NOVEL MODEL ORDER REDUCTION AND CONTROLLER DESIGN TECHNIQUE USING BIG BANG BIG CRUNCH OPTIMIZATION ALGORITHM

A DISSERTATION

Submitted in partial fulfillment of the
requirements for the award of the degree
of
MASTER of TECHNOLOGY
in
ELECTRICAL ENGINEERING
(with specialisation in Systems and Control)

Submitted by

SHIVANAGOUDA BIRADAR

(Enrl No.:14530016)

Under the guidance of

Dr. Yogesh V. Hote

DEPARTMENT OF ELECTRICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY ROORKEE
ROORKEE - 247 667 (INDIA)

MAY 2016

CANDIDATE'S DECLARATION

I hereby declare that this research report entitled **NOVEL MODEL ORDER REDUCTION AND CONTROLLER DESIGN TECHNIQUE USING BIG BANG BIG CRUNCH OPTIMIZATION ALGORITHM**, submitted to the Department of Electrical Engineering, Indian Institute of Technology Roorkee, India, in partial fulfillment of the requirements for the award of the Degree of Master of Technology in Electrical Engineering is an authentic record of the work carried out by me during the period from July 2015 to May 2016 under the supervision of **Dr. Yogesh Vijay Hote, Department of Electrical Engineering, Indian Institute of Technology, Roorkee.** The matter presented in this report has not been submitted by me for the award of any other degree of this institute or any other institute.

Date:

Place: Roorkee

Shivanagouda Biradar

CERTIFICATE

This is to certify that the above statement made by the candidate is true to the best of my knowledge and belief.

Dr. Yogesh Vijay Hote
Associate Professor
Department of Electrical Engineering
Indian Institute of Technology Roorkee

ABSTRACT

This dissertation focuses on a meta-heuristic optimization algorithm i.e. big bang big crunch optimization (BBBC) algorithm and major setbacks in BBBC algorithm with respect to its conceptual and working structure. A modified BBBC optimization algorithm is proposed, which works better than original BBBC. But, it is observed that BBBC and modified BBBC like many other meta-heuristic optimization algorithm suffers from the problem of getting trapped in local minima. Therefore, modified BBBC is combined with chaos which effectively enhances the searching efficiency and greatly improves the searching quality. These algorithms validity is quantified using various benchmark function.

Further, this thesis, contributes various results, techniques and focuses on application of BBBC in areas of System and Control. Starting with Model Order Reduction (MOR), which is an integral part of System Engineering. MOR techniques have proved to be an important technique for accelerating time-domain simulation in a variety of CAD tools for highly complex system and controller design. There are various reduction techniques available in literature and most of them are either complex i.e. they are too difficult to understand while other techniques work for particular class of problems. In this report, a novel MOR technique has been proposed using BBBC and time moment matching method, which works for many class of problems. Now, moving onto field of Control Engineering, utility of this algorithm for controller design has been elaborated. In this method, a multi-objective function has been formulated and BBBC is used as an optimization tool for fine tuning the PID controller. Above work i.e. MOR and controller design have been validated on automatic voltage regulator system.

Another contribution of this thesis is study of utility of statistical methods in area of Control System and Optimization. As it is known that BBBC is a relatively new optimization technique, before which, many famous techniques like PSO and GA are widely used in all field of engineering and talked about in optimization society. But question always raises which one of these algorithms are better in respect to solution finding capability (effectiveness) and computational efficiency. In this report we have used inferential statistics as a tool to analyze this problem and bring out a concrete conclusion in this regard.
At last we have used Taguchi method, a statistical technique, combined with BBBC to fine tune the controller parameters.

Acknowledgements

My deepest gratitude to my advisor Associate Prof. Yogesh V Hote for his invaluable support, advice and guidance in all stages of this work. His detailed and constructive comments and our fruitful discussions constituted the basis of this dissertation. His ideas and insights went far beyond the subject of order reduction to being advices for a successful academic research.

I am grateful to my seniors Sahaj Saxena, Sandeep Hanwate, V Siddartha, K Raju, P. Sudarshana, Pushkar P Arya and my colleague T.H.V.V.R.N Sunil in Controls and Robotics Laboratory for their assistance, scientific and non-scientific discussions, and for the friendly environment they provided.

My time at IIT Roorkee was made enjoyable and a learning experience in large part due to the many friends and groups that became a part of my life. I am grateful for time spent with roommates and friends, for my backpacking buddies and our memorable trips we took at time when we were stressed out, and for many other people and memories.

I owe a lot to my family who unremittingly supported me during my years of study. I would like to pay high regards to my mother and father for being a true motivator through out my life. Their relentless backing made this work possible and I would like to dedicate it to them.
Above all, I owe it all to Almighty God for granting me health, wisdom, and strength.

Shivanagouda Biradar

Contents

List of Figures

List of Tables

Chapter 1

Big Bang Big Crunch Algorithm (BBBC)

1.1 Fundamentals of BBBC

Big Bang-Big Crunch[1] is basically an optimization method. This algorithm was proposed by Erol and Eksin in 2006, which is inspired by theories of evolution of universe. BBBC essentially consist of two phases: a big bang phase (BBP) and a big crunch phase (BCP). In BBP, candidate solutions are uniformly disseminated over search space, where in search space is limited by boundary constraints as in most metaheuristic optimization technique. This uniform randomness is equivalent to energy dissipation in nature[1]. The BBP is followed by BCP which can be visualised as transformation from disordered state of energy (uniform randomness) to ordered state of energy. The big crunch phase acts as a concurrence operator which has only one output for many inputs or more precisely, randomly distributed solutions are drawn into order, which can be named as center of mass. Here, the term mass refers to the inverse of the fitness function value[1]. The centre of mass $\vec{C}(\vec{x})$ is a function of position of each candidate (position vector) in a designed search space and for k^{th}iteration it is computed using formula.

$$\vec{C}^k(\vec{X}^k) = \frac{\sum_{i=0}^{N} \frac{\vec{x}_i^k}{f_i^k}}{\sum_{i=0}^{N} \frac{1}{f_i^k}} \tag{1.1}$$

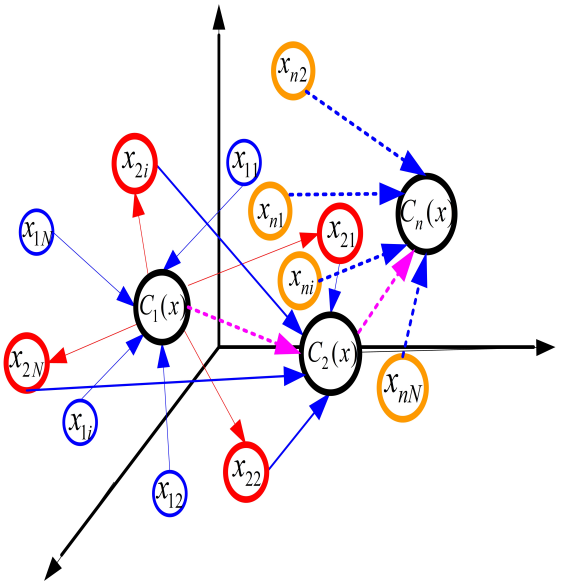


FIGURE 1.1: Illustration of BBBC.
Blue color represents candidates moving to organised states, Red color represents center of mass moving to disorganised state and Pink color represent movement of center of mass.

such that $\vec{x}_i^k \in \vec{X}^k$, wherein a set $\vec{X}^k = \{\vec{x}_i^k \,|\vec{x}_i^k \in \mathbb{R}, 1 \leq i \leq N, x_{\min} \leq x_i^k \leq x_{\max}\}$. In (1.1), $\vec{x}_i^k$ is i^{th} candidate in k^{th} iteration of n-dimensional search space and f_i^k is treated as an objective function value or fitness function value corresponding to i^{th} candidate of k^{th} iteration. More clear representation is shown in (1.2).

$$f_i^k \in f_o(\vec{X}^k) = \{f_i^k | f_i^k \in \mathbb{R}, 1 \leq i \leq N\} \tag{1.2}$$

and N is the population size in BBP. Population size must be optimally chosen which depends on the range of search space and number of iterations. After BCP, new candidates ($(k+1)^{th}$ iteration) are generated in designed search space, to be used for BBP based on the knowledge of $\vec{C}^k(\vec{X}^k)$, expressed as

$$\vec{x}_i^{k+1} = \vec{C}^k(\vec{X}^k) + \delta_i \tag{1.3}$$

such that $\vec{x}_i^{k+1} \in \vec{X^{k+1}}$ wherein a set $\vec{X^{k+1}} = \{\vec{x}_i^{k+1} \,\big|\vec{x}_i^{k+1} \in \mathbb{R}, 1 \leq i \leq N, x_{\min} \leq x_i^{k+1} \leq x_{\max}\}$. δ_i (spread/variance factor) is deviation of newly generated candidates with respect to $\vec{C}^k(\vec{X}^k)$ and is calculated using following equation:

$$\delta_i = \frac{r_i \alpha \left(x_{\max} - x_{\min}\right)}{K} \tag{1.4}$$

In equation (1.4), α is the constant parameter generally taken to be in between $[0, 1]$. This parameter α limits the size of the solution space; r_i is a random number from a

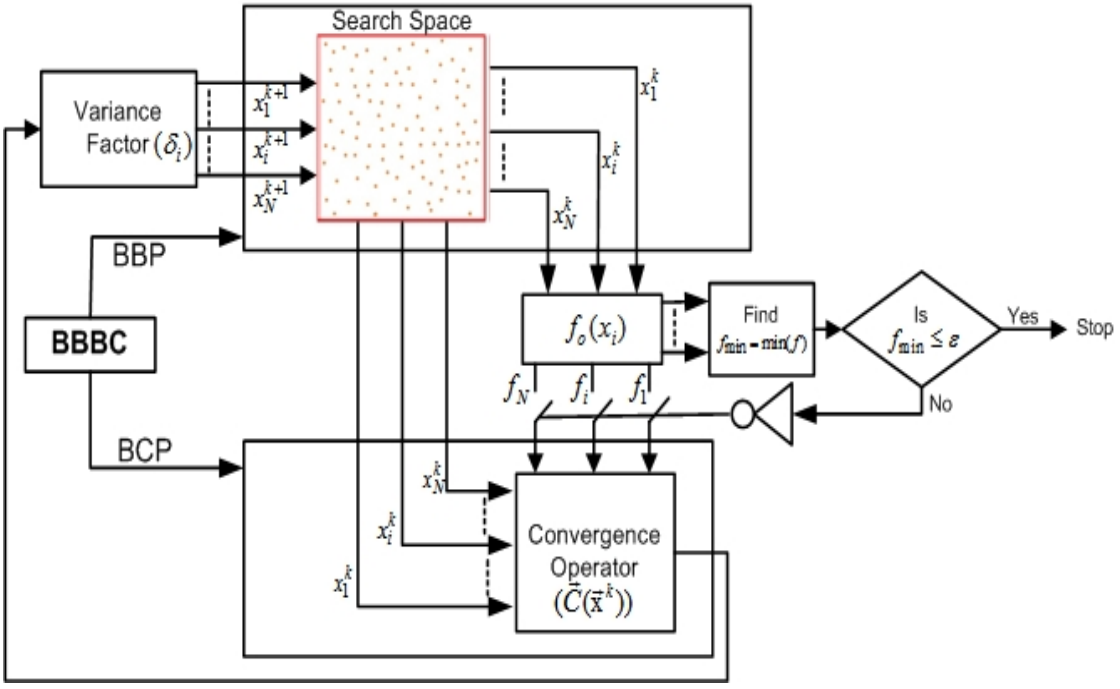


FIGURE 1.2: Schematics Big bang big crunch algorithm

standard normal distribution which changes for each candidate such that $r_i \in (0, 1]$. The greater the value of r_i the more scattered the candidate solutions are around $\vec{C}^k(\vec{X}^k)$; $x_{\max}$ and $x_{\min}$ are the upper and lower bounds on the values of the optimization problem variables, and K is a variable which increases with step size of one. This is done for better and faster convergence to solution. After second explosion, $\vec{C}^{k+1}(\vec{X}^{k+1})$ is calculated. These successive explosions, i.e., BBP, and contraction phases, i.e., BCP are carried repeatedly until a stopping criterion has been met. This process is diagrammatically represented in Fig.1.1. Pseudo code for BBBC is presented below to illustrate coding methodology. In order to get clearer idea about basic BBBC algorithm, it is explained using flowchart which is shown in Fig.1.2.

Pseudocode

Step 1: Initialize r, α, N and $k(iteration) = 0$;

Step 2: Generate population of size N, such that population set

$\vec{X}^k = \{\vec{x}_1^k, \vec{x}_2^k, .., \vec{x}_i^k, .., \vec{x}_N^k\} \in [x_{\max}, x_{\min}], \forall i = 1 \ldots N$;

Step 3: Evaluate objective function for each particle

$f_i^k = f_o(x_n^k), n = 1 \ldots N$, where f_o is objective function.

Step 4: Calculate the center of mass $\vec{C}^k(\vec{X}^k)$ as defined in equation (1.1).

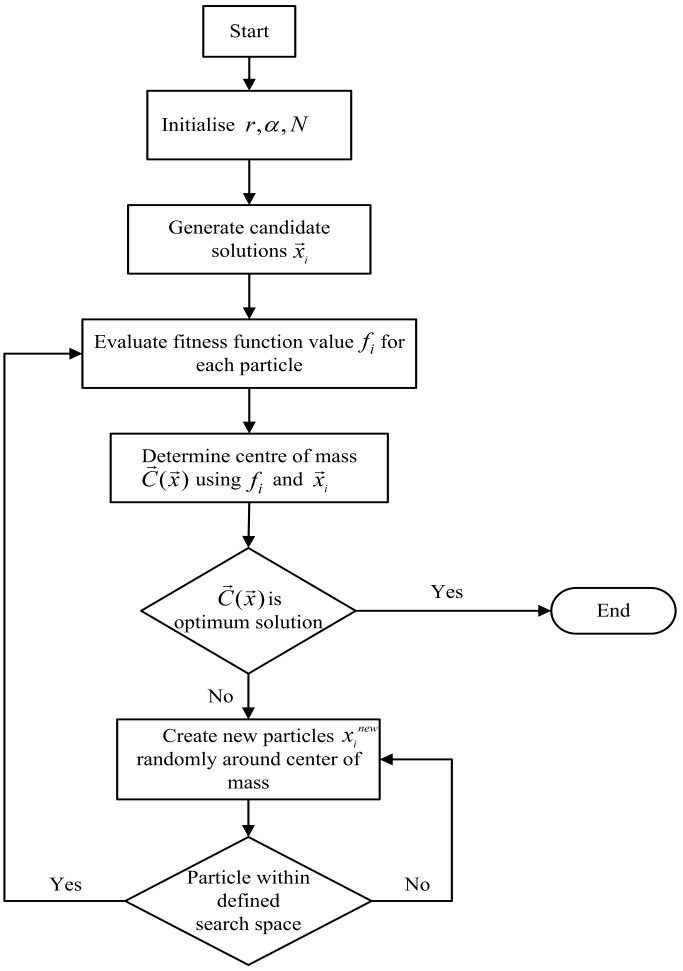


FIGURE 1.3: Schematics of BBBC algorithm

Step 5: Find and store optimum value (say minimum).

$leastf_i = \min(f^k)$,

$if(leastf^k \leq \varepsilon)$, then jump to step 9.

Otherwise go to step 6;

Step 6: $k \leftarrow k + 1$;

Step 7: Generate new solution around center of mass

$\vec{X}^k = \{\vec{x}_1^k, \vec{x}_2^k, \ldots, \vec{x}_N^k\} = \vec{C}^{k-1}(\vec{X}^{k-1}) + \delta$

Step 8: Go to step 3.

Step 9: At $k = iter$, ($iter$ is an instant of iteration where it is assumed that optimized value or solution is obtained). Return optimized $\vec{C}^{iter}(\vec{X}^{iter})$ corresponding to $\vec{X}^{iter} = \{\vec{x}_1^{iter} \ldots \vec{x}_i^{iter} \ldots \vec{x}_N^{iter}\}$.

1.2 Issue's with BBBC

BBBC does not imitate the big bang big crunch theory to the fullest. According to the scientific theory, universe was born out of singularity which is commonly known as *big bang singularity*. After big bang the (unstable) hot mass of bodies traveling (candidate solutions) at very high velocities under gravity took millions of years to cool down (stable). As of now universe is in stable state. At some point of time the ever expanding universe will experience big crunch moving back to singularity (ultimate solution or center of mass). However, we are interested in the process of big bang and big crunch. It can be very well visualised that after big bang large masses of bodies were traveling at high velocities under influence of gravity. This process structure is not imbibed in original BBBC, i.e. original BBBC does not considers velocity and gravity which is inherent to big bang. Modified BBBC takes into account both velocity and gravity. These moving masses in stable state collided (inelastic collision) to form planets. As mentioned above, universe will end with singularity, but formation of singularity will need perfect condition in terms of mass of bodies, velocity, energy of bodies etc. Again this part of process has also been neglected in original BBBC. The Modified BBBC considers conditioned (best) particles or swarms for formation of solution or singularity.

