ISSN (Online) 2321-2004
ISSN (Print) 2321-5526

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH IN ELECTRICAL, ELECTRONICS, INSTRUMENTATION AND CONTROL ENGINEERING
Vol. 3, Issue 5, May 2015

ORDER REDUCTION OF LINEAR DYNAMIC DISCRETE SYSTEMS USING IMPROVED GENERALISED LEAST-SQUARES METHOD AND DIFFERENTIAL EVOLUTION ALGORITHM

J.Nancy Namratha[1], Y.Hema Latha[2]

Assistant professor, Department of Electrical Engineering, R.V.R & J.C College of Engineering, Guntur, A.P, India[1]

M.E. Student, Department of Electrical Engineering, ANITS College of Engineering, Visakhapatnam, A.P, India[2]

Abstract: The authors present an algorithm for order reduction of linear dynamic SISO discrete systems using the combined advantages of the improved generalised Least squares method and error minimization by Differential Evolution technique (DE). The denominator of the reduced order model is obtained by improved generalise least squares method and the numerator coefficients are determined by minimizing the integral square error between the transient responses of original and reduced order models using DE technique, pertaining to unit step input. The reduction procedure is simple, efficient and computer oriented. The algorithm is illustrated with the help of two numerical examples to highlight the advantages of the approach and the results are compared with the other existing techniques.

Keywords:Improved Least Squares, Integral square error, Order Reduction, Differential algorithm.

I. INTRODUCTION

Mathematical modelling of most physical systems is carried out using theoretical considerations. This modelling procedure leads to high order state space model in time domain or state space representation and a high order transfer function model in frequency domain representation . It is often desirable to represent such models by equivalent lower order state variable or transfer function models for control and other purposes. Several methods [1]-[5] have been proposed for solving the model approximation problem and they may be grouped into two major categories, the performance-oriented and the non performance-oriented approaches. In using a model approximation method that is not performance-oriented, the original system model is first transformed into a canonical form.For a performance oriented one , approximate models are obtained by minimizing certain approximation error criteria , such as H^2 – norm [10] . Differential evolution (DE) is a simple evolutionary algorithm [11] that mutates vectors by adding weighted, random differentials to them. The choice of differential evolution algorithm for numerical optimization is based on its useful features [12]-[13] .

II. DESCRIPTION OF PROBLEM

Consider an n^{th} order linear time invariant discrete system represented by

$$G_n(z) = \frac{N(z)}{D(z)} = \frac{A_0 + A_1 z + \ldots\ldots + A_{n-1} z^{n-1}}{B_0 + B_1 z + \ldots\ldots + B_{n-1} z^{n-1} + B_n z^n} = \frac{\sum_{i=0}^{n-1} A_i z^i}{\sum_{j=0}^{n} B_j z^j} \quad \ldots\ldots (1)$$

Apply linear transformation $z = 1 + p$ to $G_n(z)$, then

$$G_n(p) = \frac{N(p)}{D(p)} = \frac{a_0 + a_1 p + \ldots\ldots\ldots\ldots + a_{n-1} p^{n-1} + a_n p^n}{b_0 + b_1 p + \ldots\ldots\ldots + b_{n-1} p^{n-1} + b_n p^n}$$

The objective is to find an r^{th} order model that has a transfer function$(r < n)$:

$$R(p) = \frac{N_r(p)}{D_r(p)} = \frac{c_0 + c_1 p + \ldots\ldots + c_{r-1} p^{r-1}}{d_0 + d_1 p + \ldots\ldots + d_{r-1} p^{r-1} + p^r} = \frac{\sum_{i=0}^{r-1} c_i p^i}{\sum_{j=0}^{r} d_j p^j}$$

Apply inverse linear transformation $p = z - 1$ to $R(p)$, then

$$R(z) = \frac{N_r(z)}{D_r(z)} = \frac{C_0 + C_1 z + \ldots\ldots + C_{r-1} z^{r-1} + C_r z^r}{D_0 + D_1 z + \ldots\ldots + D_{r-1} z^{r-1} + D_r z^r} = \frac{\sum_{i=0}^{r} C_i z^i}{\sum_{j=0}^{r} D_j z^j} \quad \ldots\ldots (2)$$

Where

$A_i (0 \le i \le n-1); B_j (0 \le j \le n)$ & $C_i (0 \le i \le r)$; $D_j (0 \le j \le r)$ are scalar constants.

