Çukurova Üniversitesi Mühendislik Mimarlık Fakültesi Dergisi, 32(3), ss. 243-249, Eylül 2017
*Çukurova University Journal of the Faculty of Engineering and Architecture, 32(3), pp. 243-249, September 2017*

# A New Adaptive Particle Swarm Optimization Based on Self-Tuning of PID Controller for DC Motor System

**Hussein ALRUIM ALHASAN[1], Mahit GÜNEŞ[*1]**
[1]*Kahramanmaraş Sütçü İmam Üniversitesi, Mühendislik Fakültesi, Elektrik-Elektronik Mühendisliği Bölümü, Kahramanmaraş*

*Geliş tarihi: 10.05.2017 Kabul tarihi: 25.09.2017*

## Abstract

This paper presents a new adaptive particle swarm optimization algorithm for optimal self-tuning of PID controller in dc motor system. Manual tuning of PID controllers does not provide good performance, time consuming, difficult and tedious. The tuning process of PID controller is done by PSO algorithm. Inertia weight is the most important parameter in PSO algorithm, which gives a control of the exploration-exploitation characteristics of PSO algorithm. Since the beginning of Inertia Weight in PSO algorithm, Different strategies of PSO algorithm have been proposed in order to determine the inertia weight. In this paper, we propose a completely new strategy to adapt the inertia weight based on the fitness value of the particles. Comparing with standard PSO algorithm and time varying inertia weight PSO algorithm, the proposed adaptive PSO algorithm gives better performance in terms of quick convergence capability and continues movement toward the optimal solution region.

**Keywords:** PID controller, Adaptive particle swarm optimization, Inertia weight, DC motor

# Yeni Bir Adaptif Parçacık Sürü Optimizasyon Algoritması Kullanarak DC Motor için Öz Ayarlamalı PID Kontrolör Tasarımı

## Öz

Bu çalışmada, DC motor sisteminde yeni bir adaptif parçacık sürü optimizasyon algoritması kullanarak öz ayarlamalı PID kontrolör tasarlanmıştır. PID kontrolörleri parametre ayarlarının manuel yapılması zaman alıcı olması, uzun sürmesi ve hassas olmaması gibi nedenlerden dolayı her zaman iyi bir performans sağlamaz. Bu çalışmada PID kontrolörün parametre ayarları, Parçacık Sürü Optimizasyonu (PSO) algoritması ile yapılmıştır. Eylemsizlik ağırlığı PSO algoritmasında en önemli parametredir. Eylemsizlik ağırlığı, PSO algoritmasının arama özelliğini kontrol eder. PSO algoritmasının başlangıcından bu yana, uygun eylemsizlik ağırlığı belirlemek için farklı PSO algoritması stratejileri önerilmiştir. Bu çalışmada, parçacıkların uygunluk değerlerinin karşılaştırılmasına dayanılarak eylemsizlik ağırlığı ayarlamak için yeni bir strateji önerilmiştir. Standart PSO algoritması (S-PSO) ve zaman değişen eylemsizlik ağırlığı PSO algoritması (TVIW-PSO) ile karşılaştırıldığında, önerilen adaptif PSO algoritması, hızlı yakınsama ve optimal çözüme doğru harekete devam etmesi açısından daha iyi performans verdiği gözlenmiştir.

**Anahtar Kelimeler:** PID Kontrolör, Adaptif Parçacık Sürü Optimizasyonu, Eylemsizlik Ağırlığı, DC motor

[*]Sorumlu yazar (Corresponding author): Mahit GÜNEŞ, *mgunes@ksu.edu.tr,*

## 1. INTRODUCTION

DC motors have been widely used in many industrial applications such as lathe machines, centrifugal pumps, fans and robotic manipulators due to simple and wide control characteristics. There are different controller types used to control the DC motor such as PID Controller, LQR Controller, Fuzzy Logic Controller. PID controllers are widely used in industrial process because of its simplicity of implementation, remarkable effectiveness and broad applicability. The performance of a PID controller completely depends on the tuning process of the proportional, integral and derivative gains. Manual tuning of PID controllers requires experienced personnel in order to provide acceptable performance. A lot of intelligent approaches have been proposed to improve the capabilities of PID controller by tuning its parameters such as PSO based PID [1], bees algorithm based PID [2], Neural Networks based PID [3] and Genetic Algorithm based PID [4]. Particle swarm optimization (PSO) is a population based computational search and optimization method. The PSO algorithm have been applied widely to solve many complex optimization problems. This algorithm is used to improve the performance and enhance the efficiency of different controllers such as PID controller [1], LQR controller [5], Fuzzy logic controller [6] and artificial neural network controller [7].

