al. (2007) and Wang et al. (2011).

2.5. Differential evolution algorithm with ensemble of parameters (EPSDE)

In the standard DE algorithm, there are five different mutation strategies and two different crossover strategies (Storn, 1999; Storn and Price, 1997; Price and Storn, 1997). As a result, the problem-solving success of the DE algorithm depends on the strategies selected for *mutation* and *crossover* and on the predetermined values of the mutation and crossover coefficients. EPSDE algorithm uses the mutation strategies existing in the DE algorithm (Mallipeddi et al., 2011). In EPSDE algorithm, a mutation strategy existing in DE algorithm is assigned to each element of the population. F value of the *mutation coefficient* to be used for the mutation strategy selected in EPSDE algorithm is randomly selected from a pool in such a way that it will remain between 0.4 and 0.9 in each iteration. The pool containing F values contains all values starting from 0.4 to 0.9 by increments of 0.1. The crossover coefficient to be used by a mutation strategy selected in a similar way as randomly selected from a pool in such a way that the CR value shall remain between 0.1 and 0.9 in each iteration.

The pool containing CR values contains all values starting from 0.1 to 0.9 by increments of 0.1. The mutation strategy, which is successful in iteration, and F and CR values used by that mutation strategy maintain themselves in the next iteration. Otherwise, the mutation operator selected for the respective population element is randomly changed. The initial values of each mutation operator selected new are also randomly selected from the respective pool.

Detailed study results concerning structure of EPSDE algorithm and its success in solving problems are given in Mallipeddi et al. (2011) and Wang et al. (2011).

2.6. Gravitational search algorithm (GSA)

GSA algorithm has been developed for solving the real-value numerical optimization problems. GSA algorithm has been inspired by the universal gravitational laws (Rashedi et al., 2009). Random solution of the respective problem desired to be solved in GSA have been modeled as artificial-bodies that apply gravitational force to each other. Mass of an artificial-body is related to the quality of the solution that artificial-body provides for the respective problem. Higher the quality of the solution, slower the speed that artificial-body abandons that position due to the gravitation force applied to it by other artificial-bodies. Speed of the artificial-bodies with inferior quality of solution is higher in the search-space. This phenomenon allows GSA to search the search space very efficiently to find a solution for a problem.

2.7. Particle swarm optimization algorithm (PSO2011)

PSO algorithm simulates choreographic movements of the groups of living beings moving as a *superorganism* (e.g. school of fish swimming together, team of birds flying together) (Clerc and Kennedy, 2002; Eberhart and Shi, 2001). In PSO algorithm, each random solution of the problem to be solved corresponds to an artificial particle moving together with the homogenous one in the search space. The concept *particle* corresponds to the concept *chromosome* used in the genetic algorithm. Unlike the behaviors of the chromosomes used in the genetic algorithm, the particles used in PSO algorithm benefit from their own individual experience and social experience of the swarm. It is supposed that this situation simulates communication among the living beings moving as a *superorganism*. It is considered that the individual experience of the particles and social experience of the swarm in PSO algorithm corresponds to the concepts of *local search* and *global search* in the global-search algorithms. There are a number of PSO structures, which are introduced in the scientific literature and much more successful than standard-PSO structure (e.g., Liang et al., 2006).

In this paper, PSO2011 algorithm is used to a very important extent, which contains developments gained as a result of studies on PSO algorithm that have lasted for many years (Omran and Clerc, 2011). PSO2011 algorithm has a structure much more complex than the standard-PSO algorithm (Clerc and Kennedy, 2002; Trelea, 2003; Civicioglu and Besdok, 2011; Liang et al., 2006); however, its success in problem-solving is also very high.

2.8. Covariance matrix adaptation evolution strategy (CMA-ES)

CMA-ES algorithm is a population-based evolutionary-search algorithm (Hansen et al., 2009, Zhang, 2011). CMA-ES has been used for solution of many *unconstrained* or *bounded* constraint optimization problems. In CMA-ES algorithm, the elements of parents and offspring are produced by using multivariate Gaussian distribution. The best offspring become the next parents. Functioning of CMA-ES is simple; in the *first step*, random solutions are produced by using multivariate Gaussian distribution; in the *second step*, the solutions thus produced are ordered again by their objective function value and in the *third step* the covariance matrix of the related multivariate Gaussian distribution is modified to achieve better solutions.

3. Differential search algorithm (DS)

DS is an algorithm developed for solution of optimization problems. DS algorithm simulates the *Brownian-like random-walk* movement used by an organism to migrate.

Capacity and efficiency of the food areas existing in the nature (i.e. pastures, water supplies) often vary due to the periodical *climatic changes* during the year. For this reason, many species of the living beings, *eusocial*, *subsocial* or *presocial*, show seasonal migration behavior throughout the year. Migration behavior allows the living beings to move from a habitat where capacity and diversity of natural sources reduce to a more efficient habitat. In nature, many species of living beings have a periodical cycle of immigration (e.g. many species of birds (Sekercioğlu, 2007), monarch butterflies, fire ants, honeybees, whales). In the migration movement, the migrating species of living beings constitute a *superorganism* containing large number of individuals. Then the *superorganism* starts to change its position by moving toward more fruitful areas. Movement of a *superorganism* can be described by a *Bownian-like random-walk* model (Vito et al., 2011).

There are a number of *Computational-Intelligence* algorithms that model the behaviors of the *superorganisms* (e.g., PSO, Cuckoo-Search, ABC, Ant Colony). Many species of predatory living beings control fertility of the site to which they desire to migrate. If the potential of a site controlled for an intention of migration can meet needs of the *superorganism* at that moment, *superorganism* settles in the new site at least for a time and continues its migration by repeating its movement to find more fertile areas.

It is assumed, in DS algorithm, that a population made up of random solutions of the respective problem corresponds to an artificial-*superorganism* migrating. In DS algorithm, artificial-*superorganism* migrates to global minimum value of the problem. During this migration, the artificial-*superorganism* tests whether some randomly selected positions are suitable temporarily during the migration. If such a position tested is suitable to stop over for a temporary time during the migration, the members of the artificial-*superorganism* (i.e. artificial-organisms) that made such discovery immediately settle at the discovered position and continue their migration from this position on.

Pseudo-code indicating the function of DS algorithm is given in Fig. 1.

In DS algorithm, artificial-organisms (i.e., X_i, $i=\{1,2,3,\ldots,N\}$) making up an artificial-organism (i.e., $Superorganism_g$, $g=1,2,3,\ldots$, maxgeneration) contain members as much as the size of the problem (i.e., $x_{i,j}$, $j=\{1,2,3,\ldots,D\}$) . Here, N signifies number of elements in the *superorganism* and D indicates size of the respective problem.

In DS algorithm, a member of an artificial-organism in initial position is defined by using Eq. (7).

$$x_{i,j} = \mathrm{rand}\cdot(\mathrm{up}_j - \mathrm{low}_j) + \mathrm{low}_j \tag{7}$$

In such case, the artificial-organisms are defined by $X_i=[x_{i,j}]$ and the artificial-*superorganism* made up of the artificial-organisms is indicated by $Superorganism_g=[X_i]$.

In DS algorithm, the mechanism of finding a *stopover site* at the areas remaining between the artificial-organisms may be described by a *Brownian-like random walk* model (Vito et al., 2011). Randomly selected individuals of the artificial-organisms move towards the targets of donor$=[X_{\mathrm{random_shuffling}(i)}]$ in order to discover *stopover sites*, which are very important for a successful

Algorithm : Differential Search Algorithm

Require:

N: *The size of the population, where* $i=\{1,2,3,\ldots,N\}$.

D: *The dimension of the problem.*

G: *Number of maximum generation.*

1: $Superorganism = initialize()$, where $Superorganism = \left[ArtificialOrganism_i\right]$

2: $y_i = Evaluate\left(ArtificialOrganism_i\right)$

3: **for** cycle=1: G **do**

4: $donor = Superorganism_{Random\_Shuffling(i)}$

5: $Scale = randg[2\cdot rand_1]\cdot(rand_2 - rand_3)$

6: $StopoverSite = Superorganism + Scale\cdot(donor - Superorganism)$

7: $p_1 = 0.3\cdot rand_4$ and $p_2 = 0.3\cdot rand_5$

8: **if** $rand_6 < rand_7$ **then**

9: **if** $rand_8 < p_1$ **then**

10: $r = rand(N,D)$

11: **for** *Counter1*=1:N **do**

12: $r(Counter1,:) = r(Counter1,:) < rand_9$

13: **endfor**

14: **else**

15: $r = ones(N,D)$

16: **for** *Counter2=1:N* **do**

17: $r(Counter2,randi(D)) = r(Counter2,randi(D)) < rand_{10}$

18: **endfor**

19: **endif**

20: **else**

21: r=ones(*N,D*)

22: **for** *Counter3=1:N* **do**

23: $d = randi(D,1,\lceil p_2\cdot rand\cdot D\rceil)$

24: **for** *Counter4=1:size(d)* **do**

25: $r(Counter3,d(Counter4)) = 0$

26: **endfor**

27: **endfor**

28: **endif**

29: $individuals_{I,J} \leftarrow r_{I,J} > 0 \;|\; I\in i,\, J\in[1\ D]$

30: $StopoverSite(individuals_{I,J}) := Superorganism(individuals_{I,J})$

31: **if** $StopoverSite_{i,j} < low_{i,j}$ *or* $StopoverSite_{i,j} > up_{i,j}$ **then**

32: $StopoverSite_{i,j} := rand\cdot(up_j - low_j) + low_j$

33: **endif**

34: $y_{StopoverSite;i} = evaluate(StopoverSite_i)$

35: $y_{Superorganism;i} := \begin{cases} y_{StopoverSite;i} & If\ \ y_{StopoverSite;i} < y_{Superorganism;i} \\ y_{Superorganism;i} & else \end{cases}$

36: $ArtificialOrganism_i := \begin{cases} StopoverSite_i & If\ \ y_{StopoverSite;i} < y_{Superorganism;i} \\ ArtificialOrganism_i & else \end{cases}$

37: **endfor**

Fig. 1. Pseudo-code of the proposed DS algorithm.

migration (random_shuffling function of Fig. 1 randomly changes the order of the numbers of the elements in the set of $i=\{1,2,3,\ldots,N\}$). Size of the change occurred in the positions of the members of the artificial-organisms is controlled by the *scale* value.

In DS algorithm, the *scale* value is produced by using a gamma-random number generator (i.e., *randg*) controlled by a uniform-random number generator (i.e., rand) working in the range of [0 1] together. The structure used for calculation of the *scale* value (*please see* Fig. 1, line 5) allows the respective artificial-*superorganism* to radically change direction in the habitat.

In DS, a *stopover site* position is produced by using Eq.(8);

$$StopoverSite = Superorganism + Scale \times (donor\text{-}Superorganism) \qquad (8)$$

In DS, the members (i.e. individuals) of the artificial organisms of the *superorganism* to participate in the search process of *stopover site* are determined by a random process. Structure of the respective random process is given in Fig. 1, lines 8–29. In DS algorithm, if one of the elements of *stopover site* is, for one reason, goes beyond the limits of the habitat (i.e. *search space*), the said element is randomly deferred to another position in the habitat (*please see* Fig. 1, lines 31–33).

If, in DS algorithm, a *stopover site* is more fertile than the sources owned by the artificial-organism of which the individuals that discover that *stopover site*, that artificial-organism moves to that *stopover site*. While the artificial-organisms change site, the *superorganism* containing the artificial organisms continues its migration towards the global minimum.

Unlike the algorithms DE, JDE, JADE, EPSDE, SADE and ABC; DS algorithm may simultaneously use more than one individual. In contrary to the algorithms *DE/best/1*, JADE, PSO and ABC; DS algorithm has no inclination to correctly go towards the *best* possible solution of the problem. For this reason, it has a successful search strategy for solution of multimodal functions.

DE algorithm has only two control parameters (i.e., p_1 and p_2) (*please see* Fig. 1, line 7). Detailed tests were conducted to determine the most appropriate value for p_1 and p_2 and it has been seen that the values $p_1 = 0.3 \cdot rand$ and $p_2 = 0.3 \cdot rand$ provide the best solutions for the respective problems. With respect to all benchmark functions used in this paper, DS is not much sensitive to the value of p_1 and p_2. Algorithmic structure of DS is very simple. This situation allows easy application of DS for different engineering problems.