The derivation of successful reduced model $R(s)$ for the original higher order model $G_n(s)$ is done by minimizing the error index 'E', known as ISE, employing Generalized Least Squares method for denominator and Differential Evolution Algorithm for numerator and is given by:

$$E = \int_0^\infty [G_n(t) - R(t)]^2 dt \quad \ldots\ldots\ldots (3)$$

ISSN (Online) 2321-2004
ISSN (Print) 2321-5526
INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH IN ELECTRICAL, ELECTRONICS, INSTRUMENTATION AND CONTROL ENGINEERING
Vol. 3, Issue 5, May 2015

Where $G_n(t)$ and $R(t)$ are the unit step response of original and reduced order systems.

III. DIFFERENTIAL EVOLUTION

A. OVERVIEW

Let f(x) be a function of n-parameter vector

$$x=[x_1, x_2 \ldots x_n]^T \quad (4)$$

Determine the values of the parameter vector within the intervals

$$x_k \in (x_k^-, x_k^+) , k=1,2,\ldots,n \quad (5)$$

which minimize the cost function f(x).

Differential evolution (DE) is a simple evolutionary algorithm for parameter optimization [9]. The most distinct feature of DE is that it mutates vectors by adding weighted, random vector differentials to them. Usually, the performance of a DE algorithm depends on three variables: the population size N_p, the mutation scaling factor F_s, and the crossover rate C_r. In applying a DE algorithm to solve the above parameter optimization problem, it starts by generating a population of N_p real valued n-dimensional vectors whose initial parameter values being chosen at random from within bounds set by the user. This population undergoes evolution in a form of natural selection. In every generation, each vector in the population becomes a target vector .Each target vector crossovers with a donor vector, which is generated by mutating a randomly-selected population vector with the difference between two randomly-selected population vectors, in order to produce a trial vector. If the cost of the trial vector is less than that of the target, the target is replaced by the trial vector in the next generation.

B. POPULATION INITIALIZATION

The DE algorithm is started with generating a population of N_p real-valued n-dimensional vectors

$$x_j= [x_{j,1}, x_{j,2} \ldots x_{j,n}]^T , j=1,2,\ldots N_p \quad (6)$$

whose initial parameter values are chosen at random from within user-defined bounds

$$x_{j,k} \in [\underline{x}_k, \overline{x}_k] , k=1,2,\ldots,n \quad (7)$$

It is noted that the search region or interval $[\underline{x}_k, \overline{x}_k]$ set by a user for the parameter x_k may be smaller than, but within, the allowable interval (x_k^-, x_k^+) in the case $x_k^-=-\infty$ of and/or $x_k^+=\infty$. Once the initial population is generated, the cost of each population vector is evaluated and stored for future reference.

C. MUTATION

Mutation is an operation that adds a vector differential to a population vector. In the DE algorithm, a population vector x_α is mutated into $z=[z_1,z_2,\ldots z_n]^T$ by adding to x_α the weighted difference of two randomly selected but different population vectors x_β and x_γ , i.e.,

$$z=x_\alpha+F_s*(x_\beta + x_\gamma) \quad \ldots\ldots (8)$$

where F_s is a scaling factor in the interval (0,2). In the event that mutation causes a parameter z_k to shift outside the allowable interval (x_k^-, x_k^+) then z_k is set to x_k^- if $z_k< x_k^-$, and x_k^+ if $z_k> x_k^+$. The mutated vector z will be used as a donor vector for producing a trial vector.