In this paper, A mathematical modelling of dc motor system and investigations of performance comparison between different strategies to adapt the inertia weight in PSO algorithm [8]. The inertia weight is the most important parameter in particle swarm optimization. It is the key to balance the global search ability or exploration and local search ability or exploitation. Many researchers have recommended that the value of inertia weight should be large in the exploration state and small in the exploitation state [9-11]. The proper adapting of inertia weight increase the efficiency of PSO. Where the efficiency of PSO is determined as the number of iterations to reach the optimal solution.

## 2. MATHEMATICAL MODEL OF DC MOTOR

In this model, the dc motor dynamics are idealized; for instance, the magnetic field is assumed to be constant. Figure 1 shows the schematic diagram of a separately excited dc motor driving an inertial load. The torque $T_m$ generated by a dc motor is proportional to the armature current (i) induced by the applied voltage.

$$T_m(t) = K_m\, i(t) \tag{1}$$

Symbol $K_m$ is the armature constant. The back electromotive force $V_{emf}$ is a voltage proportional to the angular velocity w of the shaft.

$$V_{emf}(t) = K_b\, w(t) \tag{2}$$

Where $K_b$, the emf constant. The mechanical part of the motor equations is derived using Newton's law.

$$J\frac{dw}{dt} = -K_f\, w(t) + K_m\, i(t) \tag{3}$$

Where $K_f$ is a linear approximation for viscous friction. The following equation describes the electrical part of the motor.

$$V_{app}(t) - V_{emf}(t) = L\frac{di}{dt} + R\, i(t) \tag{4}$$

After simplifying the above equations, we will obtain a two differential equations that describe the dynamics of the motor. The first for the induced armature current,

$$\frac{di}{dt} = -\frac{R}{L}i(t) - \frac{K_b}{L}w(t) + \frac{1}{L}V_{app}(t) \tag{5}$$

and the second for the angular velocity,

$$\frac{dw}{dt} = -\frac{K_f}{J}w(t) + \frac{K_m}{J}i(t) \tag{6}$$

The above (5) and (6) equations show the dynamics of the DC motor system. A state-space representation of the DC motor system is obtained from the two linear differential equations. The induced armature current (i) and the angular velocity w are the state variables. The applied voltage, $V_{app}$, is the system input, while the angular velocity w is the system output.

$$\frac{d}{dt}\begin{bmatrix} i \\ w \end{bmatrix} = \begin{bmatrix} \frac{-R}{L} & \frac{-K_b}{L} \\ \frac{K_m}{J} & \frac{-K_f}{J} \end{bmatrix} \begin{bmatrix} i \\ w \end{bmatrix} + \begin{bmatrix} \frac{1}{L} \\ 0 \end{bmatrix} V_{app}(t) \tag{7}$$

$$y(t) = \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} i \\ w \end{bmatrix} + \begin{bmatrix} 0 \end{bmatrix} V_{app}(t)$$

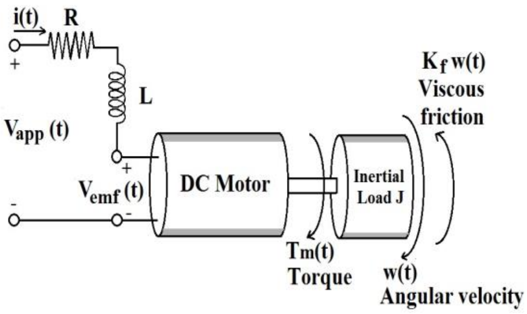


**Figure 1.** The schematic diagram of a separately excited DC motor driving an inertial load

The dc motor parameters and constants used in simulation are detailed below in Table 1.