1.3 Proposed modified BBBC (MBBBC)

Above section discusses about issues in process structure of original BBBC. In this section, solution(/modification) to these issues are elucidated and logical reasoning have been presented to validate these modification. Initially, randomly generated swarms are mapped one to one to randomly formed velocities i.e. a uniform distribution function U is defined such that $U : \vec{X}^{k=0} \mapsto \vec{V}^{k=0}$ defined on closed set $\vec{X}^{k=0}$ of real plane. Now say, in k^{th} iteration swarm $\vec{x}_i^k$ is mapped one to one to v_i^k i.e. a uniform distribution on $\vec{X}^k$ is a function $U : \vec{X}^k \mapsto \vec{V}^k$ defined on closed set $\vec{X}^k$ of real plane. A center of mass $(\vec{C}^k(\vec{x}^k))$ is calculated in BCP using equation (1.5).

$$\vec{C}^k(\vec{x}^k) = \frac{\sum_{i=1}^{N} \frac{\vec{x}_i^k}{f_i^k}}{\sum_{i=1}^{N} \frac{1}{f_i^k}} \tag{1.5}$$

Crunching of all particles into the center of mass is considered as an inelastic collision and hence the center of mass has a velocity ($\vec{V}_C^k$) which is given in (1.6) as

$$\vec{V}_C^k = \frac{\sum_{i=1}^{N} \frac{\vec{v}_i^k}{f_i^k}}{\sum_{i=1}^{N} \frac{1}{f_i^k}} \tag{1.6}$$

$\vec{v}_i^k$ is the velocity and f_i^k is the fitness value of i^{th} candidate in k^{th} iteration respectively. After creation of center of mass and calculating its velocity, now in $(k+1)^{th}$ iteration fresh swarms $\vec{x}_i^{k+1} \in \vec{X}^{k+1}$ are generated in search space, to be used for BBP based on the knowledge of center of mass and velocity of center of mass in k^{th} iteration. New swarms $\vec{x}_i^{k+1}$ are generated using formula (1.7):

$$\vec{x}_i^{k+1} = \vec{C}^k(\vec{x}^k) + \vec{v}_i^{k+1} \tag{1.7}$$

where $\vec{v}_i^{k+1}$ i.e. velocity of i^{th} candidate in $(k+1)^{th}$ iteration and is calculated using formula (1.8):

$$\vec{v}_i^{k+1} = \vec{V}_C^k.g \tag{1.8}$$

g in (1.8) can be defined as *inverse adaptive gravity factor* and is expressed in (1.9) as:

$$g = \left(\frac{f_i^k - f_{\min}^k}{f_{\max}^k - f_{\min}^k} \right) \tag{1.9}$$

or

$$g = \left(\frac{f_i^i - f_{\min}^k}{f_{avg}} \right) \tag{1.10}$$

$\vec{x}_i^{k+1}$ and $\vec{v}_i^{k+1}$ are again mapped one to one i.e. a uniform distribution defined on $\vec{X}^{k+1}$ is a function $U : \vec{X}^{k+1} \mapsto \vec{V}^{k+1}$ defined on closed set $\vec{X}^{k+1}$ of the real plane. And f_i^k is the fitness value of i^{th} candidate in k^{th} iteration, f_{min}^k and f_{max}^k are minimum and maximum fitness value among N swarms in k^{th} iteration respectively. This process of big crunch and big bang is carried out till best solution or more appropriately best center of mass $\vec{C}(\vec{x})$ is calculated. Detailed pseudocode is given below:

Pseudocode

Step 1: Initialise x_{max}, x_{min}, N, $k = 0$.

Step 2: Generate population of size N, such that population set
$\vec{X}^k = \{\vec{x}_1^k, \ldots \vec{x}_i^k, \ldots \vec{x}_N^k\} \in [x_{\min}, x_{\max}], \forall i = 1 \ldots N$

Step 3: Each candidate in population $\vec{x}_i^k$ is mapped one to one to randomly generated velocities $\vec{v}_i^k \in \vec{V}^k$. Such that $\vec{V}^k = \{\vec{v}_1^k, \dots \vec{v}_i^k, \dots \vec{v}_N^k\} \in [v_{\min}, v_{\max}], \forall i = 1 \dots N$

Step 4: Evaluate objective function for each particle

$f_i^k = f_0(x_n),\ n = 1 \dots N$

such that $f^k = \{f_1^k, \dots f_i^k \dots, f_N^k\}$,$\forall i = 1 \dots N$, where f_o is an objective function.

Step 5: Calculate the center of mass $\vec{C}^k(\vec{x}^k)$ as defined in equation (1.5).

Step 6: Find and store optimum value (say minimum),

$leastf_i = \min(f^k)$,

check $if(\text{leastf}_\text{i} \leq \varepsilon)$,

then jump to step 11.

Otherwise go to step 7.

Step 7: Calculate the velocity of center of mass $\vec{V}_C^k$ using expression (1.6).

Step 8: $k \leftarrow k + 1$.

Step 9: Generate new solution around center of mass $\vec{C}^{k-1}(\vec{x}^{k-1})$ using expression (1.7), such that $\vec{X}^k = \{\vec{x}_1^k, \dots \vec{x}_i^k, \dots \vec{x}_N^k\} \in [x_{\min}, x_{\max}], \forall i = 1 \dots N$. And the velocity term $v_i^{k=k+1}$ in equation (1.7) is calculated using equation (1.8).

Step 10: Go to step 4.

Step 11: At $k =\ iter$, return optimized $\vec{C}^{iter}(\vec{x}^{iter})$ corresponding to $\vec{X}^{iter} = \{\vec{x}_1^{iter} \dots \vec{x}_i^{iter} \dots \vec{x}_N^{iter}\}$.

1.4 Conceptual drawbacks in original BBBC

In this section, we will discuss various factors that were introduced in MBBBC in contrast to original BBBC. Further, advantages of MBBBC over original BBBC is also explained.

- From expression (1.3) and (1.7) we can infer that

$$\vec{x}_i^{k+1} \alpha \{x_{\max}^k - x_{\min}^k\} \tag{1.11}$$

$$x_i^{k+1} \alpha \{f_i^k - f_{\min}^k\} \tag{1.12}$$

  the expression (1.11) pertaining to original BBBC is tuned per iteration and its dependency is on x_{max} and x_{min} only. Hence, exploration capability of algorithm decreases while exploitation capability remains fairly unchanged. Expression (1.12) related to MBBBC, is tuned for each and every particle. Each of i^{th} particle's position at $k + 1^{th}$ iteration is decided based upon the fitness value value of i^{th} particle at k^{th} iteration. Particle whose fitness value is close to minimum fitness value ($f_{\min}^k$) will be properly tuned to move closer to $\vec{C}^k(\vec{x}^k)$, hence improving exploitation capability of algorithm. Particle whose fitness value is high as compared

to minimum fitness value ($f^k_{\min}$), expression (1.12) will be high and hence placing the new particle away from $\vec{C}^k(\vec{x}^k)$, improving exploration efficiency. Every partcle will have equally responsible role in finding best solution in contrast to original BBBC.

- From expression (1.4) and (1.8) it can be concluded that

$$\delta_i \alpha \frac{1}{K} \tag{1.13}$$

$$v_i^{k+1} \alpha \frac{1}{f^k_{\max} - f^k_{\min}} \tag{1.14}$$

or

$$v_i^{k+1} \alpha \frac{1}{f^k_{avg}} \tag{1.15}$$

δ_i and v_i^{k+1} in (1.13) and (1.14) or (1.15) pertaining to expression (1.4) and (1.8) decides the spread or variance of each particle around the center of mass. In original BBBC, K increases in step with iteration. The idea to introduce K is to reduce variance of particle with respect to center of mass as iteration increases, however this factor does not consider the state of each particle. Whereas in (1.14) terms f^k_{avg} and/or $(f^k_{\max} - f^k_{\min})$ collectively consider the state of particles and contributes for improvement in exploitation of solutions to very large extent.

- In expression (1.4) of original BBBC, it is observed that variance factor has two random factors to be taken care of i.e. α and r_i which effect the exploration and exploitation nature of algorithm, whereas $\vec{v}_i^{K+1}$ in MBBBC is free from any randomness and is deterministic at every iteration, hence making it more predictable, easy to understand and more efficient.

- From expression (1.9), the factor g is coined as *inverse adaptive gravity factor* and it works exactly as it is named. Consider for instance that f_i^k is nearer to f^k_{min}, then g would be small and hence $\vec{v}_i^{k+1}$ is also small, ultimately placing particles $\vec{x}_i^{k+1}$ close to center of mass and vice versa if difference between f_i^k and f^k_{min} is large.

1.5 Proposed chaos modified BBBC (CMBBBC)

In above section, we successfully improved various important aspect of original BBBC. But still it is seen that both algorithm i.e. original BBBC and MBBBC get stuck in

local optima, which is common to all widely used meta-heuristic algorithm like PSO, GA and ant colony optimization.

What is CHAOS?

Chaos can generally be understood as unpredictable or random behavior. Particularly, chaos has a characteristic of nonlinear systems, which is bounded unstable dynamic behavior, that exhibits sensitive dependence on initial condition and includes infinite unstable periodic motion [2].

From past many years, research on chaos is gaining impetus in physics, chemistry, biology and in nonlinear engineering systems. Due to its ease of implementation and ability to escape local optimum, concept of chaos in optimization techniques has aroused great interest in field of soft-computing. Hence proposed modified BBBC (MBBBC) is then combined with chaos (CMBBBC) which further shows excellent improvement in convergence rate and searching quality solutions.

Before proceeding to CMBBBC, we will develop an algorithm to be inserted into MBBBC, which is called as *Chaotic Velocity Adjustment (CVA)*. This algorithm i.e. CVA forms a inevitable part in development of proposed CMBBBC.

1.5.1 Chaotic velocity adjustment (CVA)

In this paper a well known Verhulst model [3], which displays sensitive dependence on initial condition is incorporated to develop CMBBBC. The Verhulst model is defined as follows.

- Continuous model

$$\frac{dx}{dt} = \mu * x(1-x), 0 \leq x_0 \leq 1 \tag{1.16}$$

- Discrete model

$$x_{n+1} = \mu * x_n(x_{\max} - x_{\min}), 0 \leq x_0 \leq 1 \tag{1.17}$$

Discrete Verhuslt model is used in evolutionary optimization algorithm, where μ is the driving or control parameter, x is a variable and $n = 0, 1, 2, \ldots$. Verhuslt model exhibits chaotic dynamics at $\mu = 3.56994$ and a minuscule variation in initial condition of chaotic variable x_0 results in substantial difference in long term behavior. Generally chaotic variable possesses properties like a) Irregularity b) Ergodicity c) Pseudo-randomness. The process of CVA can be defined using following chaotic logistic equation or verhulst

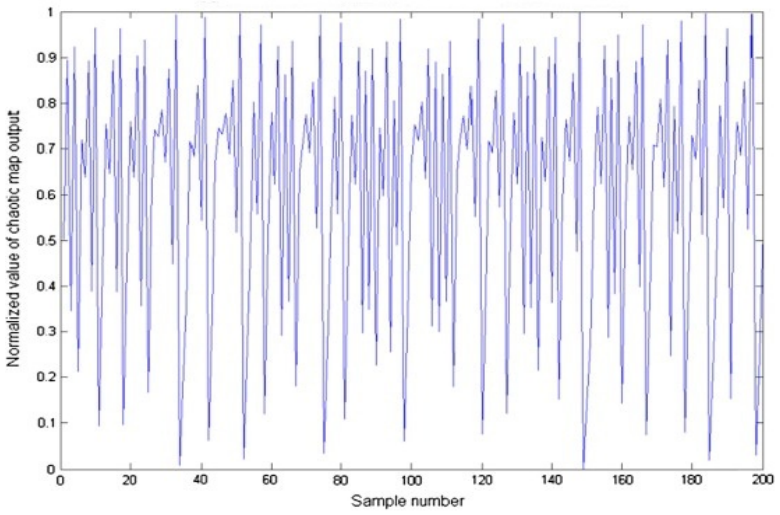


FIGURE 1.4: Dynamics of Verhulst chaotic model

model

$$cv_i^{k+1} = 3.56994 * cv_i^k(1 - cv_i^k) \tag{1.18}$$

where cv_i^k is the i^{th} chaotic velocity variable at k^{th} iteration and $cv_i^k \in (0, 1)$ under a constraint that $cv_i^0 \in (0, 1)$. The procedure of CVA can be elucidated as follows:

Step 1: For k^{th} iteration, form a one to one mapping between velocity variable $v_i^k \in (V_{\min}^k, V_{\max}^k)$ and chaotic velocity variable $cv_i^k \in (0, 1)$, where mapping function $\Lambda(V^k)$ is given as:

$$\Lambda(V^k) = cv_i^k = \left(\frac{v_i^k - V_{\min}^k}{V_{\max}^k - V_{\min}^k}\right), i = 1 \dots N \tag{1.19}$$

such that

$cv_i^k \in cV^k = \{cv_i^k \,|cv_i^k \in (0, 1)\, and 1 \leq i \leq N\}$

and one to one map can be represented as $\Lambda : V^k \mapsto cV^k$.

Step 2: Evaluate cv_i^{k+1} using Verhuslt or logistic function mentioned in (1.18).

Step 3: Evaluate new velocities for same iteration that is ${}^{new}v_i^k$ for k^{th} iteration again using expression (1.20) using cv_i^{k+1} in step 2.

$${}^{new}v_i^k = V_{\min}^k + cv_i^{k+1}(V_{\max}^k - V_{\min}^k) = \Pi(V^k), i = 1 \dots N \tag{1.20}$$

such that ${}^{new}V^k = \{{}^{new}v_1^k, \dots {}^{new}v_i^k, \dots {}^{new}v_N^k\} \in [V_{\min}, V_{\max}] \forall i = 1 \dots N$

One to one mapping between cV^k and ${}^{new}V^k$ can be expressed mathematically as $\Pi : cV^{k+1} \mapsto {}^{new}V^k$. From all the steps, it can be inferred that there is an indirect one to one mapping between V^k and ${}^{new}V^k$.

The procedure of CMBBBC is elaborated which uses CVA:

Pseudocode for CMBBBC

Step 1: Initialise x_{max}, x_{min}, N and $k = 0$.

Step 2: Generate population of size N,such that population set

$\vec{X}^k = \{\vec{x}_1^k, \ldots \vec{x}_i^k, \ldots \vec{x}_N^k\} \in [x_{\min}, x_{\max}] \forall i = 1 \ldots N$

Step 3: Each candidate in population is mapped to randomly generated velocities.

$\vec{V}^k = \{\vec{v}_1^k, \ldots \vec{v}_i^k, \ldots \vec{v}_N^k\} \in [\vec{V}_{\min}^k, \vec{V}_{\max}^k] \forall i = 1 \ldots N$

Step 4: Implement CVA algorithm on the velocity of particles obtained in step 3.

Step 5: Evaluate objective function for each particle

$f_i^k = f_0(x_n),\ n = 1 \ldots N$

such that $f^k = \{f_1^k, \ldots f_i^k \ldots, f_N^k\}$,$\forall i = 1 \ldots N$, where f_0 is objective function.

Step 6: Calculate the center of mass $\vec{C}^k(\vec{x}^k)$ as defined in equation (1.5).

Step 7: Find and store optimum value (say minimum), $least f_i = \min(f^k)$

check $if(\text{leastf}_\text{i} \leq \varepsilon)$, then jump to step 12.

Otherwise go to step 8.

Step 8: Calculate the velocity of center of mass $\vec{V}_C^k$ using expression (1.6), wherein $\vec{v}_i^k$ is replaced by ${}^{new}\vec{v}_i^k$.

Step 9: $k \leftarrow k + 1$.

Step 10: Generate new solutions around center of mass $\vec{C}^{k-1}(\vec{x}^{k-1})$ using expression (1.7), such that $\vec{X}^k = \{\vec{x}_1^k, \ldots \vec{x}_i^k, \ldots \vec{x}_N^k\} \in [x_{\min}, x_{\max}] \forall i = 1 \ldots N$

and the velocity term $\vec{v}_i^{k=k+1}$ in equation (1.7) is calculated using equation (1.8).

Step 11: Go to step 4.

Step 12: At $k =$ *iter* return optimized $\vec{C}^{iter}(\vec{x}^{iter})$ corresponding to $\vec{X}^{iter} = \{\vec{x}_1^{iter} \ldots \vec{x}_i^{iter} \ldots \vec{x}_N^{iter}\}$.