It is noted the mutation scheme (8) makes the DE a local optimizer because the vector differentials generated by a converging population eventually become infinitesimal.

D. CROSSOVER

Crossover is used to generate a trial vector by replacing certain parameters of the target vector by the corresponding parameters of a randomly generated donor vector. The crossover rate C_r determines when a parameter should be replaced. The process of producing a trial vector $x_t=(x_{t,1},\ldots x_{t,n})^T$ from the target vector $x=(x_1,\ldots x_n)^T$ and the donor vector z is begin with generating a set of n random numbers $\{r_1, r_2, \ldots r_n\}$ which are distributed uniformly in the interval (0,1). Next, a set of non uniform binary sequence $b_1,b_2\ldots b_n$ is generated by letting

$$b_i= \begin{cases} 1 \ if \ r_i \le C_r \\ 0 \ otherwise \end{cases} \quad i=1,2,\ldots n$$

Then each element $x_{t,k}$ of the trial vector x_t is taken as

$$x_{t,k}= \begin{cases} x_k \ \ for \ b_k = 1 \\ z_k \ \ for \ b_k = 0 \end{cases} \quad (9)$$

Once the trial vector has been determined, its cost is evaluated and compared with that of the corresponding target vector. The target vector will be replaced by it in the next generation if its cost is larger than that of the trial vector.

E. SEARCH-SPACE EXPANSION SCHEME

In the case that the initial search region is set excessively large, the convergence of DE search may become very slow and is prone to get trapped in a local optimum. On the other hand, if the initial search region is set too small, it may fail to locate the true optimum, though the mutation formula with a proper scaling factor F_s allows the DE algorithm to locate the minimum that lies outside the initial search region.

Hence, in the absence of the exact knowledge about the proper and finite interval within which the true optimum parameter locates, it is desired to allow the DE algorithm to expand the search space dynamically. Toward this end, we incorporate in the DE algorithm a search space expansion scheme. The implementation of this scheme for the cases of negative allowable lower bound $\underline{x}_k<0$ and positive allowable upper bound $\overline{x}_k>0$ is described as follows.

At the end of every N_c generations of population evolution, we examine each parameter in the vector x^* that has the lowest cost. If the current upper bound $\overline{x}_k$ of the k-th parameter x_k^* is positive and $\overline{x}_k \ge x_k^* > \overline{x}_k/2$, then the upper bound $\overline{x}_k$ get doubled. In the case where is still greater than the doubled upper bound, the upper bound $\overline{x}_k$ is set to γx_k^* , where $\gamma > 1$ is a user-set expansion factor. The selection of the search-space expansion factor γ depends on the size of the initial search space. Usually, it is set to 2 for a small initial search space and 1.2 for a large initial search space.

ISSN (Online) 2321-2004
ISSN (Print) 2321-5526
INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH IN ELECTRICAL, ELECTRONICS, INSTRUMENTATION AND CONTROL ENGINEERING
Vol. 3, Issue 5, May 2015

TABLE 1: VALUES OF THE DE ALGORITHM VARIABLES USED IN EXAMPLES 1 AND 2

Parameters	Values
Population Size , N_P	60
Maximum number of Generations , N_i	400
Search – space checking period , N_C	10
Search – space expansion factor , γ	1.2
Crossover constant , C_r	0.6
Mutation scaling factor , F_s	1.2

In the event that any parameter search interval has been expanded, N_P population vectors are generated from the expanded portion of search space and their costs are evaluated.

A new initial population of N_P population vectors is then formed by picking up the better population vectors among the N_P evolved population vectors and the N_P newly generated population vectors. Then this new initial population evolves with the expanded search space.

F. SELECTION

The cost of each trial vector x_t is compared with that of its parent target vector x_i . If the cost of the target vector x_i is lower than that of the trial vector, the target is allowed to advance to the next generation.