**Table 1.** Parameters of separately excited DC motor model

| Parameter | Symbol | Value | Unit |
|---|---|---|---|
| The armature resistance | R | 2 | Ω |
| The armature inductance | L | 0.5 | Henry |
| The armature constant | $K_m$ | 0.1 | |
| The emf constant | $K_b$ | 0.1 | |
| Friction coefficient | $K_f$ | 0.2 | N.m.s |
| Moment of inertia | J | 0.02 | $Kg.m^2$ |

## 3. PID CONTROLLER DESIGN

The PID control is a common strategy to control of dc motor for wide range of industrial processes due to its functional simplicity, applicability and ease of use [12]. PID controller is a type of feedback controllers. It has a single output which is the control variable (u) and a single input which is the error (e) between a reference variable (r) and measured process variable (y). The transfer function of the PID controller is expressed as follows:

$$u = K_p\, e(t) + K_i \int_0^t e(t)dt + K_d \frac{de(t)}{dt} \tag{8}$$

Where the tuning parameters $K_P$, $K_i$ and $K_d$ refer to the proportional, integral and derivative gains respectively. The PID controller computes the error value (e) as the difference between a desired input value and measured output. The aim is to minimize the error over time by modifying the control variable (u).

Manual tuning of PID controllers does not provide good performance, time consuming and difficult. In this paper, the tuning process for the PID gains is performed through standard PSO algorithm, time varying inertia weight PSO algorithm and the proposed adaptive PSO algorithm.

## 4. STANDARD PARTICLE SWARM OPTIMIZATION

Particle swarm optimization algorithm is an evolutionary computational strategy that optimizes a problem by iteratively improving a particle solution inspired by the flocking and schooling patterns of birds and fish. The PSO algorithm was first introduced by Erberhart and Kennedy in 1995 [13,14].

During each iteration of the PSO algorithm, each single solution in the search space is evaluated by its fitness function being optimized. Particles have velocities to direct them toward the optimal solution. Figure 2 presents the flow chart of a particle swarm optimization process.

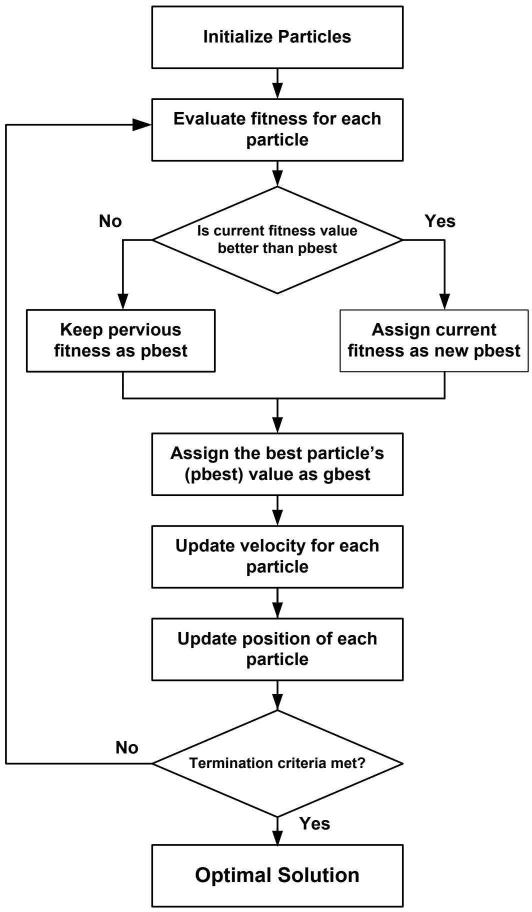


**Figure 2.** Flow chart of Particle Swarm Optimization algorithm

PSO algorithm is initialized with random particles, then it searches for the optimal solution by updating generations. Each particle represent the proportional, integral and derivative gains of PID controller. In every iteration, each particle is updated by two significant values pbest and gbest. pbest is the particle's best known position, while gbest is the swarm's best known position. Each particle updates its positions and velocities according to the two best values with following equations:

$$V_i^{(t+1)} = w * V_i^{(t)} + c_1 * r_1 * (x_{i,best}^{(t)} - x_i^{(t)}) + c_2 * r_2 * (x_{gbest}^{(t)} - x_i^{(t)}) \tag{9}$$

$$x_i^{(t+1)} = x_i^{(t)} + V_i^{(t+1)} \tag{10}$$

Where (w) is the inertia weight, ($c_1$) and ($c_2$) are acceleration coefficients; while $r_1$ and $r_2$ are two independently generated random numbers within the range [0, 1]. In standard PSO algorithm, the inertia weight and acceleration coefficients are constants. Generally, w=1 and $c_1$=$c_2$=2.