1.6 Simulation validation and results

1.6.1 Simulation constraints

We compare CMBBBC and MBBBC with PSO and BBBC. In CMBBBC, MBBBC and BBBC, the population size is 100. In BBBC, $r \in [0, 1]$ and $\alpha = 0.2$. For PSO the population size N is 100, c_1 and c_2 are taken as 1.494 and inertia weight varies linearly from 0.9 at the beginning of search to 0.4 at the end.

To test the performance of proposed algorithms, three famous benchmark optimization

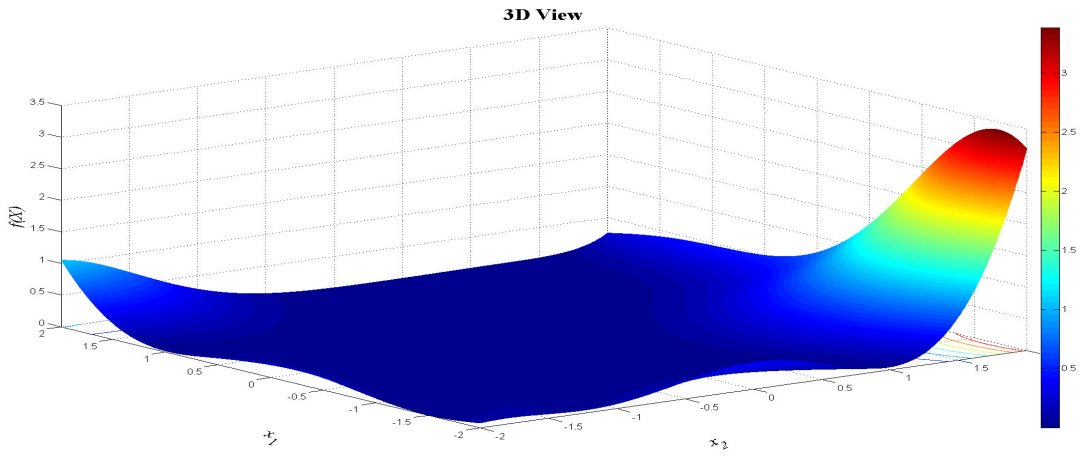
problems are used i.e.

1) ***Goldstein-Price (GP) function*** [4]:

Function: $f(x) = [1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)] * [30 + (2x_1 - 3x_2)^2(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)]$.

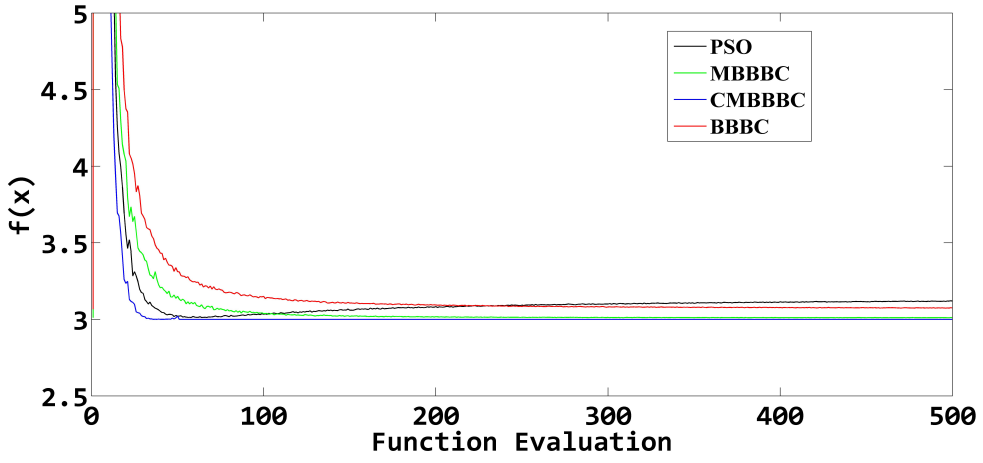
Input domain: Function is evaluated on $x_i \in [-2, 2], \forall i = 1, 2$.

Global minimum: $f(x^*) = 3, \mathrm{x}^* = (0, -1)$.



(a)

FIGURE 1.5: Goldstein-Price function



(a)

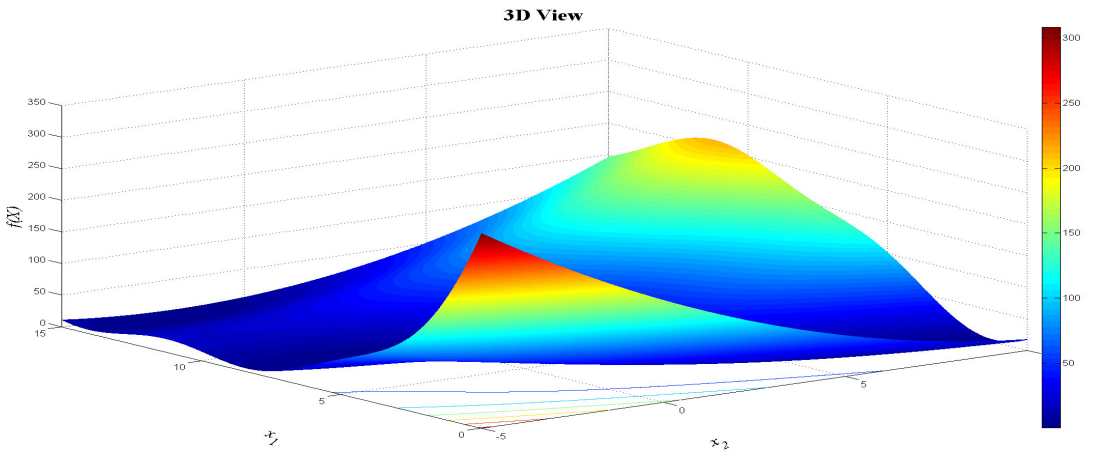
FIGURE 1.6: Comparision for Goldstein-Price function

2) ***Branin (BR) function***[4]:

Function: $f(x) = (x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6)^2 + 10(1 - \frac{1}{8\pi})\cos(x_1) + 10$.

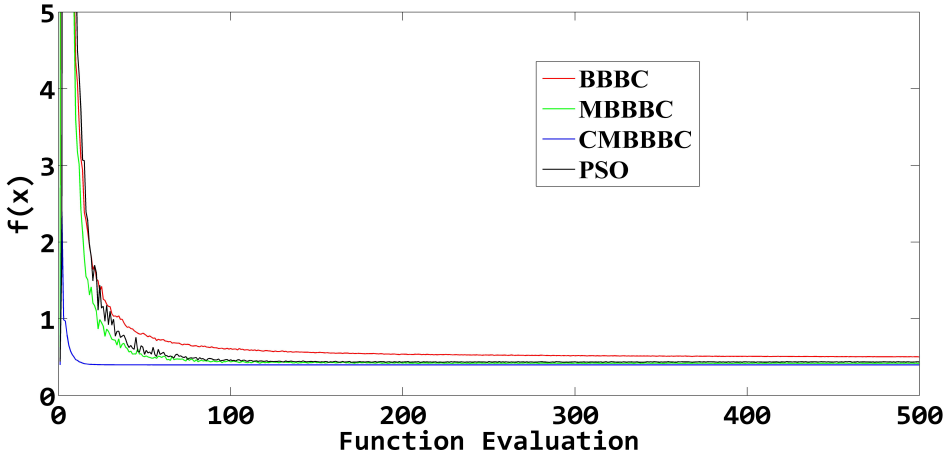
Input domain: Function is evaluated on $x_1 \in [-5, 10]$ and $x_2 \in [0, 15]$.

Global minimum: $f(x^*) = 0.397887, x^* = (-\pi, 12.275), (\pi, 2.275)$ and $(9.42478, 2.475)$.



(a)

FIGURE 1.7: Branin function



(a)

FIGURE 1.8: Comparision for Branin function

3) ***Egg-holder (EGG) function*** [4]

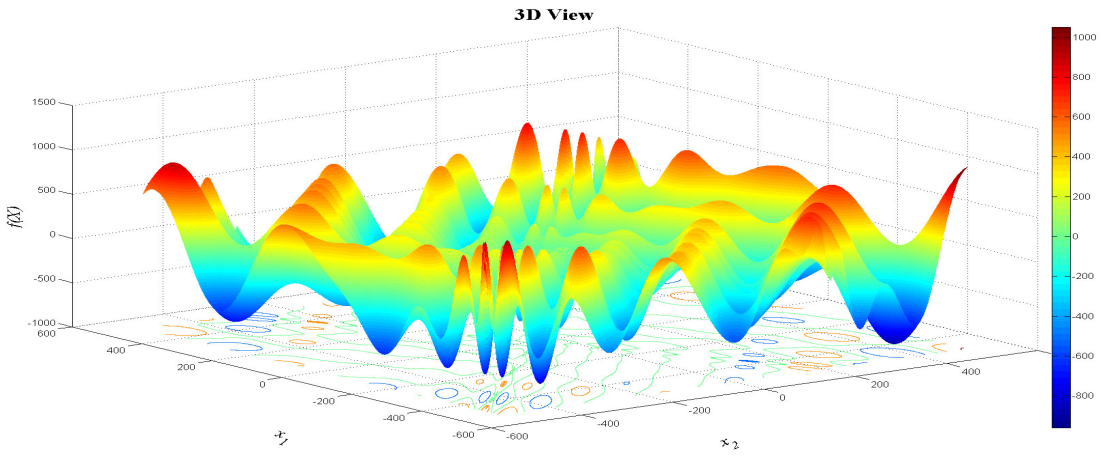
Function:$f(x) = -(x_2 + 47)\sin\left(\sqrt{\left|x_2 + \frac{x_1}{2} + 47\right|}\right) - x_1 \sin\left(\sqrt{|x_1 - (x_2 + 47)|}\right)$.

Input domain:Function is evaluated on $x_i \in [-512, 512], \forall i = 1, 2$.

Global minimum: $f(x^*) = -959.6407, x^* = (512, 404.2319)$.

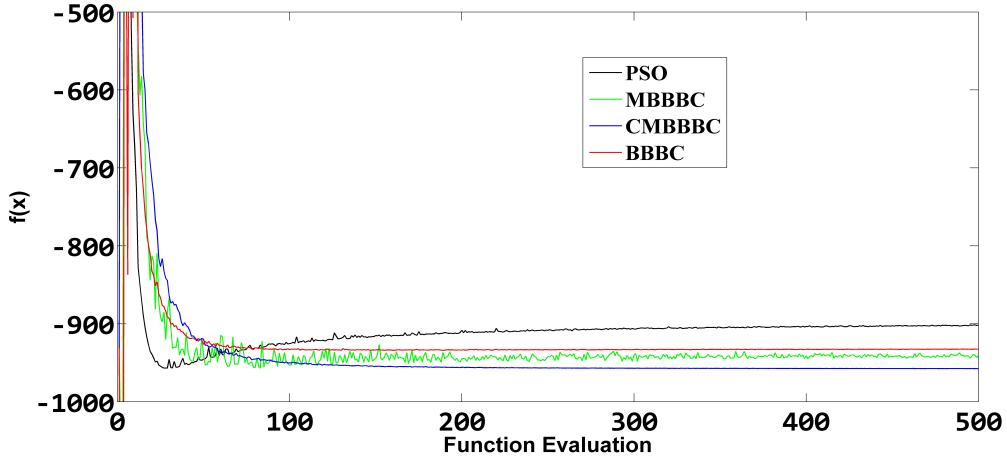
1.6.1.1 Fixed iteration results

We fix the total number of function evaluation as 500. Table 1.1 lists the average best function value and standard deviation of 50 independent trials. Figs. 1.3-1.5 show the performance of algorithms for solving three highly non-linear functions. From Table 1.1, it can be seen that the results of CMBBBC are almost equal to their original global minimum, and hence CMBBBC is superior to original BBBC, MBBBC, and PSO. From table



(a)

FIGURE 1.9: Egg holder function



(a)

FIGURE 1.10: Comparision for Egg holder function

TABLE 1.1: Fixed iteration results of 50 trial runs

f	$CMBBBC$	$BBBC$	$MBBBC$	PSO
f_{GP}	3.0000 ± 0.00002261	4.3624 ± 2.0310	3.2694 ± 0.2204	4.4345 ± 2.9654
f_{BR}	0.3988 ± 0.00009270	0.4555 ± 0.0504	0.4265 ± 0.02489	0.43344 ± 0.04309
f_{EGG}	-959.6127 ± 0.00934	-891.525 ± 33.9857	-904.976 ± 34.5468	-858.3028 ± 41.3207

1.1, it can be also be inferred that MBBBC is better than original BBBC and has efficiency comparable to PSO. So, it can be concluded that CMBBBC is the most efficient. Fig.1.3 depicts comparison for goldstein-price function and it can be inferred that MBBBC is far better than BBBC as rate of convergence is fast and rate of convergence is very high for CBBBC. In Fig.1.4 comparison is carried out for Branin function and again CMBBBC is a top performer with highest rate of convergence and MBBBC is better than original BBBC. Whereas in Fig.1.5, CMBBBC performs well as compared to

MBBBC, PSO and BBBC.

1.7 Conclusion

This chapter investigated the shortcomings of original BBBC and then a modified BBBC has been presented which is better than original BBBC. It is observed that BBBC and MBBBC are getting trapped in local optima. Therefore, MBBBC is then combined with chaos using CVA algorithm to overcome this problem, thereby improving solution exploration phenomenally.

Chapter 2

A Comparision of BBBC, PSO and GA Using Inferential Statistics

2.1 Introduction

Particle swarm optimization (PSO) was formulated by Kennedy and Eberhart in 1995 with an aim to mimic graceful motion of swarms of bird as a part of a sociocognitive study investigating the notion of "collective intelligence" in biological population [5]. In PSO, randomly generated swarm moves in designed search space towards optimal solution based on the information sharing between particles about there local and global best positions and also about there search space.

The Genetic Algorithm (GA) was formulated by John Holland and his group at University of Michigan in 1975. The GA mimics the behavior of reproduction in biotic population, basically it is influenced by concepts of evolution and genetics. This algorithm exploits the principal of "survival of the fittest" [5], in its search for optimal solution and generate individuals that fit to their environment. Thus, over number of iteration desirable attribute will evolve and persist in genome composition of population, whereas weak characteristics will subside over number generations (iteration).

In this chapter, first the PSO, GA and BBBC that are used in comparative study, and is briefed. This comparative study is carried using Statistical testing procedure which is explained subsequently. Secondly three well known benchmark function are used

to compare the efficiency and effectiveness of PSO, GA and BBBC in this statistical analysis.

2.2 Nomenclature

$c_1 = self\ confidence\ factor$

$c_2 = swarm\ confidence\ factor$

$f = fitness\ function$

$g = constra\,\text{int}\ function$

$H_o = null\ hypothesis$

$H_a = alternative\ hypothesis$

$n = sample\ size$

$N_{feval} = number\ of\ function\ evaluation$

$p_k^g = position\ of\ particle\ with\ best\ global\ fitness\ at\ current\ move\ k$

$p^i = best\ postion\ of\ particle\ i\ in\ current\ and\ all\ previous\ moves$

$r = penalty\ multiplier$

$rand = uniformly\ distributed\ random\ \text{var}\,iable\ between\ 0\ and\ 1$

2.3 PSO versus GA

2.3.1 The Particle Swarm Optimization

The PSO is one of the most widely used algorithm for solving complex nonlinear or linear engineering problems. The most elemental PSO algorithm can be subdivided into three steps. First step, being generation of particles in search space, initializing the velocity for the corresponding particles. second step being updating the velocity of particle depending upon the knowledge of position of local and global best position of the particles. Third step is position updation of candidates. These steps are explained briefly below.

In first step, new particles x_k^i and velocity v_k^i are generated randomly using upper and lower bound x_{max} and x_{min} as expressed in equation 2.1 and 2.2. Here, x_k^i represents i^{th} candidate in k^{th} iteration, and same can be defined for v_k^i i.e. velocity of particle of

i^{th} candidate in k^{th} iteration.

$$x_0^i = x_{\min} + rand(x_{\max} - x_{\min}) \tag{2.1}$$

$$v_0^i = \frac{x_{\min} + rand(x_{\max} - x_{\min})}{\Delta t} = \frac{position}{time} \tag{2.2}$$

The second step updates the velocity of particle and is given below in equation 2.3

$$v_{k+1}^i = wv_k^i + c_1 rand\frac{\left(p^i - x_k^i\right)}{\Delta t} + c_2 rand\frac{\left(p_k^g - x_k^i\right)}{\Delta t}. \tag{2.3}$$

New velocities of the candidate are designed based upon the knowledge of velocity of the previously defined candidates v_i^k, best position of the particle p^i over iteration defined using fitness function value and global best value of current swarm, p_k^g. This updated velocity v_{k+1}^i steers the candidates toward probable solution. This equation 2.3 has various random variables which has been defined in nomenclature section, some variables like *rand* ensures good coverage of design space and avoid entrapment in local optima. Three variables which has highest influence on steering velocities v_{k+1}^i are c_1, c_2 and w which are defined in nomenclature section. Here, c_1 weighs velocity of the best position of particle over course of iteration. whereas, c_2 weighs velocity of the best particle in the current swarm, and w weighs previous velocity of particle v_k^i.
Last step is updating the position of particle x_{k+1}^i, as given below in equation 2.4

$$x_{k+1}^i = x_k^i + v_{k+1}^i.\Delta t \tag{2.4}$$

and is depicted diagrammatically in Figure

These steps i.e. velocity and position update and fitness calculation are repeated until stopping criterion has been met.