Otherwise, the target vector is replaced by the trial vector in the next generation.

Differential Evolution Algorithm

1) Choose population size N_p, mutation scaling factor F_s, search-space checking period N_c, space expansion factor γ, and the maximum number of iterations N_i .
2) Specify the initial search intervals $[\underline{x}_k, \overline{x}_k]$, k=1,2,…n .
3) Generate randomly N_p,vectors x_j, from the specified search space X= $\left\{x: x_k \in [\underline{x_k}, \overline{x_k}]\right\}$, k=1,2,…n and evaluate their costs C_j, j=1,2…N_p.
4) Initialize the iteration index I=1 .
5) Set the initial target index j=1 .
6) Choose at random three different integers α,β and γ from the set { j=1,2…N_p}
7) Mutate the vector x_α to z=$x_\alpha + F_s$*($x_\beta - x_\gamma$).
8) Generate a non uniform binary Z-sequence b={b_1,b_2,…b_n}.
9) Obtain the trial vector x_t from the mutated vector Z and the target vector x_j using crossover scheme (9).
10) Evaluate the cost of the trial vector x_t and denote it by C_t . If $C_t > C_j$ then $x_j^{new} = x_j$ and $c_j^{new} = c_j$, otherwise $x_j^{new} = x_t$ and $c_j^{new} = c_t$
11) Increment the target index j by 1. If $j < N_p$ then go to Step 6.
12) Do the replacement $x_j = x_j^{new}$ and $c_j = c_j^{new}$ for j=1,2…N_p .
13) If i is a multiple of N_c , then execute the search-space expansion scheme.
14) Increment the iteration index I by 1. If I <N_i , then go to Step 5.
15) Stop.

III. ORDER REDUCTION BY GENERALISED LEAST-SQUARES METHOD

Here, the model reduction by generalized least-squares method suggested in [11] is discussed in brief

Consider the n[th] order system transfer function, given by :

$$G_n(s) = \frac{b_0+b_1 s+\cdots+b_{n-1}s^{n-1}}{a_0+a_1 s+\cdots+a_{n-1}s^{n-1}+a_n s^n} \quad (10)$$

For a reduced r[th] order model of $G_n(s)$ in (1), given by :

$$G_r(s) = \frac{d_0+d_1 s+\cdots+d_{r-1}s^{r-1}}{e_0+e_1 s+\cdots+e_{r-1}s^{r-1}+s^r} \quad (11)$$

which retains (r + t) time moments and (r − t) Markov parameters (0 ≤ t ≤ r) the coefficients e_k , d_k in (2) are derived from following set of equations :

$$\left.\begin{aligned} d_0 &= e_0 c_0 \\ d_1 &= e_1 c_0 + e_0 c_1 \\ &\vdots \quad \vdots \quad \vdots \\ d_{r-1} &= e_{r-1}c_0 + \cdots\cdots + e_0 c_{r-1} \\ 0 &= e_{r-1}c_1 + \cdots\cdots + e_1 c_{r-1} + e_0 c_r \\ 0 &= e_{r-1}c_2 + \cdots\cdots + e_1 c_r + e_0 c_{r+1} \\ &\vdots \quad \vdots \quad \vdots \\ 0 &= e_{r-1}c_t + \cdots\cdots + e_1 c_{r+t-2} + e_0 c_{r+t-1} \end{aligned}\right\} \quad (12)$$

and

$$\left.\begin{aligned} d_{r-1} &= m_1 \\ d_{r-2} &= m_1 e_{r-1} + m_2 \\ &\vdots \quad \vdots \quad \vdots \\ d_t &= m_1 e_{t+1} + m_2 e_{t+2} + \cdots + m_{r-t} \end{aligned}\right\} \quad (13)$$

where, the c_i and m_j are the time moment proportionals and Markov parameters of the system, respectively. Elimination of the d_j (j = t, t +1, ..., r −1) in (13) by substituting into (12) gives the reduced denominator coefficients as the solution of :