Software code of DS algorithm can be found in (Civicioglu, 2011).

Table 1
Benchmark functions used at Test Set-1.

Benchmark function (FNC)		Dim	Low	Up
*F*1	Goldsteinprice	2	−2	2
*F*2	Ackley	30	−32	32
*F*3	Beale	5	−4.5	4.5
*F*4	Bohachecsky1	2	−100	100
*F*5	Bohachecsky2	2	−100	100
*F*6	Bohachecsky3	2	−100	100
*F*7	Booth	2	−10	10
*F*8	Branin	2	−5	10
*F*9	Colville	4	−10	10
*F*10	Easom	2	−100	100
*F*11	Fletcher	5	−pi	Pi
*F*12	Griewank	30	−600	600
*F*13	Hartman3	3	0	1
*F*14	Langermann	2	0	10
*F*15	Matyas	2	−10	10
*F*16	Michalewics	2	0	Pi
*F*17	Michalewics	5	0	Pi
*F*18	Michalewics	10	0	Pi
*F*19	Perm	4	−4	4
*F*20	Powell	24	−4	5
*F*21	Powersum	4	0	4
*F*22	Quartic	30	−1.28	1.28
*F*23	Rastrigin	30	−5.12	5.12
*F*24	Rosenbrock	30	−30	30
*F*25	Schaffer	2	−100	100
*F*26	Schwefel	30	−500	500
*F*27	Schwefel_1_2	30	−100	100
*F*28	Schwefel_2_22	30	−10	10
*F*29	Shekel10	4	0	10
*F*30	Shekel5	4	0	10
*F*31	Shekel7	4	0	10
*F*32	Shubert	2	−10	10
*F*33	Camelback	2	−5	5
*F*34	Sphere2	30	−100	100
*F*35	Step2	30	−100	100
*F*36	Stepint	5	−5.12	5.12
*F*37	Sumsquares	30	−10	10
*F*38	Trid	6	−36	36
*F*39	Trid	10	−100	100
*F*40	Zakharov	10	−5	10

4. Statistical analysis method

For recognition of a new calculation method, it is recommended to compare success of that method for solution of problem with those of the widely-used calculation methods by use of statistical methods. For comparison of the problem-solving success of the *Computational-Intelligence* based optimization techniques by statistical methods, Wilcoxon Rank Sum Test, which is frequently employed in the literature for comparison of problem-solving success of the *Computational-Intelligence* algorithms, has been used (Yen and Leong, 2009; Derrac et al., 2011; Wang et al., 2011).

Because of probabilistic nature of the global search algorithms, a metaheuristic algorithm may find, by pure chance, a solution very different than any solutions found for any problem. As a result, in order to correctly compare success of the *Computational-Intelligence* algorithms by statistical tools, some corrections should be made with the values of level (α) or critical-value (p-value). There are many corrections developed by the researchers to use in the statistical tests (i.e., Duncan, Bonferroni, Sheffe, Fisher, Holm, and Tukey correction (Bratton and Kennedy, 2007). In this paper, Wilcoxon Rank Sum Test with Bonferroni–Holm correction method for significance level of $\alpha = 0.05$, has been used (Wenyin et al., 2011). The last rows of Tables 9 and 10 show the success of DS (proposed) in comparison with its competitors as $W/T/L$, which means that DS wins in W functions, ties in T functions, and loses in L functions (Wang et al., 2011).

5. Experiments

In this paper, two different tests have been performed. In the first test with its results given in Section 5.1, the problem-solving success of DS algorithm has been studied in detail by means of statistical methods. In the second test with its results given in Section 5.2, focus is on the solution of the problem for transforming the geocentric cartesian coordinates into geodetic coordinates.

Table 2
Benchmark functions used at Test Set-2.

Benchmark function (FNC)		Dim	Low	Up
*F*1	Shifted sphere function	10	−100	100
*F*2	Shifted Schwefel's problem	10	−100	100
*F*3	Shifted rotated high Conditioned elliptic function	10	−100	100
*F*4	Shifted Schwefel's problem 1.2 with noise in fitness	10	−100	100
*F*5	Schwefel's problem 2.6 with global optimum on bounds	10	−100	100
*F*6	Shifted Rosenbrock's function	10	−100	100
*F*7	Shifted rotated Griewank's function without bounds	10	0	600
*F*8	Shifted rotated Ackley's function with global optimum on bounds	10	−32	32
*F*9	Shifted Rastrigin's function	10	−5	5
*F*10	Shifted rotated Rastrigin's function	10	−5	5
*F*11	Shifted rotated Weierstrass function	10	−0.5	0.5
*F*12	Schwefel's problem 2.13	10	$-\pi$	π

5.1. Experiments-1

In the tests conducted in this section, success of DS algorithm introduced in this paper has been compared with the success of the algorithms ABC, JDE, JADE, SADE, EPSDE, GSA, PSO2011 and CMA-ES for solution of numerical optimization problems by using two different tests. In each test conducted, a benchmark function has been solved 30 times by using a different starting population. The same starting population has been used for each algorithm during the tests. Data gained as a result of the tests performed such as the global minimum value, the best function evaluation number value and the best runtime value have been saved for statistical analyses. All tests were performed by using Matlab.

Arihtmetic-precision level to be used for comparison of the problem-solving success of two algorithms is very important for realistic comparison of the problem-solving success of the respective algorithms. Especially, for many practical applications with which the Geomatics and Astro-Geodesy engineering is involved, it is sufficient that arithmetic-precision is in the range of 10^{-12} and 10^{-18}. For this purpose, the arithmetic-precision level has been determined as 10^{-18} in this paper.

Initial setting values of the algorithmic control parameters of the *Computational-Intelligence* methods used in this paper are given below:

1. Artificial Bee Colony Algorithm (ABC); $limit = N \cdot D$, number of *employed bees* = size of population/2.
2. Self-Adaptive Differential Evolution Algorithm (JDE); $F_{\text{inital}} = 0.5$, $CR_{\text{initial}} = 0.90$, $\tau_1 = \tau_2 = 0.1$.
3. Adaptive Differential Evolution Algorithm (JADE); $F \sim N(\mu_c, 0.1)$, $\text{CR} \sim \text{Cauchy}(\mu_F, 0.1)$, $c = 0.1$, $p = 0.05$, *mutation strategies* and *crossover strategies* as in Zhang and Sanderson, (2009).
4. Strategy Adaptation based Differential Evolution Algorithm (SADE); $F \sim N(0.5, 0.3)$, $CR \sim N(CR_m, 0.1)$, *mutation strategies* and *crossover strategies* as in Qin and Suganthan, (2005).
5. Differential Evolution Algorithm with Ensemble of Parameters (EPSDE); $F \in \{0.4, 0.5, 0.6, 0.7, 0.8, 0.9\}$, $CR \in \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9\}$, *mutation strategies* and *crossover strategies* as in Mallipeddi et al., (2011).
6. Gravitational Search Algorithm (GSA); Rnorm = 2, Rpower = 1, alpha = 20, G0 = 100, finalper = 2 as in Rashedi et al. (2009) and the software code given in Rashedi, (2011).
7. Particle Swarm Optimization Algorithm (PSO2011); $w = 1/(2 \cdot \log(2))$, $c_1 = 0.5 + \log(2)$, $c_2 = c_1$ as in Omran and Clerc (2011).
8. CMA-ES; $\sigma = 0.25$, $\mu = \left\lfloor \frac{4 + \lfloor 2 \cdot \log(N) \rfloor}{2} \right\rfloor$ as in Zhang (2011).

Common control parameters used for *Computational-Intelligence* algorithms in this paper are given below:

1. Size of population = 30.
2. Maximum number of Function Evalution value = 2,000,000.

Stop-criteria used in this paper to stop the calculation processes of the *Computational-Intelligence* algorithms are as follows:

1. Stop if the number of function evaluation number during the calculations reaches to 2,000,000.

Table 3
Mean global minimum values that are obtained by ABC, JDE and JADE algorithms for the benchmark functions used at Test Set-1.

FNC	ABC	JDE	JADE
*F*1	3.000000046542302000	2.999999999999920500	2.999999999999919600
*F*2	0.000000000000034046	0.081101705642285957	0.000000000000005861
*F*3	0.000000000000002760	0.000000000000000000	0.000000000000000000
*F*4	0.000000000000000000	0.000000000000000000	0.000000000000000000
*F*5	0.000000000000000000	0.000000000000000000	0.000000000000000000
*F*6	0.000000000000000566	0.000000000000000000	0.000000000000000000
*F*7	0.000000000000000002	0.000000000000000000	0.000000000000000000
*F*8	0.397887357729738160	0.397887357729738160	0.397887357729738160
*F*9	0.071567506072597015	0.000000000000000000	0.000000000000000000
*F*10	−0.999999999999999890	−0.999999999999999890	−0.999999999999999890
*F*11	0.021868849833187230	0.944372865543282590	66.095800643550064000
*F*12	0.000000000000000048	0.004819357854318506	0.005163255348051714
*F*13	−3.862782147820753100	−3.862782147820753100	−3.862782147820753100
*F*14	−1.080938442134438100	−1.076428076265744800	−1.080938442134438100
*F*15	0.000000000000000446	0.000000000000000000	0.000000000000000000
*F*16	−1.821043683677681300	−1.821043683677681300	−1.821043683677681300
*F*17	−4.693468451957112800	−4.689345693261705300	−4.692094199058645100
*F*18	−9.660151715641349700	−9.639723098613243800	−9.650961715549245300
*F*19	0.083844001403803228	0.015410513005585573	0.002190275113020903
*F*20	0.000260433001346215	0.000000000000000066	0.000000000000000000
*F*21	0.007790531109495759	0.002018511626149027	0.000000000000000000
*F*22	0.025016325252703038	0.001301031618067892	0.000576445259770008
*F*23	0.000000000000000000	1.127620264705736500	0.000000000000000000
*F*24	0.285683346590413220	1.063099694480250000	1.063099694480249100
*F*25	0.000000000000000005	0.003886363951413971	0.000000000000000000
*F*26	−12569.486618173018000000	−12304.974337534099000000	−12206.275725355399000000
*F*27	14.566873412694815000	0.000000000000000000	0.000000000000000000
*F*28	0.000000000000000489	0.000000000000000000	0.000000000000000000
*F*29	−10.536409816692048000	−10.313043716242612000	−10.357717714190578000
*F*30	−10.153199679058224000	−9.565613576121565500	−8.225519700776185800
*F*31	−10.402940566818662000	−9.161581335473734100	−9.925751118173623800
*F*32	−186.730908831023980000	−186.730908831023980000	−186.730908831023980000
*F*33	−1.031628453489877000	−1.031628453489877000	−1.031628453489877000
*F*34	0.000000000000000447	0.000000000000000000	0.000000000000000000
*F*35	0.000000000000000000	0.900000000000000020	0.299999999999999930
*F*36	0.000000000000000000	0.000000000000000000	0.000000000000000000
*F*37	0.000000000000000473	0.000000000000000000	0.000000000000000000
*F*38	−49.999999999999659000	−50.000000000000206000	−50.000000000000206000
*F*39	−209.999999999947050000	−210.000000000002670000	−210.000000000002870000
*F*40	0.000000040238042357	0.000000000000000000	0.000000000000000000

2. Stop if the algorithm could not obtain a better solution eventually at the end of 200,000 function evaluation number.
3. Stop if the obtained error of the algorithm is smaller than 10^{-18}.

For the statistical tests performed, H_0:Common Median$=0$ (Wang et al., 2011 ; Wenyin et al., 2011; Derrac et al., 2011). Statistical significance value necessary for test of H_0 null hypothesis has been selected as $\alpha=0.05$. If p-value produced in a test is smaller than the significance level (α), H_0 hypothesis was rejected for that test. For comparison of the problem-solving success of a new proposed calculation method with the widely used calculation methods, the Test Set to be used is very important. The related Test Set should contain benchmark functions of different characteristic and different size and be widely used. In this paper, two tests containing different benchmark functions were performed to study success of DS algorithm. In Test-1, 40 different benchmark functions, widely used for study of the problem-solving success of the optimization algorithms were used (Karaboğa and Baştürk, 2007, 2008; Civicioglu and Besdok, 2011). In Test-2, the first 12 functions of unimodal an multimodal characteristics out of those benchmark functions used in CEC2005 were used (Wang et al., 2011).