2.3.2 The Genetic Algorithm

The genetic algorithm as said is inspired by theories of evolution and genetics. There are various versions of GA reported in literature. In this chapter, we have used a binary encoded GA with tournament selection, uniform crossover and low probability mutation rate is used to solve the benchmark problems.
In GA, defined design variable are encoded into binary 0's and 1's which are referred

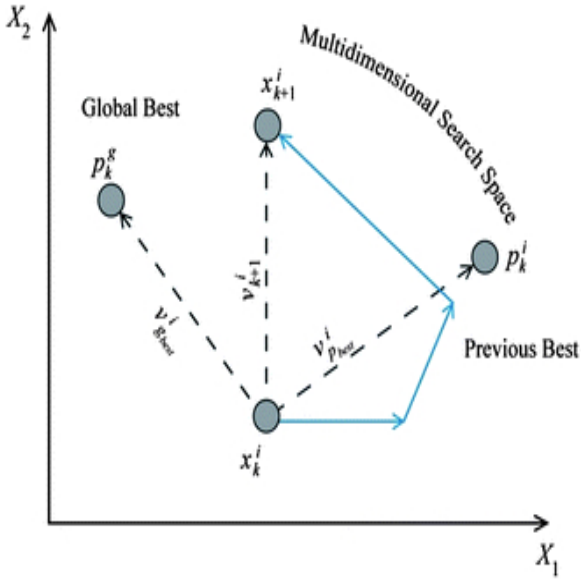


FIGURE 2.1: Depiction of the velocity and position update in PSO.

to as chromosomes. This feature of encoding ensures the design variable only take value within the defined range of search space. To perform optimization GA uses three operator i.e. Selection operator second one being crossover operator and last one is mutation operator. The first operator that is selection Operator is based on "survival of the fittest" as proposed by Darwin, wherein fittest candidates traits are passed onto generation and unfit candidates are eliminated subsequently. The "crossover opeartor" characterizes mating in biotic component in nature. And finally mutation operator helps explore diversity in population. Many research paper explains in detail the working of GA [6].

2.4 Comparision Metrics and Hypothesis Testing

The main objective of this work is to compare the efficiency and effectiveness of PSO, GA and BBBC using inferential statistics and set of benchmark function. There are various hypothesis testing method such as z-testing method, t-testing method, F-testing method and chi square test. The major difference between these test has been detailed below in Table 2.1, 2.2, 2.3, 2.4.

After studying the differences among these test it was decided to use t-testing method for comparison of the algorithms. So what is basically t-testing method?, How exactly is this test is carried out? and How are the conclusion drawn from this test?. The answer to these question will be given below in detailed manner with lucid explanation.

Table 2.1: z-test

z-test
1) A z-test is used for testing the mean of a population versus a standard, or comparing the means of two populations, with large ($n \geqslant 30$) samples whether you know the population standard deviation or not. **2)** It is also used for testing the proportion of some characteristic versus a standard proportion, or comparing the proportions of two populations. **3)** e.g.: Comparing the average engineering salaries of men versus women.

Table 2.2: t-test

t-test
1) A t-test is used for testing the mean of one population against a standard or comparing the means of two populations if you do not know the populations standard deviation and when you have a limited sample ($n \leqslant 30$). **2)** If you know the populations standard deviation, you may use a z-test. **3)** e.g.: Measuring the average diameter of shafts from a certain machine when you have a small sample.

Table 2.3: F-test

F-test
1) An F-test is used to compare 2 populations variances. **2)** The samples can be any size. **3)** e.g.: Comparing the variability of bolt diameters from two machines.

Table 2.4: chi square test

chi square test
1) Chi-square is a statistical test commonly used to compare observed data with data we would expect to obtain according to a specific hypothesis. **2)** The test is applied when you have two categorical variables from a single population. It is used to determine whether there is a significant association between the two variables. **3)** e.g.: Comparing the variability of bolt diameters from two machines.

TABLE 2.5: What is t-testing method?

What is t-testing method?
Is a statistical method used to test a hypothesis or a claim about a parameter in the population. Basically, we use a "sample"(known) from the "population"(unknown) to gain an insight on the characteristics of the unknown "population". **e.g.**: Suppose we come across an article claiming that children in India watch an average of 3 hours of TV per week. Here, hypothesis or claim is: Indian children watching TV for an average of three hours, and "population" (as in definition) is children in India.

In this research work, two hypothesis are being tested. The preliminary test is to check the effectiveness of the algorithms and second hypothesis to be tested is computational efficiency of the algorithms. Effectiveness here corresponds to characteristics of an algorithm. Which defines algorithms ability to converge sufficiently close to the actual solution when algorithms are run repeatedly. High effectiveness refers to higher probability of finding solution very close to actual solution or getting actual solution itself. Here, this effectiveness is quantified using "QUALITY OF SOLUTION (Q_{sol})", which is given below in equation 2.5,

$$Q_{sol} = |solution - known\ solution| . \tag{2.5}$$

It must be noted that this test is carried out for each algorithm separately. More precisely this test measure the effectiveness of algorithms with respect to know solution and then compare the algorithms with respect to each other.

Now lets discuss about the second test, i.e. computational efficiency test which is a very important work in this research work. This test compares the computational effort required by PSO, GA and BBBC to reach a solution. This is quantified or measured using number of function evaluation N_{eval}. The algorithms perform until reaching certain convergence criteria and lower the N_{eval} is , the more efficient the algorithm. This test and its constraints are given below.

2.5 Benchmark Test Problem

In this section, benchmark function of various characteristics have been presented. These benchmarks function are solved using PSO, GA and BBBC fifty times each, after this hypothesis test is carried out as mentioned in Table 2.6.

TABLE 2.6: Steps to t-testing method

Steps to t-testing method

Step 1: **Define the hypothesis.**

We start by defining alternate hypothesis (H_a) and null hypothesis (H_o).
H_o= This hypothesis is statement about the population parameter, such as mean of population which is given.
H_a= This hypothesis is a direct compliment of or contradicts the formed null hypothesis.
For example, $H_o : \mu = m$, then $H_a : \mu \neq m$.
This is a two sided hypothesis testing to test whether an unknown population mean μ is equal to population mean m.
Another example stating one sided hypothesis is, $H_o : \mu > m$ and hence alternate hypothesis is $H_a : \mu \leqslant m$.

Step 2: **Establish a decision criteria.**

Choose a deisred value of α, β *and* n to get $t_{critical}$ value from t-distribution tables that are available in standard statistics books.

α: Probability of rejecting a null hypothesis when it is actually true.
β: Probability of rejecting a null hypothesis when it is actually false.

Step 3: **Evaluate t-statistics.**

Calculate the t-value as described in Table 2.7 and Table 2.8.

Step 4: **Make a final conclusion about hypothesis.**

In this final step of testing method, the calculated t-value and $t_{critical}$ value are compared. If $t \leqslant t_{critical}$, H_o i.e. null hypothesis is accepted with $(1 - \alpha)$ confidence value.

Making a decision: Types of error

In this stage, we need to take a decision regarding rejecting or may be retaining null hypothesis and it is very clear that we are not observing the entire population i.e. information about population is unknown, but we only have knowledge regarding sample of population. Due to which the decision might be wrong. The Table 2.7 shows types of error in decision making.

TABLE 2.7: Four outcomes of making decision

		Decision	
		Retain the null	**Reject the null**
TRUTH IN THE POPULATION	**True**	**CORRECT**	**TYPE 1 ERROR (α)**
	False	**TYPE 2 ERROR (β)**	**CORRECT**

TABLE 2.8: Effectiveness test

Effectiveness Test
Objective to test whether $H_a : \mu_{Q_{sol}} \succ x$ (a value depending on the problem)
$H_o : \mu_{Q_{sol}} \leqslant x$
$t = \frac{\bar{Q}_{sol} - x}{s(Q_{sol})}$
taking $\alpha = 1\%,\ \beta = 1\%$ *and* $n = 10$
This is one sided test of significance of mean (table 6.10, [7])$\mapsto$ $t_{critical} = 2.0$

TABLE 2.9: Efficiency test

Efficiency Test
Objective to test whether H_{a1} : PSO $\mu_{N_{eval}} <$ GA $\mu_{N_{eval}}$
H_{a2} : PSO $\mu_{N_{eval}} <$ BBBC $\mu_{N_{eval}}$
H_{a3} : BBBC $\mu_{N_{eval}} <$ GA $\mu_{N_{eval}}$
H_{o1} : PSO $\mu_{N_{eval}} \geqslant$ GA $\mu_{N_{eval}}$
H_{o2} : PSO $\mu_{N_{eval}} \geqslant$ BBBC $\mu_{N_{eval}}$
H_{o3} : BBBC $\mu_{N_{eval}} \geqslant$ GA $\mu_{N_{eval}}$
$t_1 = \frac{GA\ \mu_{N_{eval}} - PSO\ \mu_{N_{eval}}}{\bar{s}(x)\sqrt{1/n_{GA}+1/n_{PSO}}},\ where\ \bar{s}(x) = \sqrt{\frac{(n_{GA}-1)s^2_{GA}+(n_{PSO}-1)s^2_{PSO}}{n_{GA}+n_{PSO}-2}}$
$t_2 = \frac{BBBC\mu_{N_{eval}} - PSO\ \mu_{N_{eval}}}{\bar{s}(x)\sqrt{1/n_{BBBC}+1/n_{PSO}}},\ where\ \bar{s}(x) = \sqrt{\frac{(n_{BBBC}-1)s^2_{BBBC}+(n_{PSO}-1)s^2_{PSO}}{n_{BBBC}+n_{PSO}-2}}$
$t_3 = \frac{GA\mu_{N_{eval}} - BBBC\ \mu_{N_{eval}}}{\bar{s}(x)\sqrt{1/n_{GA}+1/n_{BBBC}}},\ where\ \bar{s}(x) = \sqrt{\frac{(n_{GA}-1)s^2_{GA}+(n_{BBBC}-1)s^2_{BBBC}}{n_{GA}+n_{BBBC}-2}}$
taking $\alpha = 1\%,\ \beta = 1\%$ *and* $n_{PSO} = n_{GA} = 10$
This one sided test of significance of comparison of two mean (table 6.11, [7])$\mapsto$t$_{critical} = 2.5$

2.5.1 The Banana (Rosenbrock) Function

The banana function is one of the famous test problem for gradient based optimization algorithms. [8].

Properties

1) The function is unimodal.

2) Global minimum lies in narrow, parabolic valley [9].

3) Convergence to the minimum is difficult [10].

Function

$f(x) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2$

Input Domain

The function is evaluated on $x_i \in [-5, 5]$, $i = 1, 2$.

Global Minimum

$f(x^*) = 0,\ at\ x^* = [1, 1]$.

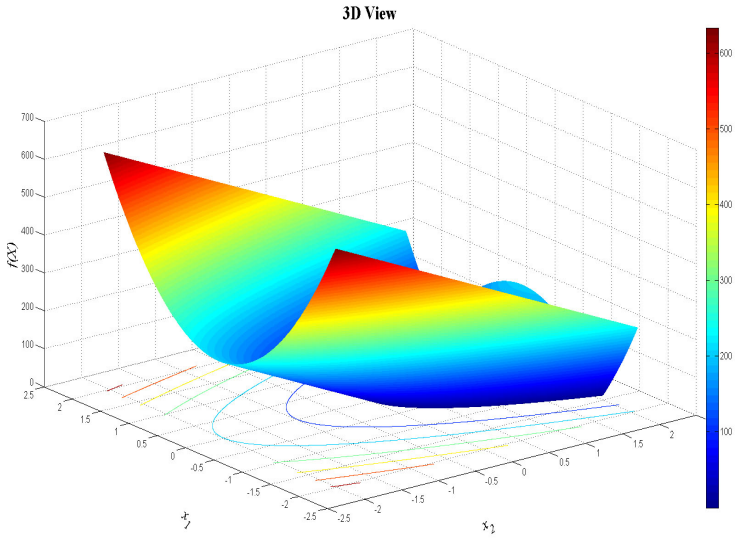


FIGURE 2.2: Rosenbrock function.

2.5.2 The Egg-Crate Function

In this function there are two design variables.

Properties

1) The function is multimodal.

Function

$f(x) = x_1^2 + x_2^2 + 25\left[\sin^2(x_1) + \sin^2(x_2)\right]$.

Input Domain

The function is evaluated on $-5 \leq x_i \leq 5$, $i = 1, 2$.

Global Minimum

$f(x^*) = 0,\ at\ x^* = [0, 0]$.

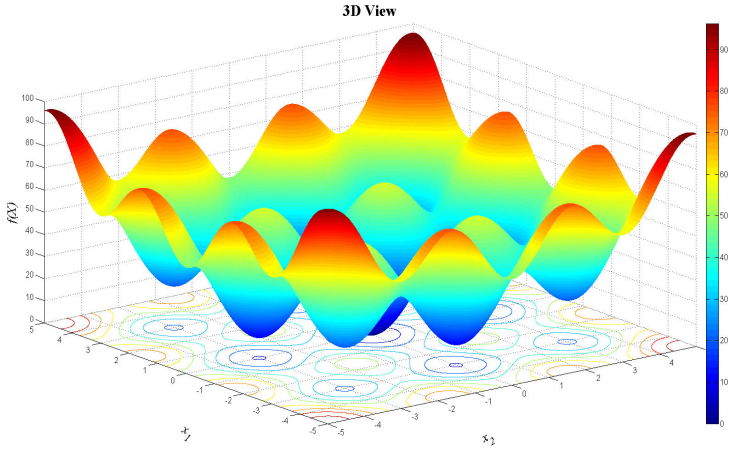


FIGURE 2.3: Egg-Crate Function.

2.5.3 The Rastrigin Function

This function is also one of the famous benchmark function which is used for testing optimization algorithms.

Properties

1) The Rastrigin function has several local minima.

2) It is highly multimodal function.

3) Locations of the minima are regularly distributed [11].

Function

$f(x) = \sum_{i=1}^{n} \left[x_i^2 - 10\cos(2\pi x_i) + 10\right],\ \forall\ i = 1, 2.$

Input Domain

The function is evaluated on $-5.12 \leq x_i \leq 5.12,\ \forall\ i = 1, 2.$

Global Minimum

$f(x^*) = 0,\ \ at\ x^* = [0, 0].$

2.5.4 The Goldstein-Price function

Explained in detail in section 1.6.1 (1).

2.5.5 The Branin function

Explained in detail in section 1.6.1 (2).

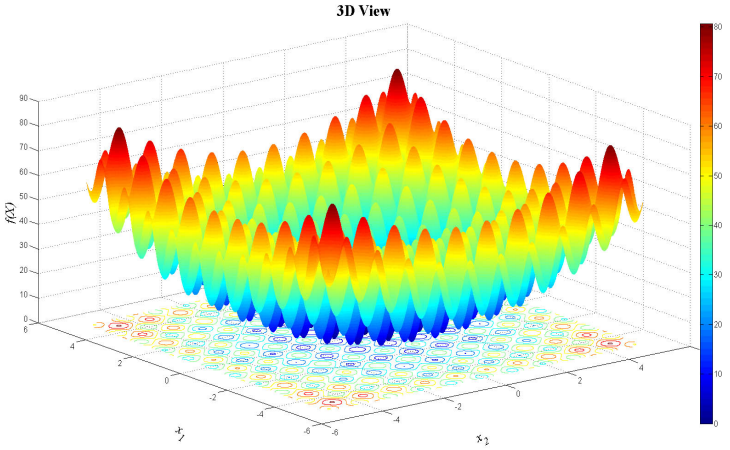


FIGURE 2.4: Rastrigin Function.

2.5.6 The Egg-holder function

Explained in detail in section 1.6.1 (3).

2.5.7 Golinski's Speed Reducer

The Golinski speed reducer is used in a airplane between the engine and propeller to allow each to rotate at its most efficient speed. Here the objective is to optimize the weight of speed reducer while gratifying all constraints imposed by gear and shaft design practices.

More precisely, our objective would be to find minimum weight (volume) of gear box, which is subjected to 11 constraints. Golinski speed reducer has seven design variables, which are defined below with appropriate figure. x_1= width of gear face, in cm.

x_2= teeth module, in cm.

x_3= number of pinion teeth.

x_4= shaft 1 length between bearings, in cm.

x_5= shaft 2 length between bearings, in cm.

x_6= diameter of shaft 1, in cm.

x_7= diameter of shaft 2, cm.