$$\begin{pmatrix} c_{r+t-1} & c_{r+t-2} & \cdots & c_t \\ c_{r+t-2} & c_{r+t-3} & \cdots & c_{t-1} \\ \vdots & \vdots & & \vdots \\ c_{r-1} & c_{r-2} & \cdots & c_0 \\ c_{r-2} & c_{r-3} & \cdots & -m_1 \\ \vdots & \vdots & & \vdots \\ c_t & c_{t-1} & \cdots & -m_{r-t+1} \end{pmatrix} \begin{pmatrix} e_0 \\ e_1 \\ \vdots \\ e_{r-1} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ m_{r-t} \end{pmatrix} \quad (14)$$

Or , H e = m in matrix vector form.

If the denominator given by e in (14) is unstable, or has a singularity, then the next Markov parameter m_{r-t+1} can be assumed to be matched by extending (13) with the equation :

$d_{t-1} = m_1 e_{t+2} + m_2 e_{t+1} + \ldots + m_{r-t+1}$ (15)

This in effect adds another row to the H matrix and the m vector in (14), given by :

[c_{t-1} c_{t-2}…c_0 $-m_1$ $-m_2$…$-m_{r-t}$] and [m_{r-t+1}],

respectively. Calculation of e from this non square system of equations can only be done in the least- squares sense, i.e. :

$e = (H^T H)^{-1} H^T m$ (16)

If the denominator polynomial is still not adequate, then the H matrix and the m vector may again be extended by

ISSN (Online) 2321-2004
ISSN (Print) 2321-5526

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH IN ELECTRICAL, ELECTRONICS, INSTRUMENTATION AND CONTROL ENGINEERING
Vol. 3, Issue 5, May 2015

assuming a matching of the next Markov parameter in the sequence and (16) is solved for the new estimate of e.

IV. ILLUSTRATIVE EXAMPLES

Example 1: Consider an Eighth order system transfer function

$$G(z) = \frac{0.4209z^7 + 0.2793z^6 - 0.0526z^5 + 0.038z^4 - 0.1291z^3 - 0.0656z^2 + 0.011z - 0.0015}{z^8 - 0.4209z^7 - 0.2793z^6 + 0.0526z^5 - 0.038z^4 + 0.1291z^3 + 0.0656z^2 - 0.011z + 0.0015}$$

Apply linear transformation , z = p+1

$$G(p) = \frac{0.4209p^7 + 3.2256p^6 + 10.4621p^5 + 18.695999p^4 + 19.8144p^3 + 12.277499p^2 + 4.0036p + 0.5004}{p^8 + 7.5791p^7 + 24.774399p^6 + 45.537903p^5 + 51.304005p^4 + 36.185604p^3 + 15.722501p^2 + 3.9964p + 0.4996}$$

Denominator by generalized least square method:
No. of time moments=3
t[0]=1.001601,t[1]=0.001603,t[2]=-6.958732
No. of markov parameters=1
m[1]=0.4209

The second order reduced denominator using generalized least square method in p-domain is
D(p) = p^2+0.419997p+0.144031

Numerator by DE technique:
No. of iterations=100
Swarm size=50
pl=0.1, pu=2

The second order reduced numerator using differential evolution technique in p domain is
N(p)=0.387715p+0.126863
The proposed second order reduced model obtained is
R(p)=$\frac{0.387715\,p+0.126863}{p^2+0.419997\,p+0.144031}$ (ISE=0.013664)

Apply inverse linear transformation , p=z-1
R(z)= $\frac{0.387715\,z-0.260852}{z^2-1.580003\,z+0.724034}$