Sizes of the benchmark functions used in Test-1 vary between 2 and 30 as in Zhang and Sanderson (2009), Karaboğa and Baştürk (2007, 2008) and Civicioglu and Besdok (2011). And the sizes of the benchmark functions used in Test-2 have been assigned as 10 as in Liang et al. (2006), Mallipeddi et al. (2011), Qin et al., 2009a,2009b. Various characteristics for the benchmark functions used in Test-1 and Test-2 are given in Tables 1 and 2. Detailed information about the related benchmark functions can be found in Karaboğa and Baştürk (2007, 2008) and Suganthan et al. (2005).

Tables 3–8 show the average values of the global minimum values obtained from the tests conducted under Test-1 and Test-2.

According to the statistical test performed, the algorithms solving significantly better in Test-1 and Test-2 are given in Tables 9 and 10.

When the Tables 9 and 10 are examined, it may be said that DS algorithm is, in general, significantly more successful than the *Computational-Intelligence* algorithms used in this paper for obtaining global minimum values for the benchmark functions.

When the average problem-solving speed of the *Computational-Intelligence* algorithms used in this paper during Test-1 and Test-2 is examined, we get a sequence from the fastest algorithm down to the slowest algorithm for the Test-1 as: JADE: 14.2051 s, CMA-ES: 15.402 s, JDE : 15.7952 s, DS: 25.7573 s, ABC: 32.8765s, EPSDE: 45.9704 s, SADE: 48.1751 s, GSA: 68.3936 s, and PSO: 119.4698s; and for the Test-2 as CMA-ES : 21.5657 s, JDE: 92.7486 s, ABC : 117.0642 s, GSA: 160.0062 s, EPSDE : 279.2111 s, JADE : 294.0554 s, PSO: 615.7871 s, DS: 698.1123 s and SADE: 843.1972 s.

Original software code of ABC algorithm used in this paper has been taken from Karaboğa (2011) and the original software codes

Table 4
Mean global minimum values that are obtained by SADE, EPSDE and GSA algorithms for the benchmark functions used at Test Set-1.

FNC	SADE	EPSDE	GSA
*F*1	2.999999999999919600	2.999999999999919600	5.246682476710645000
*F*2	0.791536822033545700	0.000000000000008230	5.410906428881933300
*F*3	0.000000000000000000	0.000000000000000000	0.007094497560333099
*F*4	0.000000000000000000	0.000000000000000000	0.011340203506080908
*F*5	0.000000000000000000	0.000000000000000000	0.013944008187213777
*F*6	0.000000000000000000	0.000000000000000000	0.004599054836527167
*F*7	0.000000000000000000	0.000000000000000000	0.005047443323093178
*F*8	0.397887357729738160	0.397887357729738160	0.399994747000155360
*F*9	0.000000000000000000	0.000000000000000000	9.755221338320463800
*F*10	−0.999999999999999890	−0.999999999999999890	−0.990593655793831740
*F*11	0.000000000000000000	0.000000000000000000	1446.486510517173000000
*F*12	0.022635932696713858	0.005984493442299691	0.792376637112038870
*F*13	−3.862782147820753100	−3.862782147820753100	−3.154283037223189400
*F*14	−1.080938442134438100	−1.080938442134438100	−1.050731560478939800
*F*15	0.000000000000000000	0.000000000000000000	0.000173637984290100
*F*16	−1.821043683677681300	−1.821043683677681300	−1.473223260813489400
*F*17	−4.688496529998376500	−4.693468451957112800	−2.227787180767796500
*F*18	−9.657203823292166000	−9.659988010714895900	−2.959541370117288600
*F*19	0.014027206669065765	0.016561152301948403	1.628669230797524300
*F*20	0.000000273380673496	0.000000000000000003	1033.331934307631600000
*F*21	0.000000000000000000	0.001202559576024210	3.018121951673780100
*F*22	0.001673076840695268	0.000957009758396907	121.764732112178270000
*F*23	0.862297849480857300	0.000000000000009473	286.822129205027410000
*F*24	1.213737744700701200	1.063099694480259500	35024.211519923185000000
*F*25	0.000647727325167624	0.000000000000000000	0.004712680551606859
*F*26	−12549.746895737275000000	−12569.486618173018000000	−2825.480944675528900000
*F*27	0.000000000000000000	0.000000000000000000	609.342946736085880000
*F*28	0.000000000000000000	0.000000000000000000	45.310294835234103000
*F*29	−10.536409816692050000	−10.357717714190578000	−1.266646916745724600
*F*30	−9.984785427767349100	−9.397199324830689100	−0.703053590378170060
*F*31	−10.402940566818662000	−10.402940566818662000	−0.897001772936718460
*F*32	−186.730908831023980000	−186.730908831023980000	−186.599328641993170000
*F*33	−1.031628453489877000	−1.031628453489877000	−0.995569562716766890
*F*34	0.000000000000000000	0.000000000000000000	31.396632889051457000
*F*35	0.000000000000000000	0.433333333333333350	35.100000000000001000
*F*36	0.000000000000000000	0.000000000000000000	0.000000000000000000
*F*37	0.000000000000000000	0.000000000000000000	337.293379893073560000
*F*38	−50.000000000000213000	−50.000000000000206000	−49.130540760698629000
*F*39	−210.000000000003180000	−210.000000000002670000	−205.643863149859560000
*F*40	0.000000000000000000	0.000000000000000000	11.850234560468328000

Table 5
Mean global minimum values that are obtained by PSO2011, CMA-ES and DS (proposed) algorithms for the benchmark functions used at Test Set-1.

FNC	PSO2011	CMA-ES	DS (proposed)
*F*1	2.999999999999920500	10.913101554846248000	2.999999999999919200
*F*2	1.521432297372501200	19.473295433416929000	0.000000000000021020
*F*3	0.000000004192296805	0.567121493926453280	0.000000000000000000
*F*4	0.000000000000000000	0.032465192448694655	0.000000000000000000
*F*5	0.000000000000000000	0.027877497684309614	0.000000000000000000
*F*6	0.000000000000000000	0.001853298662753064	0.000000000000000000
*F*7	0.000000000000000000	0.000000000000000000	0.000000000000000000
*F*8	0.397887357729738160	0.956323255792216780	0.397887357729738160
*F*9	0.000000000000000000	0.000000000000000000	0.000000000000000000
*F*10	−0.999999999999999890	−0.738499176740900260	−0.999999999999999890
*F*11	48.746516444692730000	787.984625997960280000	0.000000000000000000
*F*12	0.006894369481971326	0.001314714864809344	0.000000000000000000
*F*13	−3.862782147820753100	−3.767383305012159200	−3.862782147820753100
*F*14	−1.080938442134438100	−0.518112783232801300	−1.080938442134438100
*F*15	0.000000000000000000	0.000000000000000000	0.000000000000000000
*F*16	−1.821043683677681300	−1.725751531623896300	−1.821043683677681300
*F*17	−4.656564639705393900	−4.094025383053482500	−4.693468451957112800
*F*18	−8.971733030754931400	−7.727690007537370700	−9.660151715641349700
*F*19	0.011968722456044061	0.040574971211323960	0.001239881585104723
*F*20	0.000013071891200815	0.000000000000000000	0.000001041051900331
*F*21	0.000125488283423822	0.000000000000000000	0.000078214195963962
*F*22	0.000354834551317880	0.076689147188399684	0.006055501521948293
*F*23	25.636760225867558000	234.543612153901510000	0.000000000000001894
*F*24	2.675704311426969600	0.398662385430093020	0.265774923620062280
*F*25	0.000000000000000000	0.460765667233442480	0.000000000000000000
*F*26	−7684.610475778376900000	−7349.661835383788500000	−12569.486618173018000000
*F*27	0.000000000000000000	0.000000000000000000	0.000000000044391259
*F*28	0.000000000000000000	0.000000000000000000	0.000000000000000000
*F*29	−10.106187362165304000	−5.562428076840821400	−10.536409816692050000
*F*30	−9.537393808204546600	−6.064192913425393300	−10.153199679058224000
*F*31	−10.402940566818662000	−6.611026498592226900	−10.402940566818662000
*F*32	−186.730907356988470000	−69.527161536320449000	−186.730908831023980000
*F*33	−1.031628453489877000	−1.004422965853005300	−1.031628453489877000
*F*34	0.000000000000000000	0.000000000000000000	0.000000000000000000
*F*35	2.299999999999999800	0.000000000000000000	0.000000000000000000
*F*36	0.133333333333333330	0.400000000000000020	0.000000000000000000
*F*37	0.000000000000000000	0.000000000000000000	0.000000000000000000
*F*38	−50.000000000000206000	−50.000000000000206000	−50.000000000000206000
*F*39	−210.000000000001450000	−210.000000000002730000	−210.000000000002730000
*F*40	0.000000000000000000	0.000000000000000000	0.000000000000000000

Table 6
Mean global minimum values that are obtained by ABC, JDE and JADE algorithms for the benchmark functions used at Test Set-2.

FNC	ABC	JDE	JADE
*F*1	−450.000000000000000000	−450.000000000000000000	−450.000000000000000000
*F*2	−449.999999999972490000	−450.000000000000000000	−450.000000000000000000
*F*3	361158.738747538590000000	−449.999999999979880000	−449.999999999999940000
*F*4	219.541782151597540000	−450.000000000000000000	−450.000000000000000000
*F*5	−290.886226599789040000	−310.000000000000000000	−310.000000000000000000
*F*6	390.600793207119750000	390.398657911234690000	390.000000000000000000
*F*7	1087.045948628602400000	1087.045948628602700000	1087.045948628602400000
*F*8	−119.745178809993520000	−119.504691281187920000	−119.885061292223710000
*F*9	−330.000000000000000000	−329.801008188581310000	−330.000000000000000000
*F*10	−309.720279579249900000	−319.718763127128060000	−325.423188488130110000
*F*11	94.769571503209505000	92.759320222384559000	93.974605820462955000
*F*12	−336.798458483672280000	198.217391541843680000	−444.982275730185170000

Table 7
Mean global minimum values that are obtained by SADE, EPSDE and GSA algorithms for the benchmark functions used at Test Set-2.

FNC	SADE	EPSDE	GSA
*F*1	−450.000000000000000000	−450.000000000000000000	−443.912946663180780000
*F*2	−450.000000000000000000	−450.000000000000000000	−430.140840333899180000
*F*3	474.868432277688100000	−449.999999999939180000	240263.564064562930000000
*F*4	−450.000000000000000000	−449.999999999999940000	−426.996168381870920000
*F*5	−309.999999999998070000	−310.000000000000000000	−196.635253806207830000
*F*6	390.398657911235260000	390.531543881646260000	2598.313768245189300000
*F*7	1087.045948628602400000	1087.045948628602700000	1087.046340859325000000
*F*8	−119.780793108784120000	−119.734384534452700000	−119.714436635434590000
*F*9	−329.966834698096870000	−330.000000000000000000	−254.786026051873080000
*F*10	−323.035289455111130000	−324.889603179040420000	−245.659264482115250000
*F*11	91.913347917682145000	94.805303905259706000	99.186930607623182000
*F*12	−369.441385639203990000	−442.910162125352090000	10253.273528924188000000

Table 8
Mean global minimum values that are obtained by PSO2011, CMA-ES and DS (proposed) algorithms for the benchmark functions used at Test Set-2.

FNC	PSO2011	CMA-ES	DS (proposed)
$F1$	−450.000000000000000000	−450.000000000000000000	−450.000000000000000000
$F2$	−450.000000000000000000	−450.000000000000000000	−449.999999999999940000
$F3$	30.611110613833919000	−450.000000000000000000	−449.407565976078330000
$F4$	−450.000000000000000000	6431.382928720879100000	−449.999999999999940000
$F5$	−310.000000000000000000	−309.999999999989830000	−309.999999999994660000
$F6$	394.932336521969770000	390.132885970411560000	390.000000000000000000
$F7$	1090.277940916075100000	1087.045948628602700000	1087.045948628602700000
$F8$	−119.765824921577580000	−120.000000000000000000	−119.975358370182890000
$F9$	−324.583510152637070000	−227.586804923858550000	−330.000000000000000000
$F10$	−324.311291004546490000	−220.689633189852830000	−315.175130224857300000
$F11$	93.506018683034668000	93.231946627925510000	93.343475023652402000
$F12$	19498.572353538242000000	−131.401006931858920000	−457.537861815555800000

Table 9
For Test-1, evaluation of the success of DS (proposed) and the comparison algorithms according to the Wilcoxon Rank Sum Test analysis for significance level of $\alpha=0.05$ (W=the number of functions that DS wins, T=the number of functions that DS ties, L=the number of functions that DS loses when compared with its competitors).