The fitness value of particles is evaluated in each iteration. If the fitness value of particle is smaller than gbest fitness value, then the new particle's position becomes gbest. If the same particle's fitness value is smaller than pbest fitness value, then pbest is replaced by the current position.

PSO algorithm is continuing until stopping condition is met. The most common stopping condition is a preset number of iterations.

## 5. PROPOSED ADAPTIVE PARTICLE SWARM OPTIMIZATION

Since the initial development of particle swarm optimization by Kennedy and Eberhart, several methods to improve this algorithm have been proposed by researchers. The experiments show that inertia weight is the key to balance the global search ability (exploration) and local search ability (exploitation). When inertia weight is higher, the global search ability is strong but the local search ability is weak. Likewise, when inertia weight is lower, the local search ability is strong and global search ability is weak. This balancing improves the performance of PSO.

Eberhart and Shi [15] proposed a random inertia weight strategy and experimentally found that this strategy increases the convergence of PSO in early iterations of the algorithm.

$$w = 0.5 + \frac{rand()}{2} \tag{11}$$

The linearly decreasing strategy [16] enhances the efficiency and performance of PSO. It is found experimentally that the decreasing of inertia weight from '0.9' to '0.4' provides the excellent results. In spite of its ability to converge optimum, it gets into the local optimum solving the question of more apices function.

$$w_k = w_{\max} - (w_{\max} - w_{\min}) \times \frac{iter}{iter_{\max}} \quad (12)$$

In this paper, we suppose that the value of the inertia weight (ω) is adaptively adjusted by the proposed equation:

$$\lambda_{1,i}^{K} = \frac{1}{1+e^{\alpha(F_{p,i}^{K} - F_{pbest,i}^{K})}}$$

$$\lambda_{2,i}^{K} = \frac{1}{1+e^{\alpha(F_{pbest,i}^{K} - F_{gbest}^{K})}} \quad (13)$$

$$w_i^K = w_{\max} - (w_{\max} - w_{\min}) \times \left(\lambda_{1,i}^{K} + \lambda_{2,i}^{K}\right)$$

The inertia weight (w) ranges from '$w_{max}$=0.9' to '$w_{min}$=0.3' according to the values of ($\lambda_1$, $\lambda_2$) every iteration (K). The values of ($\lambda_1$, $\lambda_2$) refer to the evolutionary states of the swarm i.e. exploration or exploitation state. Where (α) is a constant to adjust sharpness of the sigmoid function. We suggest that this constant should achieve the following conditions ($\alpha<1$). In this paper, we suggest that ($\alpha=0.1$).

If Particles are too far from the solution (exploration state), the fitness of particles are bigger than the fitness of Pbest particles and the fitness of Pbest particles are bigger than the fitness of gbest particle (Fp,i>Fpbest,i>Fgbest), so the values of (λ1, λ2) are close to zero. In this condition, the inertia weight should be set to larger value. The larger value of inertia weight enhance the global search ability.

If Particles are close to the gbest solution (exploitation state), the fitness of particles are close to the fitness of Pbest particles and the fitness of Pbest particles are also close to the fitness of gbest particle (Fp,i ≈ Fpbest,i ≈ Fgbest), so the values of (λ1, λ2) are close to (0.5). In this condition, the inertia weight should be set to smaller value. The smaller value of inertia weight enhance the local search ability.

The fitness function used to evaluate particles in PSO algorithm is the integrated absolute error (IAE). The integrated absolute error (IAE) formula is as follows:

$$IAE = \int_0^{\infty} |e(t)|\, dt \quad (14)$$

As a result of this method, the balancing between exploration and exploitation abilities can be regulated depending on the inertia weight.

## 6. SIMULATION RESULTS

In this section a comparison and simulation results for three techniques of PSO to design PID controller of dc motor system. PID controller is designed by optimal tuning the parameters $K_P$, $K_i$ and $K_d$. These techniques are standard particle swarm optimization (S-PSO), time-varying inertia weight particle swarm optimization (TVIW-PSO) and the proposed adaptive particle swarm optimization (A-PSO).

PID controller of dc motor system is modelled and simulated in the MATLAB/Simulink environment. The speed response of PID Controller tuned by three particle swarm optimization strategies are shown in Figure 3.