Objective Function

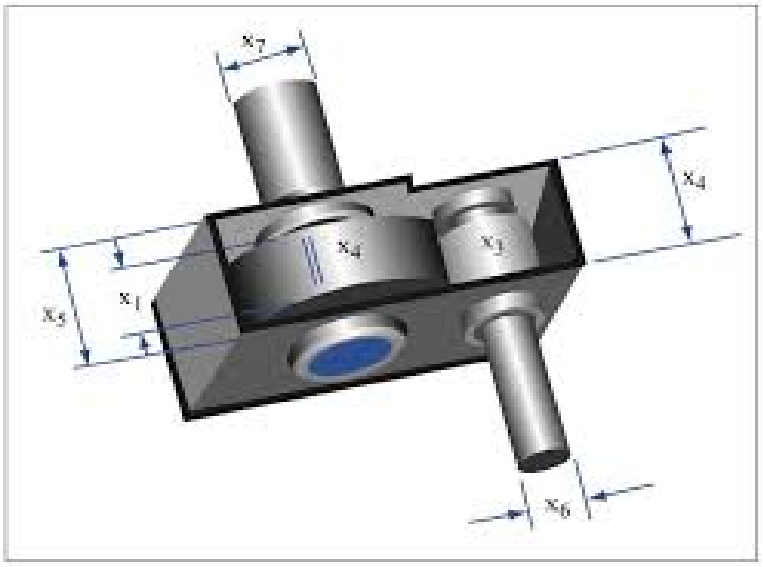


FIGURE 2.5: The Golinski Speed Reducer.

$$f(x) = C_{f1}x_1x_2^2(C_{f2}x_3^2 + C_{f3}x_3 - C_{f4}) - C_{f5}(x_6^2 + x_7^2)x_1 + C_{f6}(x_6^2 + x_7^3) + C_{f1}(x_4x_6^2 + x_4x_7^2)$$

$$C_{f1} = 0.7854 \qquad C_{f4} = 43.0934$$
$$C_{f2} = 3.3333 \qquad C_{f5} = 1.5079$$
$$C_{f3} = 14.9334 \qquad C_{f6} = 7.477$$

Constraints

$$2.6 \leqslant x_1 \leqslant 3.6 \qquad 7.3 \leqslant x_5 \leqslant 8.3$$
$$0.7 \leqslant x_2 \leqslant 0.8 \qquad 2.9 \leqslant x_6 \leqslant 3.9$$
$$17 \leqslant x_3 \leqslant 28 \qquad 5.0 \leqslant x_7 \leqslant 5.5$$
$$7.3 \leqslant x_4 \leqslant 8.3$$

Global Minimum

$f(x^*) = 2994.34\ kg$

$x^* = [3.5000\ 0.7000\ 17.0000\ 7.3000\ 7.7153\ 3.3502\ 5.2867]$

2.6 Results and Discussions

As discussed and elaborated in above sections, two test were carried i.e. effectiveness test and efficiency test using six benchmark function and a real world optimization problem i.e. Banana function, Egg-Crate function, Rastrigin function, goldstein-Price function, Branin function, Egg holder function, Golinski speed reducer problem. The first test i.e.

TABLE 2.10: Calculated $t - values$ for hypothesis test

Benchmark Function	**Null Hypothesis**	**Effectiveness Test, $t_{critical} = 2$ Calculated t-values**			**Efficiency Test, $t_{critical} = 2.5$ Calculated t-values**		
		PSO	**GA**	**BBBC**	t_1	t_2	t_3
Banana Function	$H_o : \mu_{Q_{sol}} \leqslant 1$	**1.6981**	**1.2413**	**1.9485**	**1.945**	**2.412**	**1.458**
Egg-Crate Function	$H_o : \mu_{Q_{sol}} \leqslant 1$	**1.4787**	**1.3875**	**1.1247**	**5.642**	**10.897**	**2.127**
Rastrigin Function	$H_o : \mu_{Q_{sol}} \leqslant 1$	**1.1260**	**1.1796**	**1.1787**	**10.265**	**23.457**	**1.647**
Goldstein-Price Function	$H_o : \mu_{Q_{sol}} \leqslant 16.67\%$	**0.0312**	**10.584**	**0.0003**	**4.312**	**9.457**	**0.948**
Branin Function	$H_o : \mu_{Q_{sol}} \leqslant 24.6\%$	**1.3458**	**29.456**	**1.2546**	**20.456**	**4.632**	**2.154**
Egg Holder Function	$H_o : \mu_{Q_{sol}} \leqslant 1\%$	**1.2985**	**5.6478**	**12.176**	**10.541**	**45.127**	**1.945**
Golinski's speed reducer	$H_o : \mu_{Q_{sol}} \leqslant 0.145\%$	**0.0048**	**1.2625**	**0.0012**	**19.458**	**17.456**	**1.564**

effectiveness test measures the Q_{sol} found using PSO, GA and BBBC with respect to known solution. This test inspects whether Q_{sol} is greater then 99%. The second test, measures the efficiency of the algorithms in-terms of N_{eval}. The lesser the N_{eval}, more the computational efficiency. The effectiveness test for PSO, GA and BBBC shows that for most of the cases $t < t_{critical}$ in most results in Table 2.10. This leads to acceptance of null hypothesis and rejection of alternate hypothesis. Which means that quality of solutions for all the three algorithm are near to there optimal solution. Whereas, for GA we encounter some cases where $t > t_{critical}$, which leads to the conclusion that null hypothesis is rejected and alternate hypothesis is accepted.

Now lets look into the results of efficiency test, again as the condition says if $t > t_{critical}$ then this leads to rejection of null hypothesis and the acceptance of alternate hypothesis with a confidence level of 99%. From the results of efficiency test in Table 2.10, it can be observed that for first and second hypothesis $t > t_{critical}$ except for Banana function. Which leads to conclusion that computational efficiency or effort required to reach solution for benchmark problem by PSO is better than GA and BBBC. One exceptional case that is banana function in first hypothesis, where $t < t_{critical}$ suggests that computational efficiency of GA is better than PSO. Now lets take a look on results obtained for third hypothesis i.e. $H_{a3} : BBBC\ \mu_{N_{eval}} < GA\ \mu_{N_{eval}}$, it can be seen that for all $t < t_{critical}$. This leads to conclusion that null hypothesis is accepted with

confidence level of 99%, and hence prooves that computational effort required by BBBC is to obtain the solution is lesser then GA.

2.7 Conclusion

In this research work, we have conducted two test first test is effectiveness test and other test is efficiency test. The results obtained during these test are very convincing, it is found that effectiveness of algorithm which essentially indicates algorithms ability to reach to an optimal solution or near to it, is same for all the three algorithms i.e. PSO, GA and BBBC. And hence can be easily concluded with reference to result in Table 2.10 that all the three algorithms are equally effective. From second hypothesis test regarding computational efficiency of algorithms, it was seen that PSO is better than GA and BBBC, and it is also inferred that BBBC has better computational efficiency than GA.

Chapter 3

A Novel Model Order Reduction Technique Using BBBC and Time Moment Matching Method

3.1 Introduction

The concept of model order reduction (MOR) is basically practiced in the field of systems and control engineering which analyses the properties of dynamical systems to reduce their complexity and retain their input-output behaviour as much as possible. MOR simplifies the understanding of the system, and minimizes the computational burden in the simulation studies. Moreover unlike the design of complex H_∞ and μ synthesis based control schemes, it enables the control practitioners to design simple control laws thereby making controller computationally and cost efficient [12]. Advanced robust control concepts such as H control and synthesis are widely used in various engineering applications wherein the design schemes produce highorder controllers even for a simple second-order plant [13]. To cope with the aforementioned challenges faced when dealing with large-scale dynamical systems, many studies of different MOR approaches have been proposed using variety of concepts. Over thirty years of research in MOR of linear time-invariant (LTI) system, the developed methods conceptualizes dominant pole retention, singular value decomposition and Hankel norm based approximation, Krylov subspace method, H_∞-optimization methods [14],[15],[16]. One of the important trends in these studies is the optimization (minimization) of integral error criterion between the actual plant and its reduced model. In recent years, there is a widespread interest and

research in evolutionary optimization techniques mainly because of its intuitiveness, ease of implementation, and the ability to effectively solve highly nonlinear, mixed integer optimization problems of complex engineering systems. Now-a-days, these evolutionary optimization techniques unified with conventional MOR techniques have shown highly promising results, even results better than the conventional techniques. Among various evolutionary techniques, particle swarm optimization (PSO) and genetic algorithm (GA) are considered to be highly reliable algorithm for solving optimization problems [17]. On the same lines, a novel universe inspired evolutionary technique so called Big Bang Big Crunch (BBBC) can be used for order reduction of systems of various complexities [1]. Recently, a mixed method is proposed for order reduction of LTI systems for both single-input single-output (SISO) and multi-input multi-output (MIMO) systems [18]. This method is based on unification of BBBC and Routh approximation method. It is observed that this method is better than the existing method and has been applied for low-order systems but a more accurate and improved results can also be achieved. Therefore, in this paper, an alternative mixed method is proposed to obtain reduced model using BBBC and time moment matching [19] method. In this approach, numerator coefficients of reduced-order transfer function model are optimized using BBBC technique and denominator is evaluated from time moment matching method [19] using full-order plant's information. This proposed method is applied to SISO, MIMO and time delayed system. The comparison of the existing approaches and proposed approach has been carried which show the remarkable improvement in integral square error (ISE) and other performance parameters.

3.2 Model order reduction from control system perspective

In control theory, MOR can be defined as follows.

Definition 1: Let $G(s) : u \rightarrow y$ be the transfer function of the full-order system with order$n$, then the model reduction scheme seeks a reduced-order transfer function model $\tilde{G}(s) : u \rightarrow \tilde{y}$ with $\tilde{n}$ so that $\tilde{n} < n$ and for the same input$u(t)$, we have$\tilde{y}(t) \approx y(t)$.

In other words, MOR leads to optimization problem in the following manner.

Definition 2: MOR defines the problem of finding the reduced-model $\tilde{G}(s)$ from the full-order plant $G(s)$using optimization formulation such that

$$\text{minimize } ISE = \int [y(t) - \tilde{y}(t)]^2 dt$$

$$\text{subject to } ISE < \varepsilon$$

where ε is an error tolerance.

3.3 Basics of Time Moments Matching method

Let the impulse response of high-order asymptotically stable system be $g(t)$ then

$$G(s) = \int_0^\infty g(t)e^{-st}dt$$

Now using the power series expansion for e^{-st}, G(s) can be written as

$$G(s) = \int_0^\infty g(t)\{1 - st + \frac{s^2t^2}{2!} - \frac{s^3t^3}{3!} + ...\}dt = \int_0^\infty g(t)dt - s\int_0^\infty tg(t)dt + s^2\int_0^\infty \frac{t^2}{2!}g(t)dt - ...$$

or

$$G(s) = c_0 + c_1 s + c_2 s^2 + ...$$

where,

$$c_i = \frac{(-1)^i}{i!}\int_0^\infty t^i g(t)dt, \ \forall i = 0, 1, 2, ...$$

$$= \left[\frac{(-1)^i}{i!}\right] [the\ i^{th}\ timemoment\ of\ g(t)]$$

Thus, the time moment of the reduced system $\tilde{G}(s)$ are proportional to power series expansion of $G(s)$, about $s = 0$. In time moment method [19], the idea is to match as many time moments [19] of original and reduced system as much possible.

3.4 Main Results

3.4.1 Statement of Problem

In this subsection, we define the mathematical expressions for asymptotically stable, LTI-SISO and MIMO plants.

SISO system: The original n^{th} order system can be written in, usual notation, in transfer function form as

$$G(s) = \frac{\sum_{k=0}^{m} a_k s^k}{\sum_{k=0}^{n} b_k s^k} = \frac{a_0 + a_1 s + a_2 s^2 + a_3 s^3 + \cdots + a_m s^m}{b_0 + b_1 s + b_2 s^2 + b_3 s^3 + \cdots + b_n s^n},\ m \leq n \text{ and } k,\ m\ , n \in\ I \tag{3.1}$$

where $G(s)$ is BIBO stable continuous-time original system of order n, a_k and b_k are constant coefficients of s (complex variable) in numerator and denominator polynomials respectively. Our objective is to reduce the original full-order system $G(s)$ into its approximated reduced-order model $\tilde{G}(s)$ of form

$$\tilde{G}(s) = \frac{\sum_{k=0}^{m_r} \tilde{a}_k s^k}{\sum_{k=0}^{n_r} \tilde{b}_k s^k},\ m_r \leqslant n_r \ and\ k, m_r, n_r \in I \tag{3.2}$$

where $\tilde{a}_k$ and $\tilde{b}_k$ are coefficients of s in numerator and denominator, respectively.

MIMO system: The original n^{th}order MIMO system can be written in transfer function matrix form as

$$[G(s)] = \frac{1}{\sum_{k=0}^{n} b_k s^k} \begin{pmatrix} a_{11} & \cdots & a_{1m} \\ \vdots & \ddots & \vdots \\ a_{p1} & \cdots & a_{pm} \end{pmatrix} \tag{3.3}$$

where, $[G(s)]$ is original system and dimension of $[G(s)]$ is $p \times m$. Equation (7) can also be represented as

$$[G(s)] = [g_{ij}]_{p \times m} = \begin{pmatrix} g_{11} & \cdots & g_{1m} \\ \vdots & \ddots & \vdots \\ g_{p1} & \cdots & g_{pm} \end{pmatrix},\ i, j \neq 0 \tag{3.4}$$

where,

$$g_{ij} = \frac{a_{ij}}{\sum_{k=0}^{n} b_k s^k} = \frac{A_0 + A_1 s + A_2 s^2 + \cdots + A_m s^m}{b_0 + b_1 s + b_2 s^2 + \cdots + b_n s^n}, m \leqslant n \tag{3.5}$$

The model reduction of (7) gives

$$\left[\tilde{G}(s)\right] = \frac{1}{\sum_{k=0}^{n_r} \tilde{b}_k s^k} \begin{pmatrix} \tilde{a}_{11} & \dots & \tilde{a}_{1m} \\ \vdots & \ddots & \vdots \\ \tilde{a}_{p1} & \cdots & \tilde{a}_{pm} \end{pmatrix} \tag{3.6}$$

Where, $[\tilde{G}(s)]$ is reduced system corresponding to original system $[G(s)]$ and dimension of $[\tilde{G}(s)]$ is $p \times m$. In other words, (3.6) can be written as

$$[\tilde{G}(s)] = [\tilde{g}_{ij}]_{p\times m} = \begin{pmatrix} \tilde{g}_{11} & \cdots & \tilde{g}_{m} \\ \vdots & \ddots & \vdots \\ \tilde{g}_{p1} & \cdots & \tilde{g}_{pm} \end{pmatrix} \tag{3.7}$$

where,

$$\tilde{g}_{ij} = \frac{\tilde{a}_{ij}}{\sum_{k=0}^{z} \tilde{b}_k s^k} = \frac{\tilde{A}_0 + \tilde{A}_1 s + \tilde{A}_2 s^2 + \cdots + \tilde{A}_w s^w}{\tilde{b}_0 + \tilde{b}_1 s + \tilde{b}_2 s^2 + \cdots + \tilde{b}_z s^z},\ w \leqslant z \tag{3.8}$$

where $w < m,\ z < n$

3.5 Proposed algorithm

In this subsection, we discuss the procedure of our proposed MOR algorithm. Let us consider a full-order plant described in (3.1). The required form of reduced-order model having order $p,\ p < n$ can be written as

$$\tilde{G}(s) = \frac{x_0 + x_1 s + ... + {x_{p-1}}^{p-1}}{1 + a_{13}s + a_{12}s^2 + \cdots + a_{1p}s^p} \tag{3.9}$$

such that $p < n$. The proposed algorithm constitutes two parts:

3.5.0.1 Determination of denominator polynomial using Time Moment Matching method

We follow the time moment matching [19] to obtain the denominator polynomial of reduced model.