FNC	DS–ABC	DS–JDE	DS–JADE	DS–SADE	DS–EPSDE	DS–GSA	DS–PSO2011	DS– CMA-ES
$F1$	DS	DS	JADE	DS	DS	DS	DS	DS
$F2$	DS	DS	DS	SADE	DS	DS	DS	DS
$F3$	DS	DS	DS	DS	DS	DS	DS	CMA-ES
$F4$	DS, ABC	JDE	DS, JADE	DS, SADE	EPSDE	DS	DS, PSO2011	CMA-ES
$F5$	DS, ABC	DS, JDE	DS, JADE	DS, SADE	DS, EPSDE	DS	DS, PSO2011	CMA-ES
$F6$	DS	DS, JDE	DS, JADE	DS, SADE	DS, EPSDE	DS	DS, PSO2011	CMA-ES
$F7$	DS	DS	DS	DS	DS	DS	DS	DS
$F8$	DS, ABC	DS, JDE	DS, JADE	DS, SADE	DS, EPSDE	DS	DS, PSO2011	CMA-ES
$F9$	DS	DS	DS	DS	DS	DS	DS	DS
$F10$	DS, ABC	DS, JDE	DS, JADE	DS, SADE	DS, EPSDE	DS	DS, PSO2011	DS
$F11$	DS	DS	JADE	DS	DS	DS	PSO2011	DS
$F12$	DS	DS	DS	DS	DS	DS	DS	CMA-ES
$F13$	DS	DS, JDE	DS, JADE	DS, SADE	DS, EPSDE	DS	DS, PSO2011	CMA-ES
$F14$	DS	JDE	DS, JADE	DS, SADE	DS, EPSDE	DS	DS	DS
$F15$	DS	DS	DS	DS	DS	DS	DS	DS
$F16$	DS, ABC	DS, JDE	DS, JADE	DS, SADE	DS, EPSDE	DS	DS, PSO2011	CMA-ES
$F17$	ABC	JDE	JADE	DS	EPSDE	DS	DS	DS
$F18$	ABC	JDE	JADE	DS	EPSDE	DS	DS	DS
$F19$	DS	DS	JADE	SADE	DS	DS	PSO2011	CMA-ES
$F20$	DS	DS	DS	SADE	DS	DS	DS	DS
$F21$	DS	DS	DS	DS	EPSDE	DS	PSO2011	DS
$F22$	DS	DS	DS	DS	DS	DS	DS	DS
$F23$	ABC	DS	JADE	DS	EPSDE	DS	DS	DS
$F24$	DS	JDE	JADE	DS	EPSDE	DS	DS	DS
$F25$	ABC	DS	DS, JADE	SADE	DS, EPSDE	DS	DS, PSO2011	DS
$F26$	DS	DS	DS	DS	DS	DS	DS	DS
$F27$	DS	DS	DS	DS	DS	DS	DS	DS
$F28$	DS	JDE	DS	SADE	DS	DS	DS	DS
$F29$	DS	JDE	JADE	SADE	EPSDE	DS	PSO2011	DS
$F30$	DS	JDE	JADE	SADE	EPSDE	DS	PSO2011	DS
$F31$	DS	JDE	JADE	DS	DS	DS	DS	CMA-ES
$F32$	DS	DS	JADE	SADE	EPSDE	DS	DS	DS
$F33$	DS, ABC	DS, JDE	DS, JADE	DS, SADE	DS, EPSDE	DS	DS, PSO2011	CMA-ES
$F34$	DS	JDE	DS	SADE	DS	DS	DS	DS
$F35$	ABC	JDE	JADE	SADE	EPSDE	DS	DS	CMA-ES
$F36$	DS, ABC	DS, JDE	DS, JADE	DS, SADE	DS, EPSDE	DS	PSO2011	DS
$F37$	DS	JDE	DS	SADE	DS	DS	DS	DS
$F38$	DS	JDE	JADE	DS	EPSDE	DS	PSO2011	DS, CMA-ES
$F39$	DS	JDE	JADE	DS	EPSDE	DS	DS	CMA-ES
$F40$	DS	DS	DS	SADE	DS	DS	DS	DS
W/T/L-	**28/7/5**	**18/8/14**	**15/11/14**	**18/10/12**	**18/10/12**	**40/0/0**	**24/9/7**	**26/1/13**

of the algorithms JDE, JADE, SADE, EPSDE and CMA-ES have been taken from Zhang (2011). The software code of GSA algorithm has been taken from Rashedi (2011) and software code for PSO2011 from Omran and Clerc, (2011).

5.2. Experiments-2

Along with the developments in Aero-Space technologies, many satellite systems were developed to be used in different fields such as navigation, communication, meteorology, hydrography, environment and climate safety. The orbits of the communication satellites, earth-observation satellites and satellite–based global positioning systems (i.e., GPS, GLONASS, GALILEO) that are used frequently in practical and scientific applications are of geocentric-orbit type that can be defined by means of a terrestrial reference frame (i.e., geosynchronous-orbit, geostationary-orbit). Knowing the position of geo-center with high accuracy is necessary to define the origins of terrestrial reference frames

Table 10
For Test-2, evaluation of the success of DS (proposed) and the comparison algorithms according to the Wilcoxon Rank Sum Test analysis for significance level of $\alpha=0.05$ (W=the number of functions that DS wins, T=the number of functions that DS ties, L=the number of functions that DS loses when compared with its competitors).

FNC	DS-ABC	DS-JDE	DS-JADE	DS-SADE	DS-EPSDE	DS-GSA	DS-PSO2011	DS- CMA-ES
$F1$	DS	JDE	JADE	SADE	EPSDE	DS	PSO2011	CMA-ES
$F2$	DS	DS	DS	DS	DS	DS	DS	DS
$F3$	DS	DS	DS	DS	DS	DS	DS	DS
$F4$	DS	DS	DS	DS	EPSDE	DS	DS	DS
$F5$	DS	DS	DS	DS	DS	DS	DS	DS
$F6$	DS	DS	DS, JADE	SADE	DS	DS	DS	CMA-ES
$F7$	DS	DS	DS	DS	DS	DS	DS	DS
$F8$	DS	DS	JADE	DS	DS	DS	DS	DS
$F9$	ABC	JDE	DS, JADE	SADE	DS, EPSDE	DS	DS	DS
$F10$	DS	DS	DS	DS	DS	DS	DS	DS
$F11$	DS	JDE	JADE	DS	DS	DS	PSO2011	CMA-ES
$F12$	DS	DS	DS	DS	DS	DS	DS	CMA-ES
W/T/L	**11/1/0**	**9/0/3**	**7/2/3**	**9/0/3**	**9/1/2**	**12/0/0**	**10/0/2**	**8/0/4**

(i.e., ITRS6) (Chen et al., 1999; Blewitt, 2003). As the position of geo-center changes on the basis of geodynamic reasons, providing the up to date definition of the origins of terrestrial reference frames related to the practical applications depends on monitoring the changes in the position of the geo-center on the basis of time. For this reason, algorithms that produce stable results near the geo-center are required. Due to the fact that they are used in many applications, it is common practice to use algorithms that make mutual coordinate conversions between geodetic and geocentric coordinates. Additionally, the need to develop new algorithms which produce relatively more stable results, compared to the classic algorithms which have the tendency to produce unstable results when close to the geo-center that is required to define the origins of the coordinate systems that are used in practical applications of classic algorithms, as well as when near-polar or close to the polar-axis or equator is still very much in consideration.

In this paper the use of computational-intelligence algorithms in converting geocentric coordinates into geodetic ones has been analyzed. Herein, the problem of converting geocentric coordinates into geodetic coordinates was defined by means of the optimization problem provided in Eq. (6). To be able to solve the optimization problem stated in Eq. (6), a new computational-intelligence algorithm (i.e., DS) has been introduced in the paper and the success of this new computational-intelligence algorithm to solve the numeric optimization problem has been examined in detail in the previous sections. The detailed experiments performed have shown that computational-intelligence algorithms in general, are relatively more successful in solving the optimization problem provided under Eq. (6) in comparison to classic algorithms. The most important reason for this success is the capability of computational-intelligence algorithms to avoid local solutions unlike classic algorithms. Therefore, in comparison to the classic algorithms, computational-intelligence algorithms are generally more successful in solving the problem indicated in Eq. (6). Furthermore, unlike classic algorithms, computational-intelligence algorithms, to be able to solve the problem indicated in Eq. (6), are capable of forming different initial conditions by changing the values of the parameters that they possess structurally (i.e., size of population, maximum numbers of iterations and other algorithmic-control parameters). Thanks to this quality computational-intelligence algorithms are in possession of a sufficiently flexible structure rendering them capable of solving the subject matter problem.

The probabilistic nature of computational-intelligence algorithms provides the means to achieve a near optimum solution that is quite close to one of optimum results. On the other hand, the solid structuring of classic algorithms leads them to the same result each and every time they are used. This increases the probability of attaining a local solution that would lead to an unstable result when classic algorithm is used near the geo-center, near polar, close to the polar axis and equator.

Table 11
Mean-Runtimes of the algorithms for solving of the problem given in Eq.(6).

Algorithm	Runtimes (in second)
Heikkinen-1982	0.000004
Bowring-1976	0.000012
Fukushima-2006	0.000013
Lin-1995	0.000016
Borkowski-1987	0.000018
Zhang-2005	0.000018
Jones-2002	0.00002
Shu-2010	0.000022
Borkowski, 1987	0.00009
DS (proposed)	0.422625
JADE	0.593182
JDE	0.601022
ABC	0.993859
SADE	1.49178
EPSDE	2.079373
GSA	Failed
PSO2011	Failed
CMA-ES	Failed

Table 12
The number of times an algorithm reaches the best solution among the ones derived for a test point by all the algorithms concerned.

Algorithm	The number of times to reach the best value for φ	The number of times to reach the best value for h
DS	92,813	100,000
ABC	90,395	91,618
SADE	84,767	84,959
EPSDE	84,657	84,792
JDE	84,610	84,811
JADE	83,972	84,232
Bowring-1976	39,915	54,834
Jones-2002	35,918	39,399
Borkowski-1987	27,680	27,249
Shu-2010	10,770	0
Borkowski-1989	7,745	33,958
Lin-1995	2,753	30,119
Fukushima-2006	972	23,630
Heikkinen-1982	5	286
Zhang, 2005	5	285

In general all classic and computational-intelligence algorithms produce results close to each other in terms of their success when they are near the surface. However this does not guarantee a stable result for classic algorithms when near the geo-center, near-polar, close to the polar axis and equator. Moreover, some researchers reported that some classic methods produced unstable results on the basis of increasing h-coordinates. Burtch (2006) and Toms (1995) stated that Bowring-1976 method and some versions of the same method behaved unstable in some polar regions. Zhang et al. (2005) emphasizes that Zhang, 2005 method although very successful in general, has a tendency to be very unstable close to the geo-center. Shu and Li (2010) showed that Shu, 2010 and Bowring, 1976 algorithms are very unstable when close to the geo-center. Borkowski (1989) stated that the success of all iterative methods is greatly dependent on and sensitive to initial value. This limits the success of the iterative basis classic methods to a significant degree.

Moreover, Borkowski (1989) indicates that Borkowski-1989 algorithm produced more accurate results at distances 45 km to 70 km far away from the geo-center. Featherstone and Claessens (2008) on the other hand stated that Borkowski, 1989 algorithm is unstable close to the Equator. Fukushima (1999) has shown that Bowring-1976 algorithm in close proximity to the geo-center and Borkowski-1989 algorithm near the polar axis have proven to be unstable. Furthermore, Fukushima (1999) demonstrated that Borkowski, 1989 algorithm when close to the geo-center and for $h>10^5$ km could produce unstable results. Zhu (1994) stated that

Table 13
MSE_φ ve MSE_h values that are used to analyze the transformation accuracy of the algorithms from the (X,Y,Z) coordinates into the original (φ,h) coordinates.