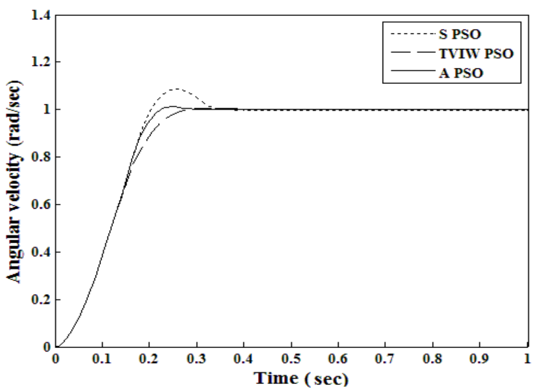


**Figure 3.** The time response of the angular velocity of the dc motor system with PID controller

The standard PSO is the first algorithm. The inertia weight is set to '1' and the acceleration coefficients are set to '2'. By tuning the PID gains using standard particle swarm optimization, the resulting PID gains are:

$$K_p = 2181,\ K_i = -1.74,\ K_d = 33.59$$

In the second method, time-varying inertia weight PSO algorithm is used to tune the PID gains. The inertia weight is adapted according to equation (12). The acceleration coefficients are set to '2'. At the end of this algorithm, the resulting PID gains are:

$$K_p = 39.41,\ K_i = 33.66,\ K_d = 1.83$$

The last algorithm is the proposed adaptive PSO algorithm. The inertia weight is adapted according to equation (13). The acceleration coefficients are set to '2'. At the end of algorithm, the resulting PID gains are:

$$K_p = 485.55,\ K_i = 34,\ K_d = 14.24$$

The best fitness value of the three PSO algorithms across number of iterations is presented in Figure 4. According to the fitness function described in equation (14), it is aimed to minimize the integrated absolute error of the output over time.

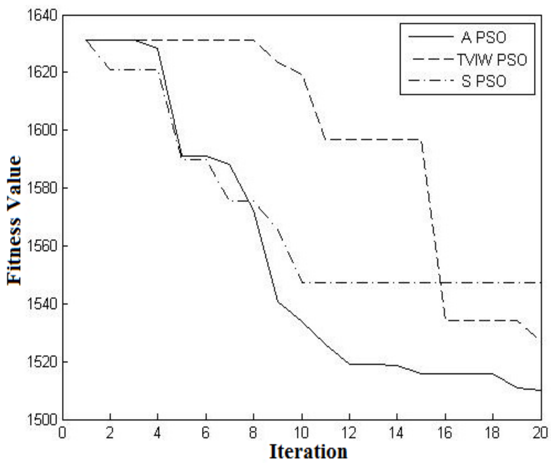


**Figure 4.** The best fitness value of the three PSO algorithms across '20' iterations

The time response specification of dc motor system with PID controllers tuned by standard PSO, time varying inertia weight PSO, and the proposed adaptive PSO techniques are given in Tables 2.

**Table 2.** The time response specification for the angular velocity of the dc motor system

| | Standard PSO | TVIW-PSO | Adaptive PSO |
|---|---|---|---|
| Settling time(s) | 0.32 | 0.25 | 0.21 |
| Rise time(s) | 0.13 | 0.15 | 0.13 |
| Overshoot(%) | 8.53 | 0.61 | 1.27 |
| Steady state error | 0 | 0 | 0 |

## 7. CONCLUSION

In this study, PID controller is designed and applied to the dc motor system. PID controller is tuned by standard particle swarm optimization (S-PSO), time-varying inertia weight particle swarm optimization (TVIW-PSO) and the proposed adaptive particle swarm optimization (A-PSO). The proposed adaptive PSO algorithm has a great improvement in quick convergence capability and continues movement toward the optimal solution region.

Adaptive particle swarm optimization algorithm (APSO) provides automatic control of inertia weight, acceleration coefficients and other algorithmic parameters over time to improve search efficiency and convergence speed. The inertia weight in PSO algorithm is used to balance the global and local search capabilities. In this paper, the proposed adaptive PSO algorithm controls the inertia weight by evaluating the fitness information of particles in order to determine the situation of the swarm at each iteration. The value of inertia weight should be large in the exploration state and small in the exploitation state.