Step 1: In (3.1), divide the numerator and denominator coefficient of $G(s)$ with b_0 to

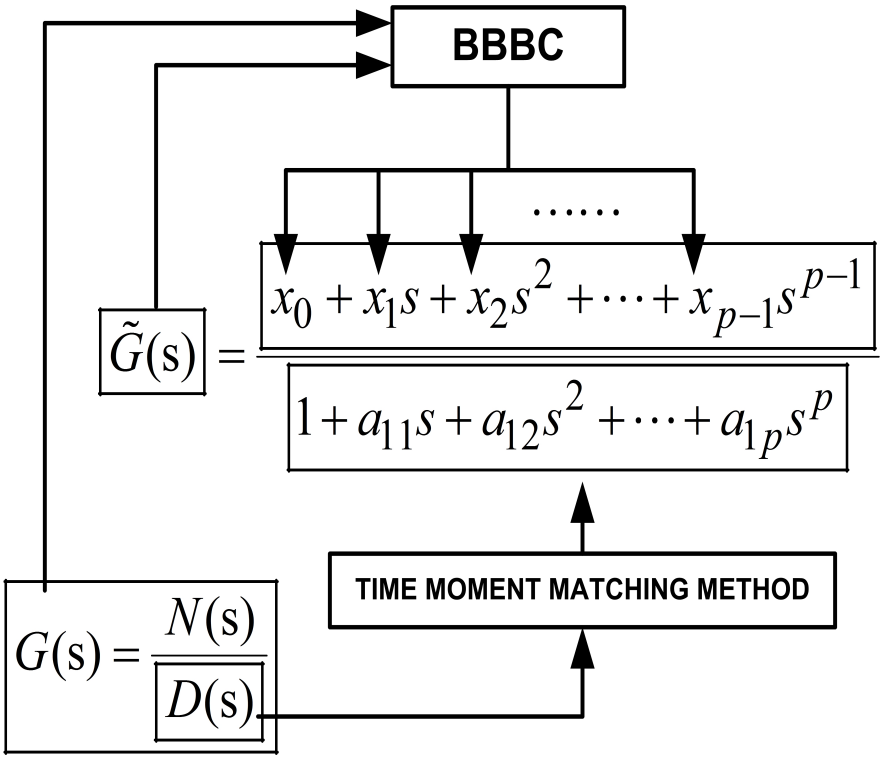


FIGURE 3.1: Schematic representation of working of proposed algorithm.

get

$$G(s) = \frac{\sum_{k=0}^{m} a_k s^k}{\sum_{k=0}^{n} b_k s^k} = \frac{a_0/b_o + a_1/b_o s + a_2/b_o s^2 + a_3/b_o s^3 + \cdots + a_m/b_o s^m}{1 + b_1/b_o s + b_2/b_o s^2 + b_3/b_o s^3 + \cdots + b_n/b_o s^n},\ m \leqslant n \tag{3.10}$$

which can be further written as

$$G(s) = \frac{a_{21} + a_{22} s + a_{23} s^2 + a_{24} s^3 + \cdots + a_{2m} s^m}{1 + a_{12} s + a_{13} s^2 + a_{14} s^3 + \cdots + a_{1n} s^n} \tag{3.11}$$

Step 2: Now, arranging the coefficient of rational polynomial function of (3.11) in Routhian array form as:

$$\begin{array}{|cccccc|} \hline 1 & a_{12} & a_{13} & a_{14} & \cdots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \cdots & a_{2m} & 0 \\ a_{31} & a_{32} & a_{33} & \cdots & a_{3m} & 0 \\ a_{41} & a_{42} & a_{43} & \cdots & a_{4m} & 0 \\ a_{51} & a_{52} & a_{53} & \cdots & a_{5m} & 0 \\ \vdots & \cdots & \cdots & \cdots & \vdots & 0 \\ \hline \end{array} \tag{3.12}$$

where, $a_{k,l} = a_{k-1,1} \times a_{1,l+1} - a_{k-1,l+1} \times a_{1,1}\ \forall k = 3, 4, ...and\ l = 1, 2, ...$ Next, the

values obtained from Routhian array can be used as,

$$G(s) = a_{21} - \ a_{31}s + a_{41}s^2 - a_{51}s^3 \ + \ \cdot \ \cdot \ \cdot$$

Step 3: Using $c_i = (-1)^{k+2} \times a_{k1} \forall k = 2, 3, ...$ and $i = 0, 1, 2, ...$ The matrix can be structured into the form as shown in Fig.3.1

$$\begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ \vdots \\ c_m \\ \hline c_{m+1} \\ c_{m+2} \\ \vdots \\ c_{m+n} \end{bmatrix} = \left[\begin{array}{cccccc|ccc} 0 & 0 & & \cdots & & 0 & 0 & \cdots & 0 \\ -c_0 & 0 & & & & & & & \\ -c_1 & -c_0 & & \ddots & & 0 & & \ddots & \vdots \\ \vdots & \vdots & & & & \vdots & \vdots & & \\ -c_{m-1} & -c_{m-2} & & \cdots & -c_0 & 0 & 0 & \cdots & 0 \\ \hline -c_m & -c_{m-1} & \cdots & -c_1 & -c_0 & & \cdots & \cdots & 0 \\ -c_{m+1} & -c_m & \cdots & & -c_1 & \cdots & & & 0 \\ \vdots & & & & & & \vdots & & \vdots \\ -c_{m+n+1} & -c_{m+n-2} & \cdots & & & 0 & \cdots & & 0 \end{array}\right] \times \begin{bmatrix} \hat{a}_{12} \\ \hat{a}_{13} \\ \hat{a}_{14} \\ \vdots \\ a_{1,n+1} \\ \hline 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} + \begin{bmatrix} \hat{a}_{21} \\ \hat{a}_{22} \\ \hat{a}_{23} \\ \vdots \\ \hat{a}_{2,m+1} \\ \hline 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

FIGURE 3.2: Moment matrix

This matrix can be written as shown in (3.12) which is called as partitioned matrix can be written as

$$\left[\frac{\hat{C}_1}{\hat{C}_2}\right] = \left[\begin{array}{c|c} C_{11} & 0 \\ \hline C_{21} & C_{22} \end{array}\right] \times \left[\frac{\hat{A}_1}{0}\right] + \left[\frac{\hat{A}_2}{0}\right] \tag{3.13}$$

where, C_{11}, C_{21}, C_{22} are of order $(m+1) \times n, \ n \times n, \ n \times (m+1)$, respectively.

Step 4: Lastly calculate $\hat{A}_1 = C^{-1}{}_{21} \times \hat{C}_2$. For calculating the denominator coefficient defined by $\hat{A}_1 = \begin{bmatrix} \hat{a}_{12} & \hat{a}_{13} & .... & \hat{a}_{1n} \end{bmatrix}^T$, then say the second-order denominator can be written as $\tilde{D}(s) = \hat{a}_{12}s^2 + \hat{a}_{13}s + 1$.

3.5.0.2 Determination of Numerator polynomial using BBBC

Once the denominator polynomial of form: $\tilde{D}(s) = \hat{a}_{1p}s^p + \hat{a}_{1p-1}s^{p-1} + ... + \hat{a}_{12}s^2 + \hat{a}_1 s + 1$ where $p < n$ is calculated. Our objective is to calculate the coefficients $[x_0, x_1, \ldots, x_{p-1}]$ of $\tilde{N}(s)$ as can be inferred from (3.9) using BBBC such that the ISE get minimized. The steps to obtain the numerator coefficients can be further illustrated in the following algorithm.

Algorithm

Let $G(s)$ be given.

Step 1: Initialise $r, \alpha, N, \Delta t$ and $iter(iteration) = 1$;

Step 2: Generate population (containing N particles)

$$X_k^{iter} = \{x_1^{iter}, x_2^{iter}, \ldots, x_N^{iter}\} \in [x_{\max}, x_{\min}], \forall k = 1 \ldots N$$

Step 3: Evaluate objective function for each particle

$$ISE_k^{iter}(x_k^i ter) = \sum [y(i\Delta t) - \tilde{y}(i\Delta t)]^2, \forall k = 1 \ldots N$$

Step 4: Find and store

$leastISE = \min(ISE_k^{iter})$

$if(leastISE \leqslant \varepsilon)$ Then jump to step 9

Otherwise Go to step 5;

Step 5: Calculate the center of mass as defined in Equation (1.1)

Step 6: $iter \leftarrow iter + 1$

Step 7: Generate of new solution around center of mass

$$X_k^{iter} = \{x_1^{iter}, x_2^{iter}, \ldots, x_N^{iter}\} = \vec{C}(x_k^{iter-1}) + \delta_i \tag{3.14}$$

Step 8: Go to step 3

Step 9: At $k =$ $iter$, return optimized $\vec{C}^{iter}(\vec{x}^{iter})$ corresponding to $\vec{X}^{iter} = \{\vec{x}_1^{iter} \ldots \vec{x}_i^{iter} \ldots \vec{x}_N^{iter}\}$

3.6 Numerical Studies

In this section, the proposed MOR method is applied to six different problems and the results are compared with other recently developed schemes.

Example 1: Consider a fourth-order system [18] expressed in transfer function form as:

$$G_1(s) = \frac{s^3 + 7s^2 + 24s + 24}{s^4 + 10s^3 + 35s^2 + 50s + 24} \tag{3.15}$$

On dividing the numerator and denominator by 24 in (3.15), we have

$$G_1(s) = \frac{1 + s + 0.292s^2 + 0.0416s^3}{1 + 2.083s + 1.458s^2 + 0.4167s^3 + 0.0416s^4} \tag{3.16}$$

Now arranging the above transfer function (3.16) in Routh array form, we get

1	2.083	1.458	0.4167	0.0416
1	1	0.292	0.0416	0
1.083	1.166	0.3751	0.0416	0
1.0898	1.2039	0.4096	0.045	0
1.0661	1.1793	0.4091	0.0453	0
1.0414	1.1453	0.3989	0.0443	0
1.0239	1.1195	0.3896	0.0433	0
1.0133	1.1032	0.3833	0.0426	0

Forming partitioned matrix using value of $c_i \forall i = 0, 1, ...$ where $c_0 = 1,\ c_1 = -1.083,\ c_2 = 1.0899,\ c_3 = -1.0663$ as in (3.17).

$$\left[\begin{array}{c} 1 \\ -1.083 \\ \hline 1.089 \\ -1.066 \end{array}\right] = \left[\begin{array}{cc|cc} 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ \hline 1.083 & -1 & 0 & 0 \\ -1.089 & 1.083 & 0 & 0 \end{array}\right] \times \left[\begin{array}{c} \hat{a}_{12} \\ \hat{a}_{13} \\ \hline 0 \\ 0 \end{array}\right] + \left[\begin{array}{c} \hat{a}_{21} \\ \hat{a}_{22} \\ \hline 0 \\ 0 \end{array}\right] \tag{3.17}$$

From partitioned moment matrix in (3.17), we get, $C_{21} = \begin{bmatrix} 1.083 & -1 \\ -1.0899 & 1.083 \end{bmatrix}$ and therefore $\hat{A}_1 = (C_{21})^{-1} \times \hat{C}_2 = \begin{bmatrix} 1.3744 \\ 0.3986 \end{bmatrix}$. After this step, the BBBC algorithm is employed to find the coefficients of numerator, such that objective functions (ISE) get minimised and the results obtained are $x_1 = 0.28389$ and $x_2 = 1.00043$. Hence, the reduced- order model of a original system obtained is

$$\tilde{G}_1(s) = \frac{1.00043 + 0.28389s}{1 + 1.3744s + 0.3986s^2} \tag{3.18}$$

The comparison of original model with proposed reduced models and other recently developed reduced order models in time domain and frequency domain is performed, which is shown in Fig.3.2, 3.3 and Fig.3.4, 3.5. respectively. It is observed that in both domains, proposed model is a better approximation with respect to original model in comparison with other existing models. Further in order to validate this, ISE are calculated for all the models. They are shown in Table 3.2. It is observed that the ISE of the proposed model is the least in comparison to other existing models. Furthermore,

the qualitative comparison is also carried out which is shown in Table 3.1. For qualitative comparison, rise time, settling time, and peak overshoot are considered. Again, it is found that the proposed method has better performance parameters.

TABLE 3.1: Qualitative comparison with respect to transient response

Model Order Reduction Technique	Rise Time (T_r)	Settling Time (T_s)	Peak Overshoot(M_p)
Original System	2.260	3.931	0
Proposed Method	2.268	3.958	0
Sikander & Prasad [20]	2.301	3.410	1.0722
Desai and Prasad [18]	2.278	3.619	0.274
Truncation Method [21]	2.737	4.080	0.564
Routh Hurwitz [22]	1.926	5.587	3.595
Pade Approximation [23]	2.260	3.947	0
Singular Perturbation [24]	2.259	3.917	0
Balanced Realization [25]	2.089	5.355	2.059
Parmar Method [26]	2.188	3.219	1.301
Chen et al [27]	2.301	3.410	1.072
Pal [28]	15.381	27.361	0
Parthasarathy and Jayasimha [29]	1.591	5.563	4.099
Davision [30]	1.696	3.257	0
Sheih and Wei [31]	4.961	8.834	0
Safonov and Chiang [32]	3.801	6.747	0

Example 2 Let us now take an eighth order system [**?**]:

$$G_2(s) = \frac{18s^7 + 514s^6 + 5982s^5 + 36380s^4 + 122664s^3 + 222088s^2 + 185760s + 40320}{s^8 + 36s^7 + 546s^6 + 4536s^5 + 22449s^4 + 67284s^3 + 118124s^2 + 109584s + 40320} \tag{3.19}$$

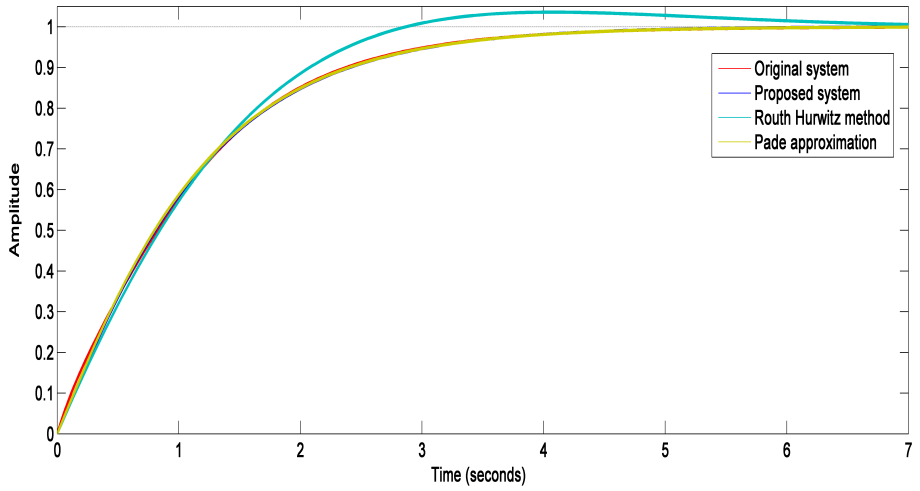
On dividing the coefficients of both numerator and denominator by 40320 in (3.19), we get

$$G_2(s) = \frac{1 + 4.607s + 5.508s^2 + 3.042s^3 + 0.9023s^4 + 0.1484s^5 + 0.01276s^6 + 0.0004464s^7}{1 + 2.718s + 2.93s^2 + 1.669s^3 + 0.5568s^4 + 0.1125s^5 + 0.01354s^6 + 0.0008929s^7 + 0.0000248s^8} \tag{3.20}$$

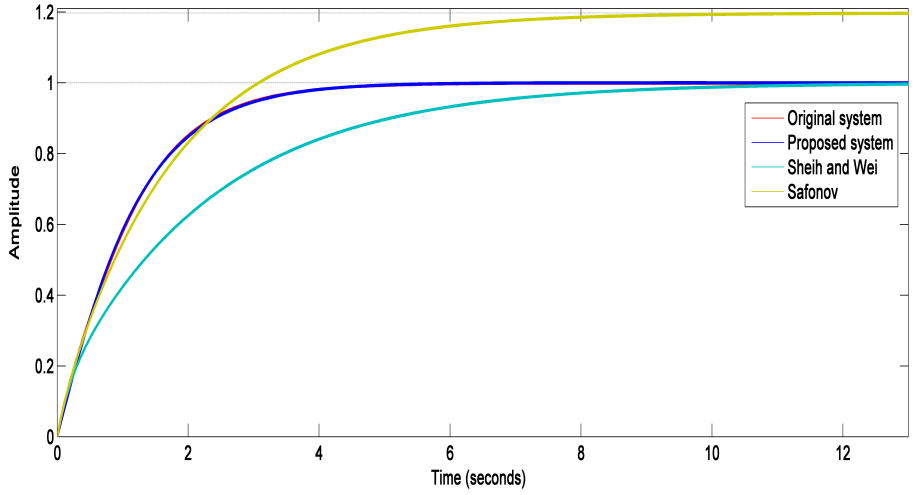
Now, for the second-order reduced model, the partition matrix is

$$\left[\begin{array}{c} 1 \\ \hline 1.889 \\ -2.5563 \\ -2.7863 \end{array}\right] = \left[\begin{array}{cc|cc} 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ \hline 1.889 & -1 & 0 & 0 \\ 2.5563 & 1.889 & 0 & 0 \end{array}\right] \times \left[\begin{array}{c} \hat{a}_{12} \\ \hat{a}_{13} \\ \hline 0 \\ 0 \end{array}\right] + \left[\begin{array}{c} \hat{a}_{21} \\ \hat{a}_{22} \\ \hline 0 \\ 0 \end{array}\right] \tag{3.21}$$