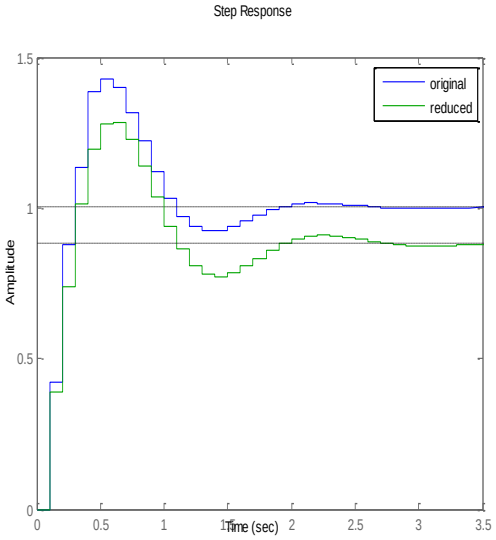

Fig. 1(a) Comparison of step responses of $G(z)$ and $R(z)$

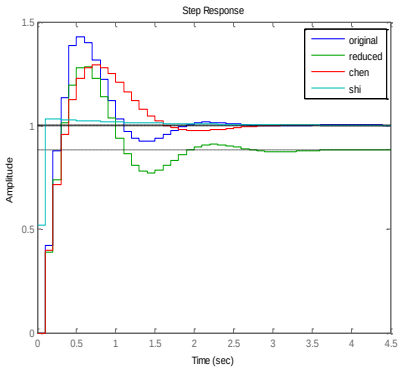

Fig. 1(b) Comparison of step responses of $G(z)$ and $R(z)$ with other reduction techniques

Example 2: Consider an fifth order system transfer function

G(z)= $\frac{3.75z^5-13.0875z^4+17.80313z^3-11.7156z^2+3.688103z-0.4358109}{z^6-4.25z^5+7.3625z^4-6.626875z^3+3.250838z^2-0.8181562z+0.0817593}$

Apply linear transformation ,z=p+1

G(p)= $\frac{3.75p^5+5.6625p^4+2.953131p^3+0.668732p^2+0.06617p+0.002262}{p^6+1.75p^5+1.1125p^4+0.323126p^3+0.045216p^2+0.002896p+0.000066}$

Denominator by generalized least square method: No. of Time moments = 4
t[0]=34.272728, t[1]=-501.269989, t[2]=8647.485352, t[3]=-159075.3125

No. of Markov parameters = 0
The second order reduced denominator using generalized least square method in p-domain is

D(p)=p^2+0.225215p+0.009092
Numerator by DE technique:
No. of iterations = 120
Swarm size = 100
pl = 0.009092 , pu = 3.017862

The second order reduced numerator using differential evolution technique in p-domain is
N(p)=3.017862p+0.306552

The proposed second order reduced model obtained is
R(p)= $\frac{3.017862\,p+0.306552}{p+0.225215\,p+0.009092}$ (ISE=0.608587)
Apply inverse linear transformation, p= z-1
R(z)= $\frac{3.017862\,z-2.71131}{z^2-1.774785\,z+0.783877}$

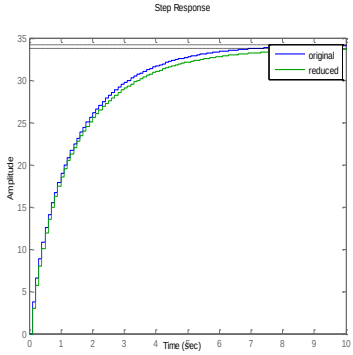

Fig. 2(a) Comparison of step responses of $G(z)$ and $R(z)$

ISSN (Online) 2321-2004
ISSN (Print) 2321-5526

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH IN ELECTRICAL, ELECTRONICS, INSTRUMENTATION AND CONTROL ENGINEERING
Vol. 3, Issue 5, May 2015