Algorithm	MSE_φ	MSE_h
DS	6.54E−06	6.195354
ABC	0.001778	41637.81
SADE	0.038871	1762933
EPSDE	0.040852	1852557
JDE	0.039817	1864400
JADE	0.04642	2067942
Bowring-1976	0.380759	1.96E+09
Jones-2002	0.602468	61061352
Borkowski-1987	0.515853	7.13E+12
Shu-2010	0.453114	1.54E+17
Borkowski-1989	0.186673	43460841
Lin-1995	0.067822	6.53E+13
Fukushima-2006	0.347945	46903874
Heikkinen-1982	0.467393	4.31E+08
Zhang, 2005	0.545727	8.67E+11

Table 14
MSE_φ and MSE_h values to analyze the accuracy of the respective algorithms used in this study in regards to the results they obtained at $h<(-R+50$ km), -1000 km $\le h \le 1000$ km and $h>1000$ km, (R=6378137.00 for GRS80 datum).

Algorithm	$h<(-R+50$ km)		-1000 km $\le h \le 1000$ km		$h>1000$ km	
	MSE_φ	MSE_h	MSE_φ	MSE_h	MSE_φ	MSE_h
DS	2.61E-05	24.78142	6.00E−34	4.66E−21	5.02E−34	3.85E−19
ABC	7.11E−03	166551.24	1.08E−33	1.48E−19	6.52E−34	4.11E−17
SADE	1.55E−01	7.05E+06	1.07E−33	1.43E−19	6.48E−34	4.30E−17
JDE	1.59E−01	7.46×10^6	1.09E−33	1.45E−19	6.46E−34	4.37E−17
EPSDE	1.63E−01	7.41E+06	9.61E−34	1.44E−19	6.49E−34	4.12E−17
JADE	1.86E−01	8.27E+06	1.06E−33	1.41E−19	6.48E−34	4.20E−17
Lin-1995	2.78E−01	1.67E+14	5.55E−27	7.73729E+11	4.69E−20	8.81E−16
Borkowski-1989	7.00E−01	1.31E+08	7.68E−28	1.40E−16	6.78E−23	2.21E−12
Fukushima-2006	1.39E+00	1.88E+08	7.21E−26	8.54E−19	2.39E−23	1.50E−15
Bowring-1976	1.52E+00	7.84E+09	4.63E−33	6.39E−19	3.30E−33	8.30E−16
Shu-2010	1.81E+00	6.16E+17	9.15E−19	12.80841	2.27E−20	882688.4729
Heikkinen-1982	1.87E+00	1.72E+09	1.33E-22	4.78E−09	9.53E−25	4.21E−09
Borkowski-1987	2.05E+00	2.84E+13	1.72E−19	5.91E−19	9.68E−33	1.15E−15
Zhang-2005	2.18E+00	3.33E+12	1.33E−22	4.78E−09	9.53E−25	4.21E−09
Jones-2002	2.41E+00	2.44E+08	7.82E−33	9.30E−19	6.77E−33	1.21E−15

Table 15
MSE_X, MSE_Y ve MSE_Z values to analyze the accuracy of the respective algorithms used in this study in regards to the results they obtained at $h<(-R+50$ km), -1000 km $\le h \le 1000$ km and $h>1000$ km, (R=6378137.00 for GRS80 datum).

Algorithm	$h<(-R+50$ km)			-1000 km $\le h \le 1000$ km			$h>1000$ km		
	MSE_X	MSE_Y	MSE_Z	MSE_X	MSE_Y	MSE_Z	MSE_X	MSE_Y	MSE_Z
DS	9.89E−24	1.68E−23	8.11E−23	6.00E−24	3.75E−25	9.61E−23	2.76E−22	1.10E−21	0
ABC	3.55E−23	3.78E−23	5.67E−22	5.41E−23	6.75E−24	9.79E−23	2.76E−22	1.10E−21	0
SADE	1.16E+02	6.84E+01	2.69E+01	3.44E−22	1.76E−22	3.60E−22	4.48E−21	1.10E−21	2.21E−20
JDE	3.48E+01	4.14E+00	9.62E+01	2.39E−22	1.87E−22	3.24E−22	2.01E−19	3.12E−19	9.93E−19
EPSDE	1.35E+02	5.07E+01	2.96E+01	2.01E−22	1.68E−22	1.20E−22	7.09E−20	1.88E−20	0
JADE	7.48E+03	7.53E+03	8.79E+03	2.12E−22	1.09E−22	1.74E−22	3.45E−22	2.21E−21	7.06E−20
Lin-1995	4.68E+13	4.68E+13	7.33E+13	1.78E+11	1.74E+11	4.22E+11	0.000927	0.000932	0.002982
Borkowski-1989	3.46E+06	3.49E+06	4.40E+07	7.28E−15	7.41E−15	1.80E−14	1.07E−11	7.30E−11	1.37E−05
Fukushima-2006	1.78E+07	1.82E+07	6.20E+07	7.78E−13	7.79E−13	7.72E−13	1.57E−07	1.59E−07	1.63E−07
Bowring-1976	1.36E−20	1.33E−20	7.75E+09	1.62E−19	1.56E−19	6.18E−19	3.52E−16	3.60E−16	6.91E−16
Shu-2010	1.45E+12	1.49E+12	6.16E+17	1.82E+13	1.88E+13	6.61E+00	4.43E+16	4.46E+16	442153.5
Heikkinen-1982	3.17E+08	3.22E+08	4.32E+08	3.51E−10	3.45E−10	9.32E−09	3.41E−10	3.41E−10	8.92E−09
Borkowski-1987	6.60E+12	6.97E+12	1.49E+13	1.36E−06	1.31E−06	2.56E−06	8.71E−16	8.83E−16	7.38E−16
Zhang-2005	1.24E+12	1.24E+12	8.48E+11	3.51E−10	3.45E−10	9.32E−09	3.42E−10	3.42E−10	8.92E−09
Jones-2002	1.77E+08	1.74E+08	3.64E−19	5.24E+12	5.49E+12	5.11E−19	1.28E+16	1.29E+16	7.24E−16

Hekkinen-1982 and Borkowski-1989 algorithms although unstable when close to the geo-center, nevertheless, provide the necessary accuracy at a distance of at least 43 km from the geo-center and near the surface of the earth. Therefore, it can be said that Hekkinen-1982 algorithm produces unstable results near the geo-center.

The experimental results mentioned in this section of the paper have shown that the computational-intelligence algorithms provide more stable results with higher accuracy near the geo-center, near polar and polar axis and in close proximity to equator, in comparison to the classic methods.

In this section, a detailed test was performed for solution of the problem defined in Eq. (6) concerning transformation of the geocentric cartesian coordinates into geodetic coordinates. In the tests performed in this section, success of nine classical methods (i.e., Borkowski-1989; Bowring-1976; Fukushima-2006; Heikkinen-1982; Jones-2002; Zhang-2005; Borkowski-1987; Shu-2010; Lin-1995) and nine Computational-Intelligence algorithms (i.e., DS (proposed), ABC, SADE, JDE, JADE, EPSDE, GSA, PSO2011 and CMA-ES) in solving the problem defined in Eq.(6) for transforming the geocentric cartesian coordinates into geodetic coordinates has been compared in terms of accuracy and calculation speed. Surprisingly, the algorithms of GSA, PSO2011 and CMA-ES could not provide an effective solution in the test performed in this section.

For the test performed in this section, the artificially designed data set contains 100,000 test points. φ, λ and h coordinates of

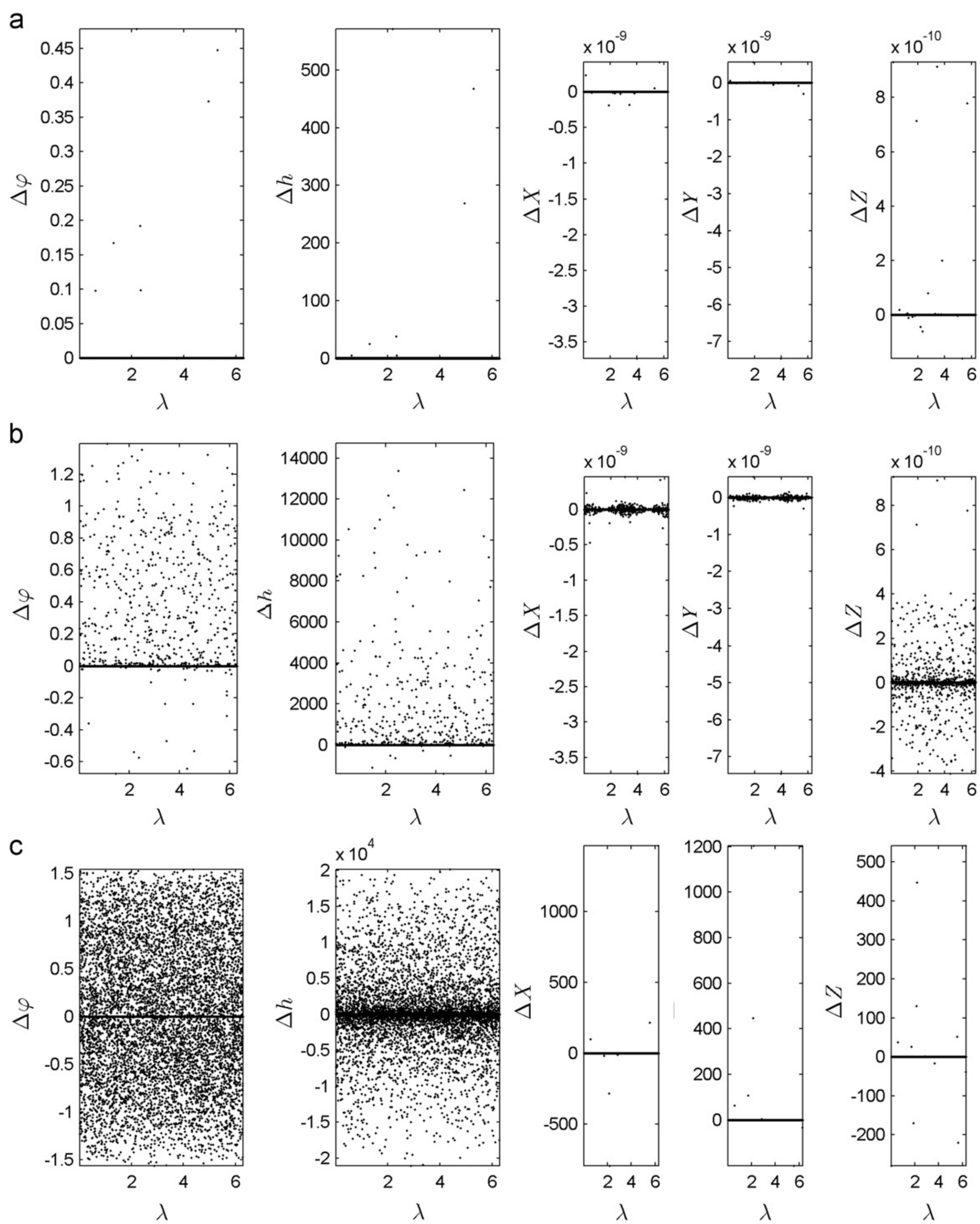

Fig. 2. $\Delta\varphi$, Δh, ΔX, ΔY, ΔZ values for $0 \le \lambda \le 2\pi$: (a) DS, (b) ABC, (c) SADE.

any test point in the designed data set have been produced by using a uniform-random number generator working in the numerical ranges of $0 \le \varphi \le \pi/2$, $0 \le \lambda \le 2\cdot\pi$ and $-6378{,}394\,m \le h \le 500 \times 10^6$ m, respectively. The calculations were made in GRS80 datum. h values substantially cover the numerical range sufficient for geophysics and astro-geodesy applications.

The φ and λ geodetic coordinates that are used in this paper are in radians. On the other hand geodetic coordinate has been defined in meters.

For the respective Computational-Intelligence algorithms related to the tests performed in this section, the required size of population has been selected as 20 and the maximum number of function evaluation value as 1,000,000. In the test performed in this section, if a Computational-Intelligence algorithm cannot reach to a better global minimum value as a result of 50,000 function evaluation operations it has performed, it is stopped.