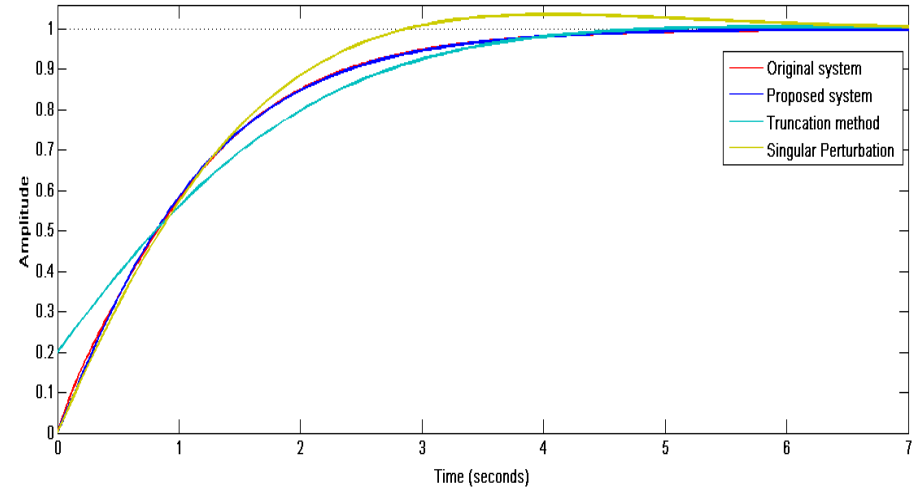
For calculating denominator using $\hat{A}_1 = C^{-1}{}_{21} \times \hat{C}_2$, we get $\hat{A}_1 = \begin{bmatrix} 1.2434 \\ 0.2075 \end{bmatrix}$. And using BBBC algorithm to search optimal coefficient of numerator, we get $x_1 = 3.10735$ and



(a)



(b)



(c)

FIGURE 3.3: Step response comparision for example 1

TABLE 3.2: Comparison between various Model Order Reduction techniques with respect to ISE

Model Reduction Technique	Reduced Order Model	ISE
Proposed Method	$\frac{0.28389s+1.00043}{0.3986s^2+1.3744s+1}$	1.1478×10^{-3}
Sikander and Prasad [20]	$\frac{0.6997s+0.6997}{s^2+1.45771s+0.6997}$	27.7989×10^{-3}
Desai and Prasad [18]	$\frac{0.8058s+0.7944}{s^2+1.65s+0.7944}$	2.8358×10^{-3}
Truncation Method [21]	$\frac{7s^2+24s+24}{35s^2+50s+24}$	70.138×10^{-3}
Routh Hurwitz [22]	$\frac{20.57s+24}{30s^2+42s+24}$	97.41283×10^{-3}
Pade Approximation [23]	$\frac{0.7311s+2.5088}{s2+3.4481s+2.5088}$	1.234×10^{-3}
Singular Perturbation [24]	$\frac{0.02957s^2+0.6925s+2.501}{s^2+3.395s+2.501}$	9.449
Balanced Realization [25]	$\frac{0.8216s+0.4542}{s^2+1.268s+0.4663}$	14.19×10^{-3}
Parmar Method [26]	$\frac{0.7442575s+0.6991576}{s^2+1.45771+0.6997}$	17.4381×10^{-3}
Chen et al.[27]	$\frac{0.6997s+0.6997}{s^2+1.4577s+0.6997}$	27.7989×10^{-3}
Pal [28]	$\frac{s+34.2465}{s^2+239.882s+34.2465}$	19.4603
Parthasarathy and Jayasimha [29]	$\frac{s+0.6997}{s^2+1.45771s+0.6997}$	342.0218×10^{-3}
Davision[30]	$\frac{-s^2+2}{s^2+3s+2}$	2735.7076×10^{-3}
Sheih and Wei [31]	$\frac{s+2.3014}{s^2+5.7946s+2.3014}$	1474.5453×10^{-3}
Saifonov and Chiang [32]	$\frac{s+5.403}{s^2+8.431s+4.513}$	1786.6177×10^{-3}

$x_2 = 1.00944$. Hence, the realized reduced-order model obtained is

$$\tilde{G}_2(s) = \frac{3.1084s + 1.0005}{0.2075s^2 + 1.2434s + 1} \tag{3.22}$$

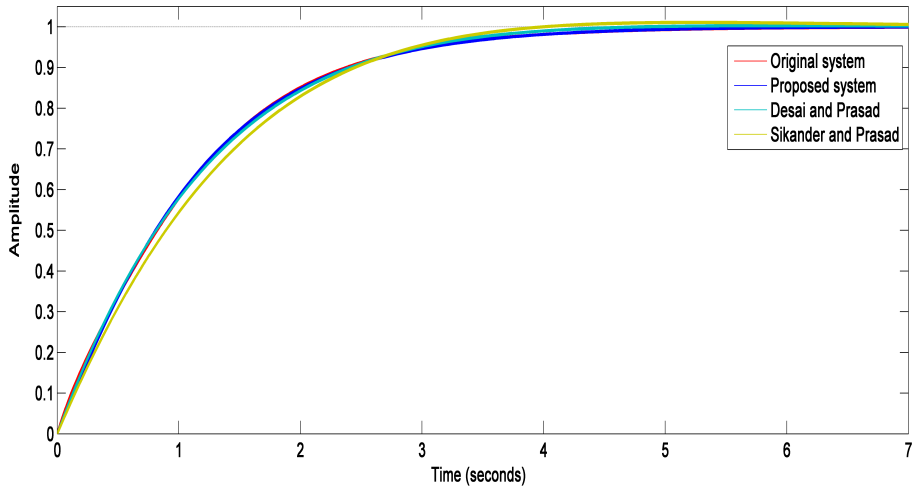
the obtained reduced order model shows **ISE of** 38.1244×10^{-3}. The obtained transfer function from Desai and Prasad mixed method [18] is

$$\tilde{G}_{DP}(s) = \frac{2.06774s + 0.43184}{s^2 + 1.17368s + 0.43184} \tag{3.23}$$

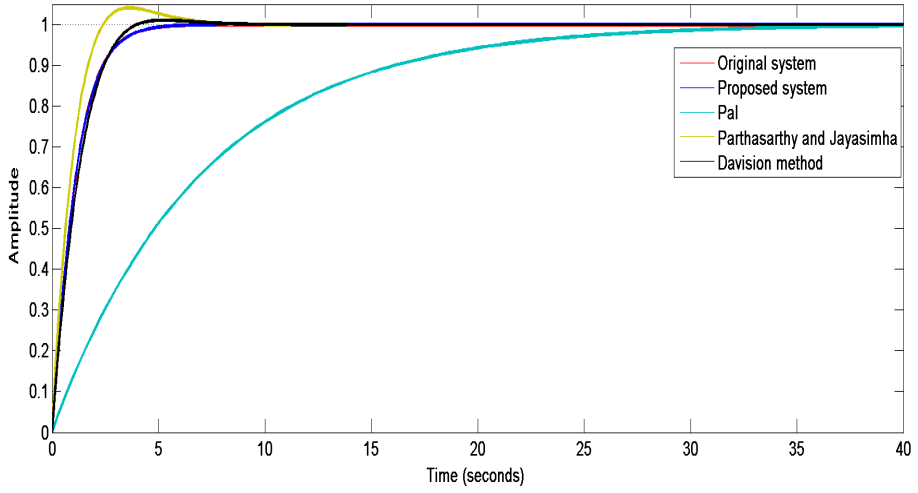
with **ISE=19.2386** and the model derived from Sikander and Prasads mixed method [20] is

$$\tilde{G}_{SP}(s) = \frac{16.92s + 5.263}{s^2 + 6.893s + 5.262} \tag{3.24}$$

with **ISE=**69.569×10^{-3}. From the ISE analysis, it is found that that the proposed method gives least ISE, i.e. modelling error is minimized. Fig. 3.6 shows the comparison of the step response of original 8th order system $G(s)$ with reduced second order



(a)



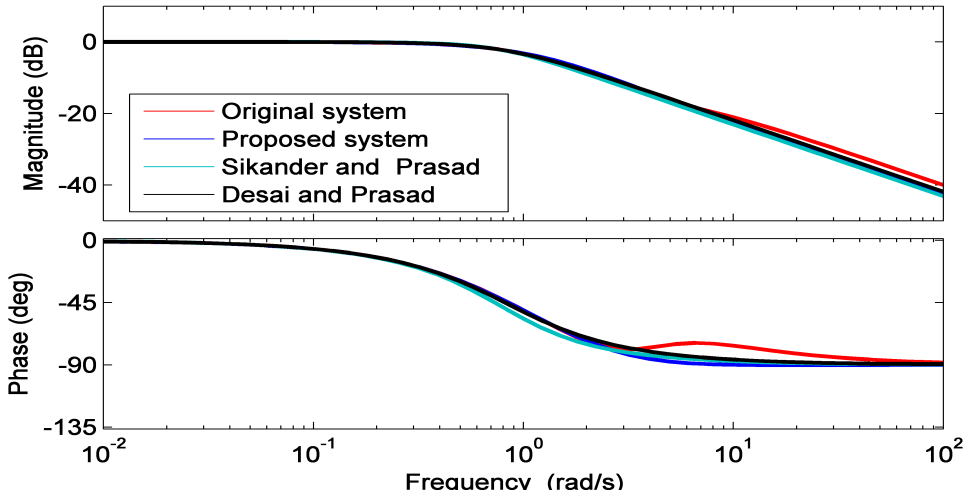
(b)

FIGURE 3.4: Step response comparision for example 1

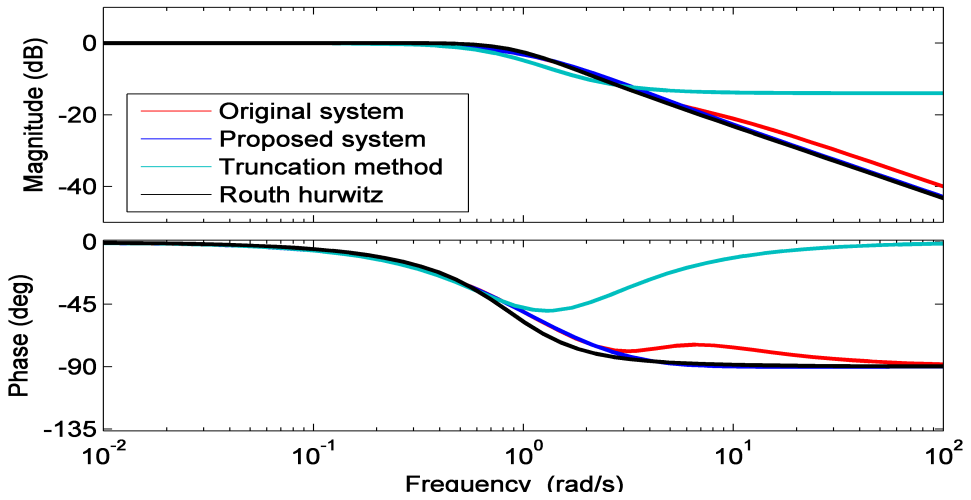
system obtained using the proposed method $\tilde{G}_2(s)$, Desai and Prasad mixed method [18] $\tilde{G}_{DP}(s)$ and Sikander and Prasad mixed method [20] $\tilde{G}_{SP}$(s). Further, from the frequency response characteristic shown in Fig. 3.7, it can be concluded that the proposed method is nearly in tune with frequency response characteristics of the original model.

Example 3: Consider an example of two-input, two-output (TITO) system from [18] as

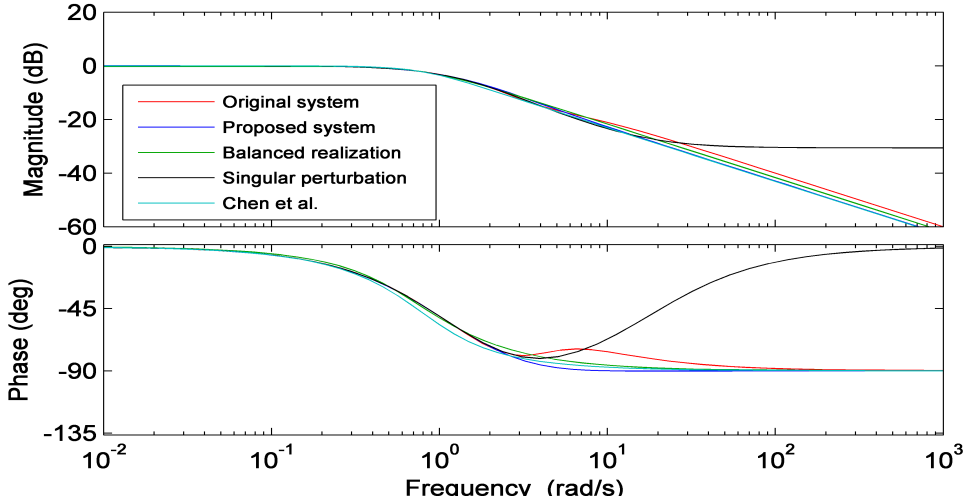
$$[G(s)] = \begin{pmatrix} G_{11} & G_{12} \\ G_{21} & G_{22} \end{pmatrix} = \begin{pmatrix} \frac{2(s+5)}{(s+1)(s+10)} & \frac{(s+4)}{(s+2)(s+5)} \\ \frac{(s+10)}{(s+1)(s+20)} & \frac{(s+6)}{(s+2)(s+3)} \end{pmatrix} \tag{3.25}$$



(a)

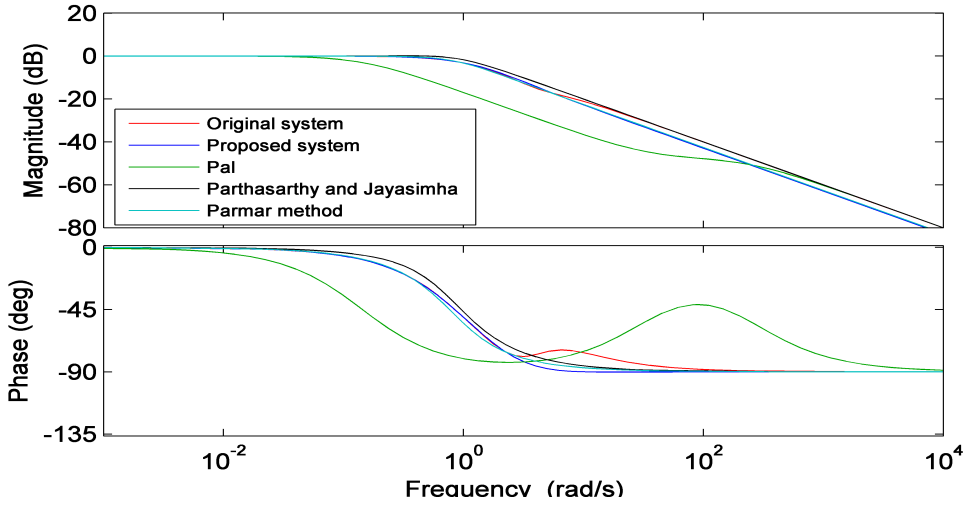


(b)

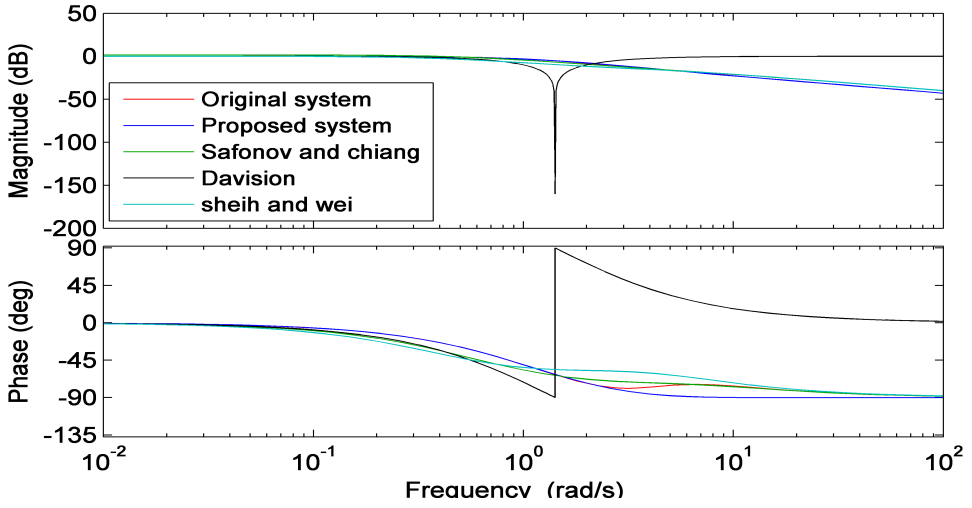


(c)

FIGURE 3.5: Frequency response comparision for example 1



(a)



(b)

FIGURE 3.6: Frequency response comparision for example 1

$[G(s)]$ can be further written as $[G(s)] = \frac{1}{D_n(s)} \begin{pmatrix} a_{11}(s) & a_{12}(s) \\ a_{21}(s) & a_{22}(s) \end{pmatrix}$ where

$$D_n(s) = s^6 + 41s^5 + 571s^4 + 3491s^3 + 10060s^2 + 13100^s + 6000$$

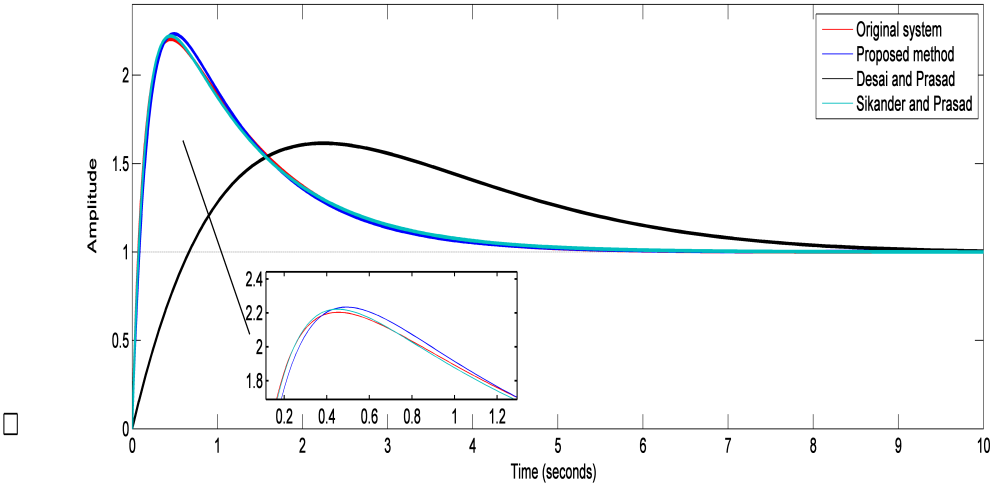


FIGURE 3.7: Step response comparisons for Example 2.