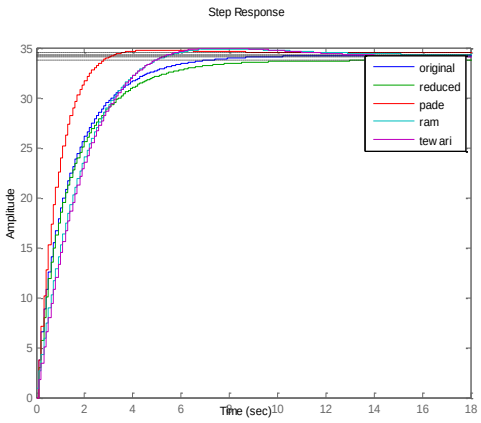

Fig. 2(b) Comparison of step responses of $G(z)$ and $R(z)$ with ther reduction techniques

IV. CONCLUSION

The authors proposed a mixed algorithm for reducing the order of linear dynamic SISO systems. In this algorithm, the concept of order reduction by generalized least squares method has been improved and employed to determine the coefficients of reduced denominator while the coefficients of reduced numerator are obtained by minimizing the integral square error between the transient response of original and reduced models using DE technique pertaining to unit step input. The algorithm is implemented in C-language. The matching of the unit step response is assured reasonably well in the algorithm. The algorithm is simple and computer oriented. A comparison of step responses for the proposed reduction algorithm and the other well known existing order reduction technique is shown from which it is clear that the proposed algorithm compares well with the other techniques of model order reduction.

REFERENCES

[1]. R.Genesio and M. Milanese, "A note on the derivation and use of reduced order models", IEEE Trans. Automat. Control, Vol. AC-21,No. 1, pp. 118-122, February 1996..
[2]. M. Jamshidi, "Large Scale Systems Modeling and Control Series", New York, Amsterdam, Oxford, North Holland, Vol. 9, 1983..
[3]. G. Parmar, R. Prasad and S. Mukherjee, "Order reduction of linear dynamic systems using stability equation method and GA", World Academy of science engg. and tech Vol-26, 2007.
[4]. C.S.Hsien, C.Hwang 1989,"Model order Reduction of Continous time Systems using a modified Routh Approximation Method" IEE Proceedings control theory appl.136:pp no:151-156
[5]. Y.Shamash,"Approximations of linear time invariant systems", Proc.Conf. on Pade approximants and their Applications, P. R. Graves-Morris(Ed.), London Academic,1973.
[6]. A. Bultheel and M. V. Barel, "Pade techniques for model reduction in linear system theory, A survey", Journal of Computational and Applied Mathematics, Vol. 14, pp.401-438,1986.
[7]. G. A. Baker, Essentials of Pade approximation, Academic Press, New York, 1975.
[8]. Z. Michalewicz, Genetic Algorithms + Data Structures = Evolution Programs, 3rd ed. New York: Springer-Verlag, 1999.
[9]. T. N. Lucas and A. R. Munro, "Model reduction by generalised Least squares method",Electronic Letters, Vol. 27, No. 15, pp. 1383-1384, 1991.
[10]. E. G. Collins, S. S. Ying, W. M. Haddad, and S.Richter, "An efficient, numerically robust homotopy algorithm for H2 model reduction using the optimal projection equations," Math.Modeling Syst.: Meth.,Tools Applicant. Eng.Rel. Sci., vol. 2,no. 2, pp. 101–133, 1996
[11]. K. Price, "Differential evolution: A fast and simple numerical optimizer," in Proc.North American Fuzzy Information Processing Conf., 1997, pp.524-527.
[12]. M.Strens & A.Moore ,"Policy search using paired comparisons", journal of machine learning research,vol-3,pp:921-950,2002
[13]. H.A.Abbass, Rahul.Sarker,&Charles Newton,"PDE: A Pareto-frontier differential evolution approach for multi-objective optimization problems ",proceedings,2001
[14]. R.Storn,"Differential Evolution,a simple efficient heuristic strategy for global optimization over continuous spaces",journal of global optimization, vol.2,Dordrecht,pp no:341- 359,1997.