The test data used in the tests performed in this section consists of (φ,λ,h) geodetic-coordinate values that have been produced for 100,000 test points by using a uniform random number generator to satisfy the condition of, $0 \le \lambda \le 2\pi$, $0 \le \varphi \le \pi/2$ and $0 \le h \le 500 \times 10^6$. The subject matter geodetic coordinates have been converted into (X, Y, Z) geocentric coordinates by using Eq. (1). Following that, Eq. (6) has been solved for each one of (X, Y, Z) geocentric coordinates to calculate $(\varphi^*, \lambda, h^*)$ geodetic coordinates. The $(\varphi^*, \lambda, h^*)$ values obtained were again converted into (X^*, Y^*, Z^*) geocentric coordinates by using Eq. (1).

Mean-Squared-Error (MSE) is a quality measure that is used widely in numerical analysis (Freymueller et al., 1999). For this

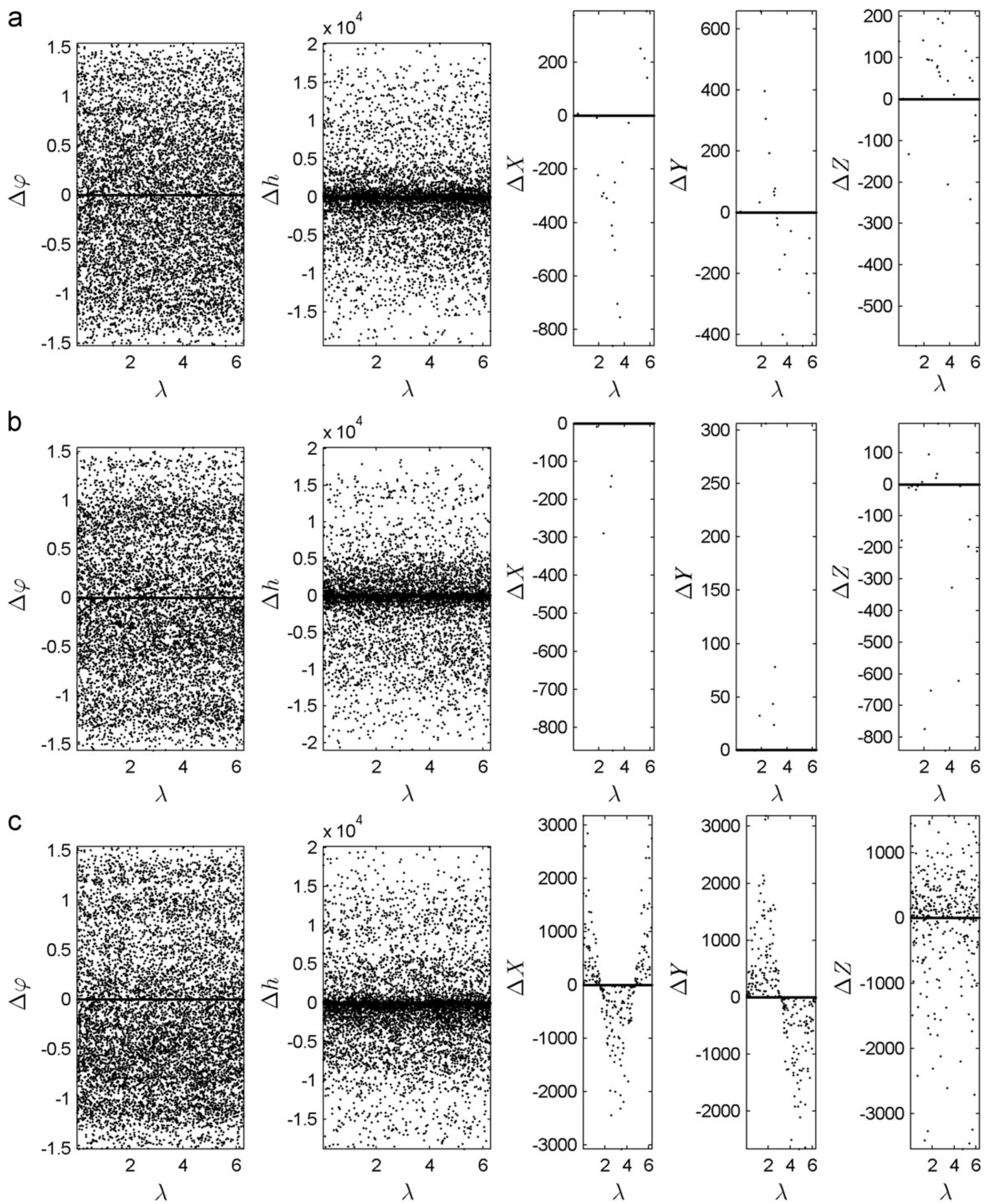

Fig. 3. $\Delta\varphi$, Δh, ΔX, ΔY, ΔZ values for $0 \le \lambda \le 2\pi$: (a) EPSDE, (b) JDE, (c) JADE.

reason, in analyzing the success of the respective algorithms in this paper, MSE values belonging to various parameters were used. To be able to evaluate the success of any algorithm to calculate geodetic coordinates numerically, the values for $MSE\varphi$ (Eq.(9)) and MSE_h (Eq. (10)) were calculated:

$$\Delta\varphi = \varphi_i - \varphi_i^* \text{ and } MSE_\varphi = \frac{1}{n}\sum_{i=1}^{n}(\Delta\varphi)^2 \tag{9}$$

$$\Delta h = h_i - h_i^* \text{ and } MSE_h = \frac{1}{n}\sum_{i}^{n}(\Delta h)^2 \tag{10}$$

In case the geodetic coordinates are re-calculated by using Eq. (1), from the geocentric values obtained by using any algorithm, the MSE_X (Eq. (11)), MSE_Y (Eq. (12)), and MSE_Z (Eq. (13)) values are used to analyze the errors by means of numerical methods;

$$\Delta X_i = X_i - X_i^* \text{ and } MSE_X = \frac{1}{n}\sum_{i}^{n}(\Delta X_i)^2 \tag{11}$$

$$\Delta Y_i = Y_i - Y_i^* \text{ and } MSE_Y = \frac{1}{n}\sum_{i}^{n}(\Delta Y_i)^2 \tag{12}$$

$$\Delta Z_i = Z_i - Z_i^* \text{ and } MSE_Z = \frac{1}{n}\sum_{i}^{n}(\Delta Z_i)^2 \tag{13}$$

here the X, Y, Z geocentric coordinates are in meters.

Solution speed of all algorithms for each point related to the problem given in Eq.(6) is shown in Table 11.

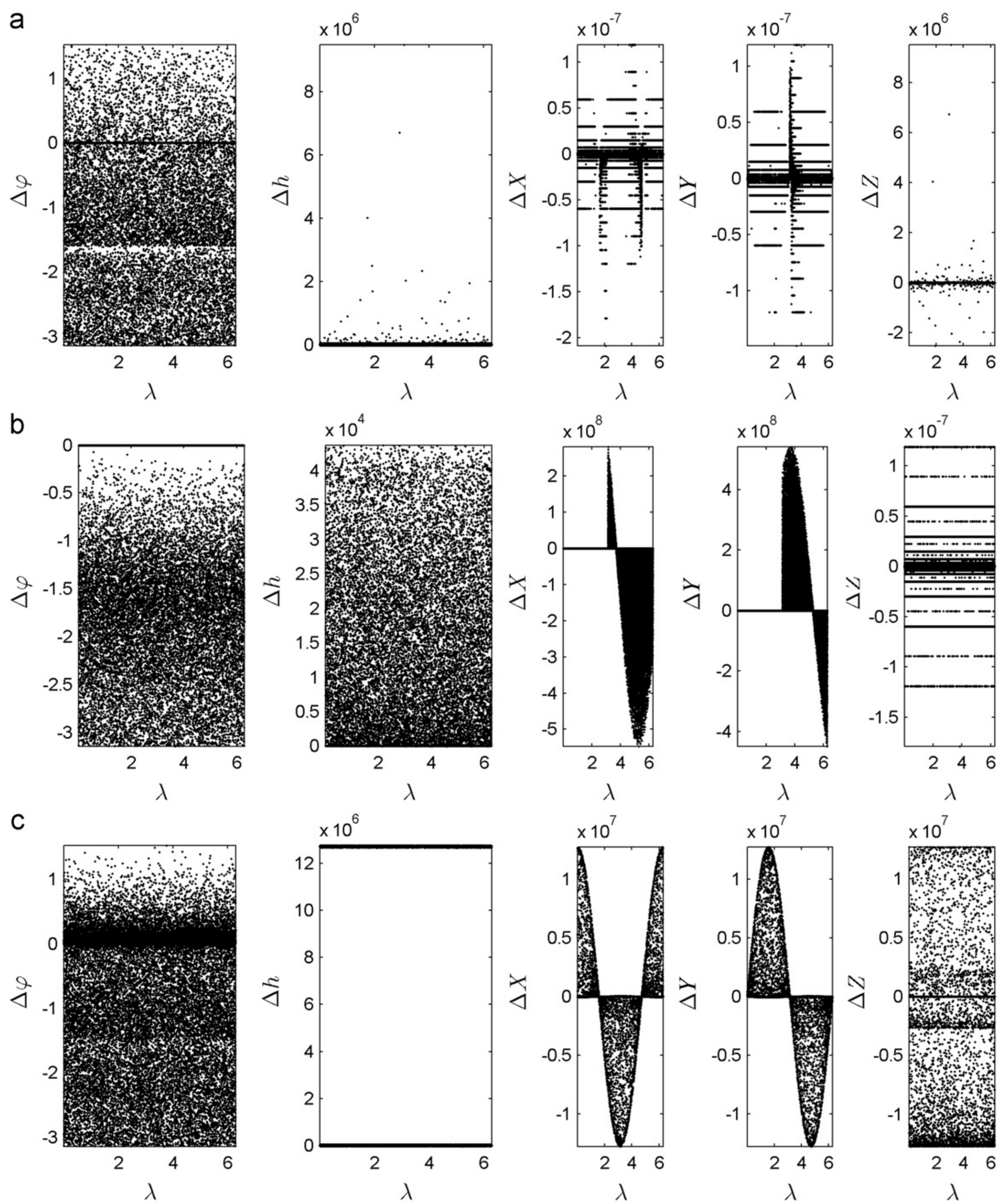

Fig. 4. $\Delta\varphi$, Δh, ΔX, ΔY, ΔZ values for $0 \le \lambda \le 2\pi$: (a) Bowring, 1976, (b) Jones, 2002, (c) Borkowski, 1987.

In Table 12, the number of times an algorithm reaches the best solution among the ones derived for a test point by all the algorithms concerned, is provided.

When Table 12 is examined it can be seen that DS algorithm, in comparison to the other algorithms, is successful at many more test points in calculating the geodetic coordinates from geocentric ones. For accuracy analysis of the results obtained at all the test points by algorithms used in this study, MSE_{φ} ve MSE_h values provided in Table 13 have been utilized.

When Table 13 is inspected, it can be said that computational intelligence algorithms are generally more successful than classic methods. Furthermore, it is seen in Table 13 that DS algorithm provides more successful solutions in comparison to all the other algorithms mentioned in this paper.

MSE_{φ} and MSE_h values in Table 14 are used to analyze the accuracy of the respective algorithms used in this study in regards to the results they obtained at $h<(-R+50\text{ km})$, $-1000\text{ km}\leq h\leq 1000\text{ km}$ and $h>1000\text{ km}$, ($R=6378{,}137.00$ for GRS80 datum).

When Table 14 is examined, it can be said that the success of all the algorithms other than Shu-2010 and Lin-1995 algorithms, is similar to each other. On the other hand, in calculations made near the geo-center, DS and other computational-intelligence algorithms clearly provide more successful results compared to classic algorithms.

MSE_X, MSE_Y and MSE_Z values in Table 15 are used to analyze the accuracy of respective algorithms used in this study in regards to the results they obtained at $h<(-R+50\text{ km})$, $-1000\text{ km}\leq h\leq 1000$ km, and $h>1000$ km.