From the performance indices of the angular velocity, the rise time for the three PID controllers is similar to be equal. The settling time for the PID controller using the proposed adaptive PSO is better than the PID controller using standard PSO and the time varying inertia weight PSO. Overshoot for the PID controller using standard

PSO algorithm is the worst. Overshoot for the PID controllers using APSO and TVIW PSO is too low. In addition, steady state error is zero for all of the PID controllers.

Figure 4 shows that, the adaptive PSO algorithm has quick convergence toward the optimal solution region better than time varying inertia weight PSO and standard PSO. The adaptive PSO algorithm reaches the best and lowest fitness value.

As a result, the proposed adaptive PSO algorithm is trustworthy to solve the different optimization problems. It is still simple and almost as easy to use as the standard PSO, whereas it improves performance in terms of convergence speed, solution accuracy and global optimality.

## 8. REFERENCES


1. Berber, Ö., Ateş, M., Alhassan, H.A., Güneş, M., 2016. Parçacık Sürü Optimizasyonu ve PID ile Mobil Robotun Optimum Yörünge Kontrolü, Kahramanmaras Sutcu Imam University Journal of Engineering Sciences, 19(3), 165-169.
2. Çoban, R., Erçin, Ö., 2012. Multi-objective Bees Algorithm to Optimal Tuning of PID Controller, Cukurova University Journal of the Faculty of Engineering and Architecture, 27(2), 13-26.
3. Chang, W.D., Hwang, R.C., Hsieh, J.G., 2003. A Multivariable On-line Adaptive PID Controller Using Auto-tuning Neurons, Engineering Applications of Artificial Intelligence, 16(1), 57-63.
4. Gündoğdu, Ö., 2005. Optimal-tuning of PID Controller Gains Using Genetic Algorithms, Journal of Engineering Sciences, 11(1), 131-135.
5. Hassani, K., Lee, W.S., 2014. May. Optimal Tuning of Linear Quadratic Regulators Using Quantum Particle Swarm Optimization, In Proceedings of the International Conference on Control, Dynamic Systems, and Robotics (CDSR’14) 14-15.
6. Çoban, R., 2011. A Fuzzy Controller Design for Nuclear Research Reactors using the Particle Swarm Optimization Algorithm, Nuclear Engineering and Design, 241(5), 1899-1908.
7. Çoban, R., 2014. Power Level Control of the TRIGA Mark-II Research Reactor Using the Multifeedback Layer Neural Network and the Particle Swarm Optimization, Annals of Nuclear Energy, 69, 260-266.
8. Bansal, J.C., Singh, P.K., Saraswat, M., Verma, A., Jadon, S.S., Abraham, A., 2011. October. Inertia Weight Strategies in Particle Swarm Optimization, In Nature and Biologically Inspired Computing (NaBIC), 633-640.
9. Shi, Y., 2001. Particle Swarm Optimization: Developments, Applications and Resources. In Evolutionary Computation, vol. 1, 81-86.
10. Shi, Y., Eberhart, R., 1998. A Modified Particle Swarm Optimizer, In Evolutionary Computation Proceedings, IEEE World Congress on Computational Intelligence, 69-73.
11. Shi, Y., Eberhart, R.C., 1999. Empirical Study of Particle Swarm Optimization. In Evolutionary Computation, vol. 3, 1945-1950.
12. Meshram, P.M., Kanojiya, R.G., 2012. Tuning of PID Controller Using Ziegler-Nichols Method for Speed Control of DC Motor, In Advances in Engineering, Science and Management (ICAESM), 117-122.
13. Kennedy, J., Eberhart, R., 1995. Particle Swarm Optimization, Proceedings of the 4th IEEE International Conference on Neural Networks, 1942-1948.
14. Eberhart, R., Kennedy, J., 1995. A New Optimizer Using Particle Swarm Theory, In Micro Machine and Human Science, MHS’95, 39-43.
15. Eberhart, R.C., Shi, Y., 2001. Tracking and Optimizing Dynamic Systems With Particle Swarms, In Evolutionary Computation, vol. 1, 94-100.
16. Xin, J., Chen, G., Hai, Y., 2009. A Particle Swarm Optimizer With Multi-stage Linearly-decreasing Inertia Weight, In Computational Sciences and Optimization, 2009. vol. 1, 505-508.