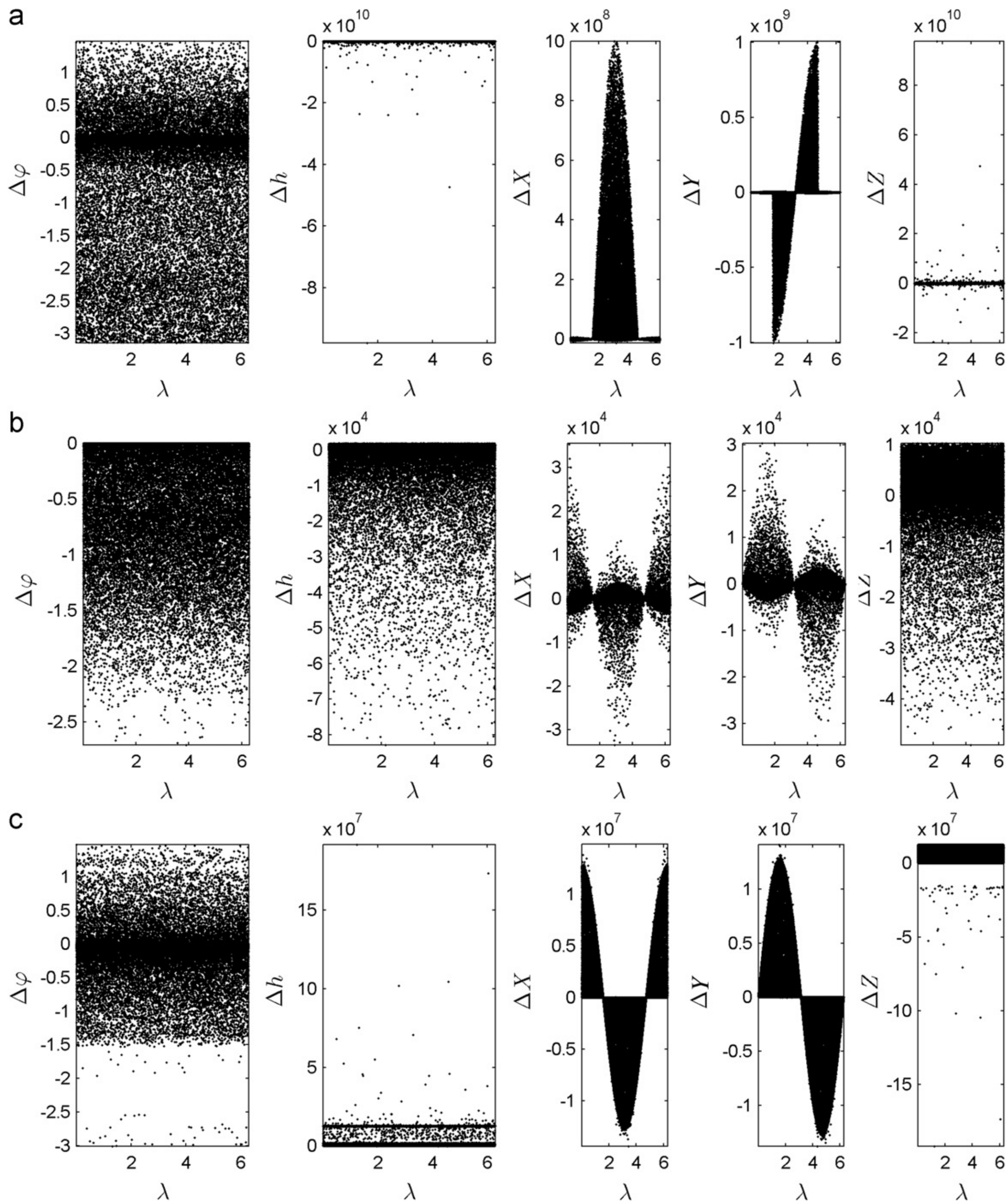

Fig. 5. $\Delta\varphi$, Δh, ΔX, ΔY, ΔZ values for $0\leq\lambda\leq 2\pi$: (a) Shu, 2010, (b) Borkowski, 1987, (c) Lin, 1995.

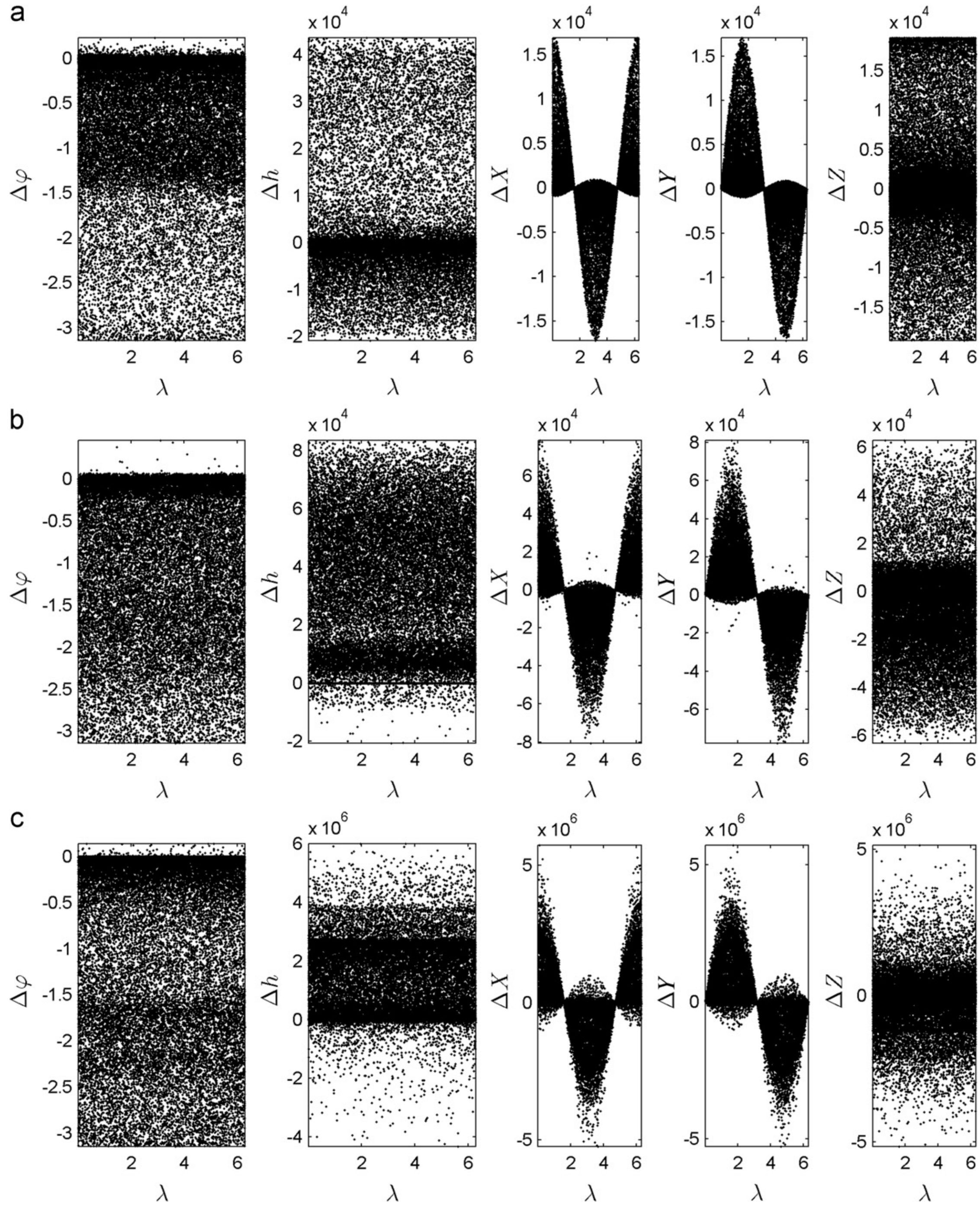

Fig. 6. $\Delta\varphi$, Δh, ΔX, ΔY, ΔZ values for $0 \leq \lambda \leq 2\pi$: (a) Fukushima, 2006, (b) Heikkinen-1982, (c) Zhang-2005.

Upon examining Table 15, we can say that near the geo-center (i.e. $h < (-R+50\text{ km})$) DS and other computational-intelligence algorithms provide more stable and successful results in comparison to the classic algorithms. Lin, 1995, Shu, 2010 and Jones, 2002 algorithms can produce unstable solutions for $h > -1000$ km.

$\Delta\varphi$, Δh, ΔX, ΔY, ΔZ values are provided in graphical format in Figs. 2–4, as calculated on the basis of the solutions obtained. In Fig. 2, by using the DS, ABC, SADE algorithms, in Fig. 3, EPSDE, JDE, JADE algorithms and in Fig. 4, Bowring-1976, Jones-2002, Borkowski-1987 algorithms. Similarly, $\Delta\varphi$, Δh, ΔX, ΔY, ΔZ values are provided in graphical format in Figs. 5 and 6, as calculated on the basis of the solutions obtained. In Fig. 5, by using Shu-2010, Borkowski-1989, Lin-1995 algorithms and in Fig. 6, Fukushima-2006, Heikkinen-1982, Zhang-2005 algorithms.

When Fig. 2–6 are examined, it can be observed that generally DS algorithm produces more stable results in comparison to all other algorithms. In comparison to the classic algorithms, DS and other computational intelligence algorithms produce more stable results in calculating the geodetic coordinates from geocentric coordinates near the geo-center, near-polar, close to the polar axis and equator.

6. Results

This paper introduces a new population-based metaheuristic optimization algorithm (i.e. DS) that may be used for solution of numerical optimization problems. Success of DS algorithm has been compared with widely used 8 different optimization algorithms with the use of statistical methods. The obtained results show that DS algorithm may be used successfully for solution of the numerical optimization problems.

In this paper, the success of DS algorithm in solving the problem of transforming the geocentric cartesian coordinates into geodetic coordinates is compared with the success of nine classical methods (i.e., Borkowski-1989; Bowring-1976; Fukushima-2006; Heikkinen-1982; Jones-2002; Zhang-2005; Borkowski-1987; Shu-2010; Lin-1995) and eight *Computational Intelligence* methods (i.e., ABC, JDE, JADE, SADE, EPSDE, GSA, PSO2011, CMA-ES) in solving the same problem. The obtained results show that DS algorithm could solve the mentioned problem at a very high level of accuracy.

Acknowledgement

The author would like to thank to the referees who have contributed to enhancement of the technical contents of this paper. The studies in this paper have been supported within the scope of the scientific research project of 110Y309 supported by TUBITAK.

Appendix A. Supplementary data

Supplementary data associated with this article can be found in the online version at doi:10.1016/j.cageo.2011.12.011. These data include Google maps of the most important areas described in this article.

References

Blewitt, G., 2003. Self-consistency in reference frames, geocenter definition, and surface loading of the solid Earth. Journal Of Geophysical Research-Solid Earth 108. B2-2103.
Borkowski, K.M., 1987. Transformation of geocentric to geodetic coordinates without approximations. Astrophysics and Space Science 139, 1–4.
Borkowski, K.M., 1989. Accurate algorithms to transform geocentric to geodetic coordinates. Bulletin Geodesique 63, 50–56.
Bowring, B., 1976. Transformation from spatial to geographical coordinates. Survey Review 23, 323–327.
Bowring, B., 1985. The accuracy of geodetic latitude and height equations. Survey Review 28, 202–206.
Bratton, D., Kennedy, J., 2007. Defining a standard for particle swarm optimization. IEEE Swarm Intelligence Symposium, Honolulu, 1-4244-0708-7.
Brest, J., Boškovic, B., Greiner, S., Zumer, V., Maucec, M.S., 2007. Performance comparison of self-adaptive and adaptive differential evolution algorithms. Soft Computing 11, 617–629.
Brest, J., Zamuda, A., Boskovic, B., Maucec, M.S., Zumer, V., 2009. Dynamic optimization using self-adaptive differential evolution. IEEE Congress on Evolutionary Computation, CEC '09, 415–422.
Burtch, R., 2006. A comparison of methods used in rectangular to geodetic coordinate transformations. ACSM Annual Conference and Technology Exhibition, Orlando, FL, April 21–26.
Caponio, A., Cascella, G.L., Neri, F., Salvatore, N., Sumner, M., 2007. A fast adaptive memetic algorithm for on-line and off-line control design of pmsm drives. IEEE Transactions on Systems, Man, and Cybernetics –Part B 37, 28–41.
Caponio, A., Neri, F., Tirronen, V., 2009. Super-fit control adaptation in memetic differential evolution frameworks. Soft Computing – A Fusion of Foundations, Methodologies and Applications 13, 811–831.
Chen, J.L., Wilson, C.R., Eanes, R.J., et al., 1999. Geophysical interpretation of observed geocenter variations. Journal of Geophysical Research-Solid Earth 104, 2683–2690.
Civicioglu, P., 2009. Removal of random-valued impulsive noise from corrupted images. IEEE Transactions on Consumer Electronics 55, 2097–2104.
Civicioglu, P., 2011. 〈http://www.pinarcivicioglu.com/ds.html〉, [accessed 02 October, 2011].
Civicioglu, P., Besdok, E., 2011. A conceptual comparison of the cuckoo search, particle swarm optimization, differential evolution and artificial bee colony algorithms. Artificial Intelligence Review. doi:10.1007/s10462011-92760. (online edition).
Clerc, M., Kennedy, J., 2002. The particle swarm – explosion, stability, and convergence in a multidimensional complex space. IEEE Transactions on Evolutionary Computation 6, 58–73.
Das, S., Abraham, A., Chakraborty, U.K., Konar, A., 2009. Differential evolution with a neighborhood-based mutation operator. IEEE Transactions on Evolutionary Computation. 13, 526–553.
Das, S., Konar, A., Chakraborty, U.K., 2005. Two improved differential evolution schemes for faster global search. In: Proceedings of the 2005 Conference on Genetic and Evolutionary Computation. ACM, New York, pp. 991–998.
Derrac, J., Garcia, S., Monila, 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 and Evolutionary Computation 1, 3–18.
Dorigo, M., Maniezzo, V., Colorni, A., 1996. Ant system: optimization by a colony of cooperating agents. IEEE Transactions on Systems Man and Cybernetics Part B – Cybernetics 26, 29–41.
Eberhart, R.C., Shi, Y.H., 2001. Particle swarm optimization: developments, applications and resources. IEEE Congress on Evolutionary Computation' CEC 2001 1–2, 81–86.
Epitropakis, M.G., Tasoulis, D.K., Pavlidis, N.G., Plagianakos, V.P., Vrahatis, M.N., 2011. Enhancing differential evolution utilizing proximity-based mutation operators. IEEE Transactions on Evolutionary Computation 15, 99–119.
Fan, H.Y., Lampinen, J., 2003a. A directed mutation operation for the differential evolution algorithm. International Journal of Industrial Engineering: Theory, Applications and Practice 1, 6–15.
Fan, H.Y., Lampinen, J., 2003b. A trigonometric mutation operation to differential evolution. The Journal of Global Optimization 27, 105–129.
Featherstone, W.E., Claessens, S.J., 2008. Closed-Form Transformation between geodetic and ellipsoidal coordinates. Studia Geophysica et Geodaetica 52, 1–18.
Feltens, J., 2008. Vector methods to compute azimuth, elevation, ellipsoidal normal, and the cartesian (x, y, z) to geodetic (φ, λ, h) transformation. Journal of Geodesy 82, 493–504.
Feng, W.Z., Kuan, H.H., Bei, Y., Ying, Z., 2008. A modified differential evolution algorithm with self-adaptive control parameters, 3rd International Conference on Intelligent System and Knowledge Engineering, ISKE20081: pp. 524–527.
Freymueller, J.,.T., Murray, M.H., Segall, P., et al., 1999. Kinematics of the Pacific North America plate boundary zone, northern California. Journal of Geophysical Research – Solid Earth 104, 7419–7441.
Fukushima, T., 1999. Fast transform from geocentric to geodetic coordinates. Journal of Geodesy 73, 603–610.
Fukushima, T., 2006. Transformation from Cartesian to geodetic coordinates accelerated by Halley's method. Journal of Geodesy 12, 689–693.
Hansen, N., Niederberger, S.P.N., Guzzella, L., Koumoutsakos, P., 2009. A method for handling uncertainty in evolutionary optimization with an application to feedback control of combustion. IEEE Transactions on Evolutionary Computation 13, 180–197.
Heikkinen, M., 1982. Geschlossene Formeln zur Berechnung raeumlicher geodaetischer Koordinaten aus rechtwinkligen Koordinaten 5, 207–211.
Jones, G.C., 2002. New solutions for the geodetic coordinates transformation. Journal of Geodesy 76, 437–446.
Karaboğa, D., 2011. 〈http://mf.erciyes.edu.tr/abc/projects.htm〉, [accessed 02 October, 2011].
Karaboğa, D., Baştürk, B., 2007. A powerful and efficient algorithm for numerical function optimization: Artificial Bee Colony (ABC) algorithm. Journal of Global Optimization 39, 459–471.
Karaboğa, D., Baştürk, B., 2008. On the performance of Artificial Bee Colony (ABC) algorithm. Applied Soft Computing 8, 687–697.
Kutoğlu, H.S., 2009. Alternative methods for improving transformation consistency between geocentric and non-geocentric (local) coordinate systems. Survey Review 41, 408–418.
Li, Y.X., Zhang, J.H., Zhang, J.Q., et al., 2010. Direct transformation from geocentric cartesian coordinates to geodetic latitude and ellipsoidal height. Journal of Geodesy 42, 166–175.
Liang, J.J., Qin, A.K., Suganthan, P.N., et al., 2006. Comprehensive learning particle swarm optimizer for global optimization of multimodal functions. IEEE Transactions on Evolutionary Computation. 10, 281–295.
Lin, K.C., Wang, J., 1995. Transformation from geocentric to geodetic coordinates using newtons iteration. Bulletin Geodesique 69, 300–303.
Mallipeddi, R., Suganthan, P.N., Pan, Q.K., et al., 2011. Differential evolution algorithm with ensemble of parameters and mutation strategies. Appied Soft Computing 11, 1679–1696.
Mininno, E., Neri, F., Cupertino, F., Naso, D., 2011. Compact differential evolution. IEEE Transactions on Evolutionary Computation. 15, 32–54.
Neri, F., Lacca, G., Mininno, E., 2011. Disturbed exploitation compact differential evolution for limited memory optimization problems. Information Sciences 181, 2469–2487.
Neri, F., Mininno, E., 2010. Memetic compact differential evolution for Cartesian robot control. IEEE Computational Intelligence Magazine 5, 54–65.
Neri, F., Tirronen, V., 2010. Recent advances in differential evolution: a survey and experimental analysis. Artificial Intelligence Review 33, 61–106.
Noman, N., Iba, H., 2008. Accelerating differential evolution using an adaptive local search. IEEE Transactions on Evolutionary Computation 12, 107–125.
Olorunda, O., Engelbrecht, A., 2007. Differential evolution in high-dimensional search spaces. In: Proceedings of the IEEE Congress on Evolutionary Computation, 1934–1941.
Omran, M.G.H., Clerc, M., 2011. 〈http://www.particleswarm.info/〉, [accessed 02 October, 2011].
Pollard, J., 2005. A new approach to the iterative calculation of geodetic latitude and its application. Survey Review 296, 117–123.

Price, K., Storn, R., 1997. Differential evolution. Dr Dobbs Journal 22, 18–24.
Qin, A.K., Suganthan, P.N., 2005. Self-adaptive differential evolution algorithm for numerical optimization. IEEE Congress on Evolutionary Computation 1–3, 1785–1791. 2005.
Qin, A.K., Huang, V.L., Suganthan, P.N., 2009a. Differential evolution algorithm with strategy adaptation for global numerical optimization. IEEE Transactions on Evolutionary Computation. 13, 398–417.
Qin, A.K., Huang, V.L., Suganthan, P.N., 2009b. Multi-objective optimization based on self-adaptive differential evolution algorithm. IEEE Transactions On Evolutionary Computation 13, 398–417.
Rashedi, E., 2011. 〈http://www.mathworks.com/matlabcentral〉 /fileexchange/, [accessed 02 October, 2011].
Rashedi, E., Nezamabadi-pour, H., Saryazdi, S., 2009. GSA: A Gravitational Search Algorithm. Information Science 13, 2232–2248.
Salvatore, N., Caponio, A., Neri, F., Stasi, S., Cascella, G.L., 2010. Optimization of delayed-state Kalman filter-based algorithm via differential evolution for sensorless control of induction motors. IEEE Transactions on Industrial Electronics 57, 385–394.
Seemkooei, A.A., 2002. Comparison of different algorithms to transform geocentric to geodetic coordinates. Survey Review 36, 627–633.
Sekercioğlu, C.H., 2007. Conservation ecology: area trumps mobility in fragment bird extinctions. Current Biology 17, 283–286.
Shu, C., Li, F., 2010. An iterative algorithm to compute geodetic coordinates. Computers & Geosciences 36, 1145–1149.
Storn, R., 1999. System design by constraint adaptation and differential evolution. IEEE Transactions on Evolutionary Computation 3, 22–34.
Storn, R., Price, K., 1997. Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces. Journal of Global Optimization 11, 341–359.
Suganthan, P.N., Hansen, N., Liang, J.J., et al., 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 No 2005005.
Tirronen, V., Neri, F., Kärkkäinen, T., Majava, K., Rossi, T., 2008. An enhanced memetic differential evolution in filter design for defect detection in paper production. Evolutionary Computation. 16, 529–555.
Toms, R.M., 1995. An efficient algorithm for geocentric to geodetic coordinate conversion. 8th international conference on parallel and distributed computing systems, Orlando, FL (United States), Report Number: UCRL-JC-121813, CONF-9509159-1.
Torge, W., 2001. Geodesy. Walter de Gruyter GmbH & Co, Berlin, Germany.
Trelea, I.C., 2003. The particle swarm optimization algorithm: convergence analysis and parameter selection. Information Processing Letters 85, 317–325.
Vanieek, P., Krakiwsky, E.J., 1986. Geodesy: the Concepts. Elsevier Science, Lincoln, UK.
Vermeille, H., 2004. Computing geodetic coordinates from geocentric coordinates. Journal of Geodesy 78, 94–95.
Vermeille, H., 2001. Direct transformation from geocentric coordinates to geodetic coordinates. Journal of Geodesy 76, 451–454.
Vermeille, H., 2011. An analytical method to transform geocentric into geodetic coordinates. Journal of Geodesy 85, 105–117.
Vito, T., Elio, T., Kevin, M.P., et al., 2011. Swarm cognition: an interdisciplinary approach to the study of self-organising biological collectives. Swarm Intelligence 5, 3–18.
Wang, Y., Cai, Z., Zhang, Q., 2011. Differential evolution with composite trial vector generation strategies and control parameters. IEEE Transactions on Evolutionary Computation 15, 55–66.
Weber, M., Tirronen, V., Neri, F., 2010. Scale factor inheritance mechanism in distributed differential evolution. Soft Computing – A Fusion of Foundations, Methodologies and Applications 14, 1187–1207.
Wenyin, G., Zhihua, C., Charles, X.L., et al., 2011. Enhanced differential evolution with adaptive strategies for numerical optimization. IEEE Transactions on Systems Man And Cybernetics Part B-Cybernetics 41, 39–7413.
Yang, X.S., Deb, S., 2010. Engineering optimisation by cuckoo search. International Journal of Mathematical Modelling and Numerical Optimisation 1, 330–343.
Yen, G.G., Leong, W.F., 2009. Dynamic multiple swarms in multiobjective particle swarm optimization. IEEE Transactions on Systems, Man, and Cybernetics – Part A: Systems and Humans 39, 890–911.
Zhang, C.D., Hsu, H.T., Wu, X.P., Li, S.S., Wang, Q.B., Chai, H.Z., Du, L., 2005. An alternative algebraic algorithm to transform Cartesian to geodetic coordinates. Journal of Geodesy 79, 413–420.
Zhang, J., Sanderson, A.C., 2009. JADE: adaptive differential evolution with optional external archive. IEEE Transactions on Evolutionary Computation 13, 945–958.
Zhang, Q., 2011. 〈http://dces.essex.ac.uk/staff/〉 qzhang/, [accessed 02 October, 2011].
Zhu, J., 1993. Exact conversion of earth-centered, earth-fixed coordinates to geodetic coordinates. Journal of Guidance Control and Dynamics 16, 389–391.
Zhu, J., 1994. Conversion of earth-centered earth-fixed coordinates to geodetic coordinates. IEEE Transactions on Aerospace and Electronic Systems 30, 957–962.