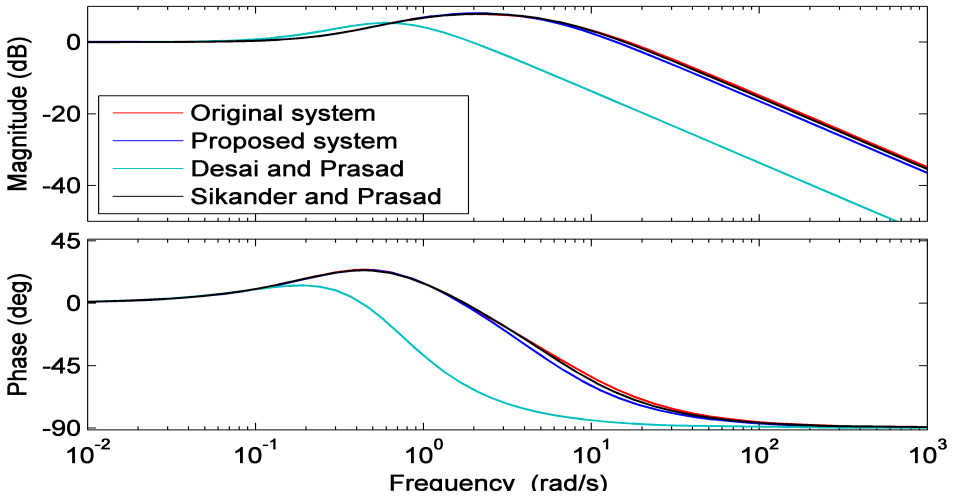


FIGURE 3.8: Frequency response comparison for Example 2.

where,

$$\begin{aligned} a_{11}(s) &= 2s^5 + 70s^4 + 762s^3 + 3610s^2 + 7700s + 6000 \\ a_{12}(s) &= s^5 + 38s^4 + 459s^3 + 2182s^2 + 4160s + 2400 \\ a_{21}(s) &= s^5 + 30s^4 + 331s^3 + 1650s^2 + 3700s + 3000 \\ a_{22}(s) &= s^5 + 42s^4 + 601s^3 + 3660s^2 + 9100s + 6000 \end{aligned} \tag{3.26}$$

The reduced model of $[G(s)]$ using our proposed scheme is presented in Table 3.3. For comparison, with model obtained using Desai and Prasads mixed method [18] and Sikander and Prasad mixed method [20]. The obtained results show that the proposed method

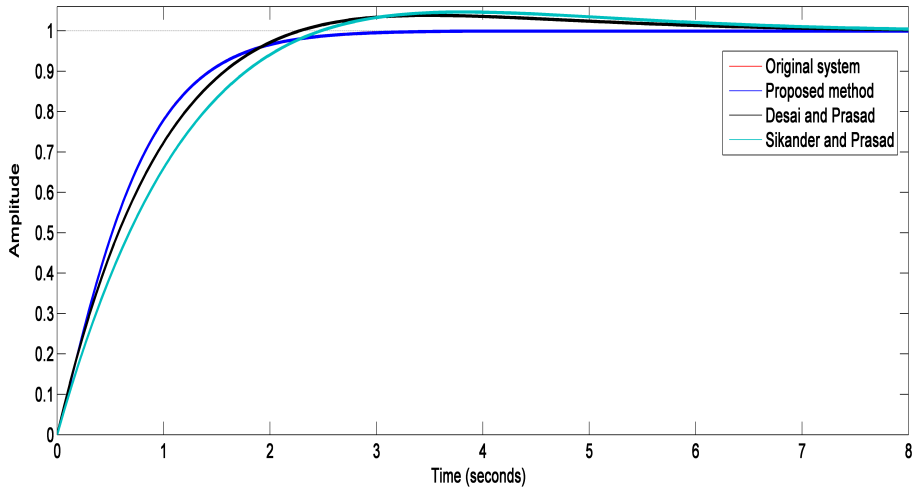
is far superior to other two methods. This is also observed through step response and frequency response characteristic as shown in Fig.3.8, and Fig.3.9, 3.10 respectively.

TABLE 3.3: Comparison between various MOR techniques with respect to ISE

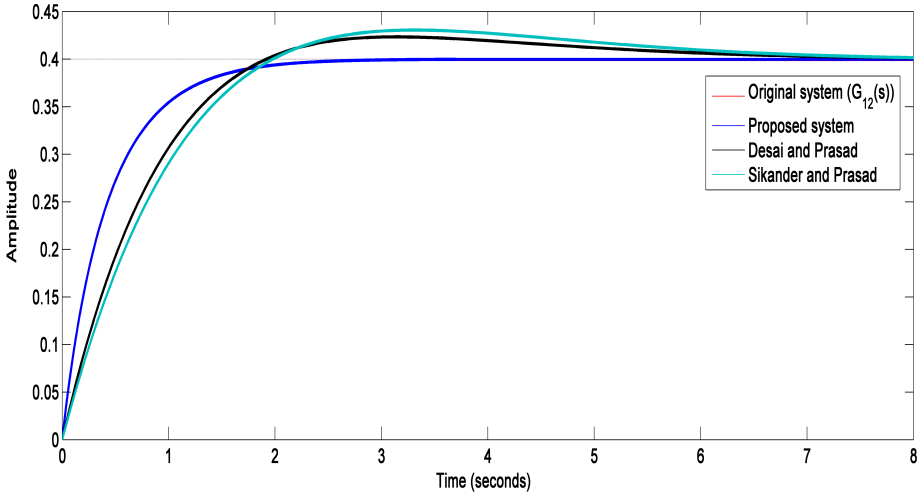
Model	Method	Transfer function	ISE
$G_{11}(s)$	**Proposed Method**	$\frac{0.2023s+1}{0.10224s^2+1.10237s+1}$	7.91154×10^{-7}
	Sikander & Prasad [20]	$\frac{0.7938s+0.6181}{s^2+1.34952s+0.6181}$	0.1672
	Desai & Prasad [18]	$\frac{0.9475s+0.7091}{s^2+1.548267s+0.7091}$	0.0701
$G_{12}(s)$	**Proposed Method**	$\frac{0.10084s+0.4}{0.10097s^2+0.7021s+1}$	1.27668×10^{-8}
	Sikander & Prasad [20]	$\frac{0.42728s+0.24717}{s^2+1.34952s+0.6181}$	0.0958
	Desai & Prasad [18]	$\frac{0.4892s+0.2837}{s^2+1.548267s+1}$	0.0602
$G_{21}(s)$	**Proposed Method**	$\frac{0.05025s+0.5}{0.05052s^2+1.0505s+1}$	5.6205×10^{-9}
	Sikander & Prasad [20]	$\frac{0.37952s+0.309}{s^2+1.34952s+0.6181}$	0.03121
	Desai & Prasad [18]	$\frac{0.455s+0.3546}{s^2+1.548267s+0.7091}$	0.0101
$G_2(s)$	**Proposed Method**	$\frac{0.1676s+1}{0.1673s^2+0.8343s+1}$	8.4009×10^{-8}
	Sikander & Prasad [20]	$\frac{0.93382s+0.6181}{s^2+1.34952s+0.6181}$	0.2004
	Desai & Prasad [18]	$\frac{1.126s+0.7091}{s^2+1.548267s+1}$	0.0636

Example 4 Let us take another example of TITO power system plant of order 10 [**?**] to strengthen the reliability of proposed method as:

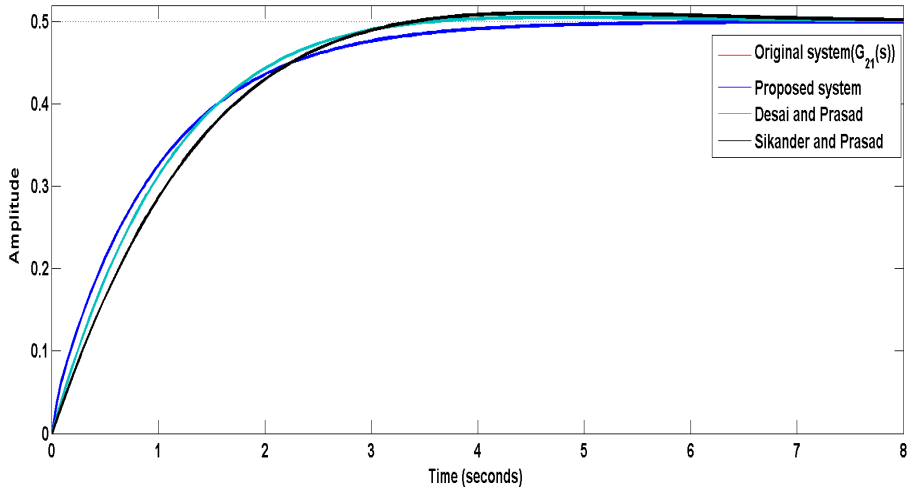
$$A = \begin{bmatrix} -0.5517 & 0 & -0.3091 & 0 & 0 & 0 & 0 & 0 & 0 & 0.1695 \\ -0.0410 & 0 & -0.3050 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 314.1593 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 9.5540 & 0 & -0.8660 & -20 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -0 & 0 & 0 & 0 & 0.0421 & -0.0328 \\ -0.1962 & 10.8696 & -0.1672 & 0 & 0 & -10.8696 & 0 & 0 & 0 & 0 \\ -0.9386 & 51.9849 & -0.7999 & 0 & 0 & -41.1153 & -10.8696 & 0 & 0 & 0 \\ -0.9386 & 51.9849 & -0.7999 & 0 & 0 & -41.1153 & -10.8696 & -0.1 & 0 & 0 \\ 0 & 0 & 0 & -1000 & -1000 & 0 & 0 & 1000 & -20 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1.0526 & -0.8211 \end{bmatrix}$$


Original system
Proposed method
Desai and Prasad
Sikander and Prasad
Amplitude
Time (seconds)

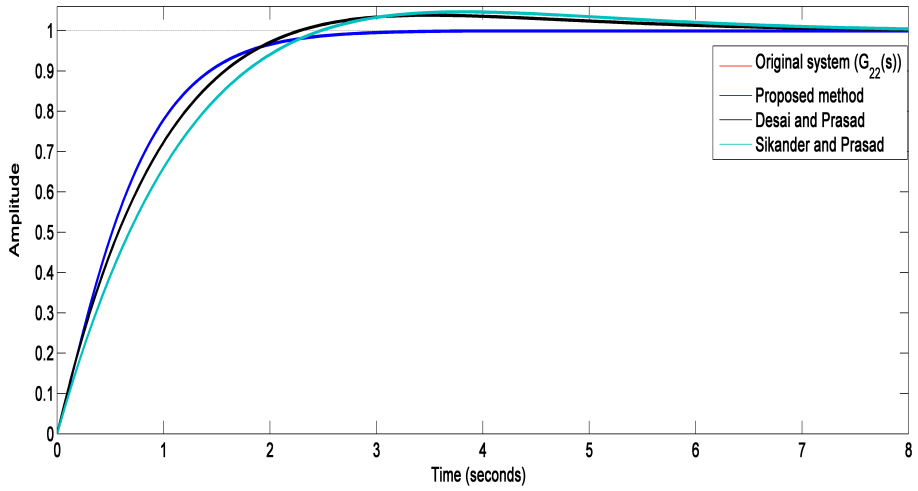
(a)


Original system ($G_{12}(s)$)
Proposed system
Desai and Prasad
Sikander and Prasad
Amplitude
Time (seconds)

(b)

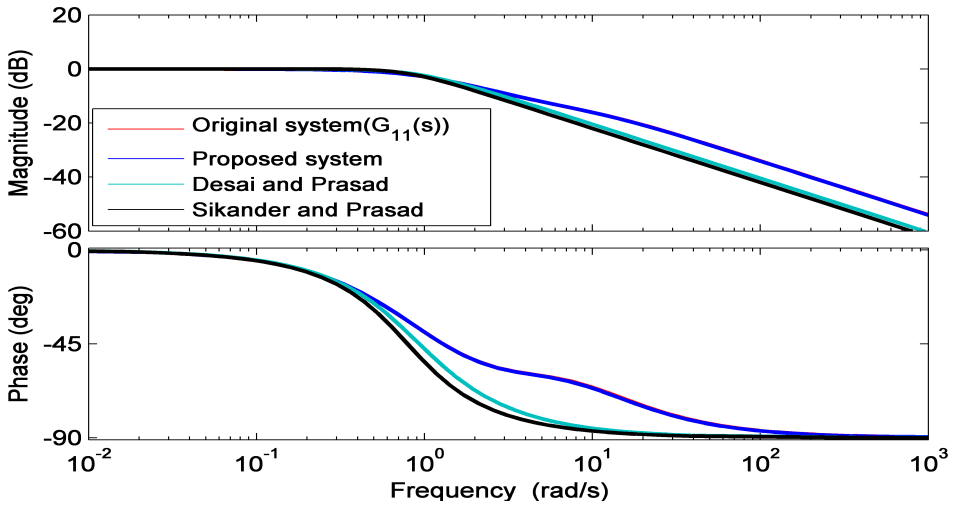

Original system($G_{21}(s)$)
Proposed system
Desai and Prasad
Sikander and Prasad
Amplitude
Time (seconds)

(c)



(d)

FIGURE 3.9: Step response comparisons of (a) $G_{11}(s)$, (b) $G_{12}(s)$, (c) $G_{21}(s)$, and (d) $G_{22}(s)$ for Example 3.



(a)

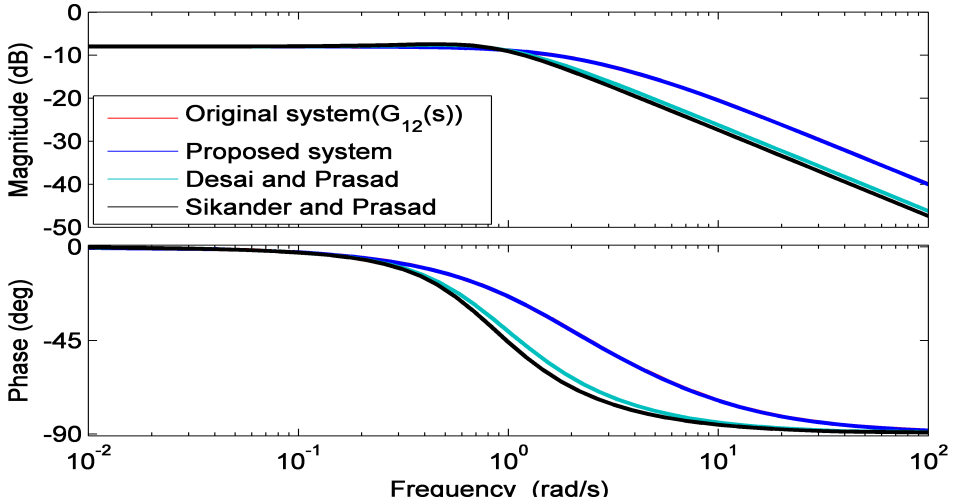
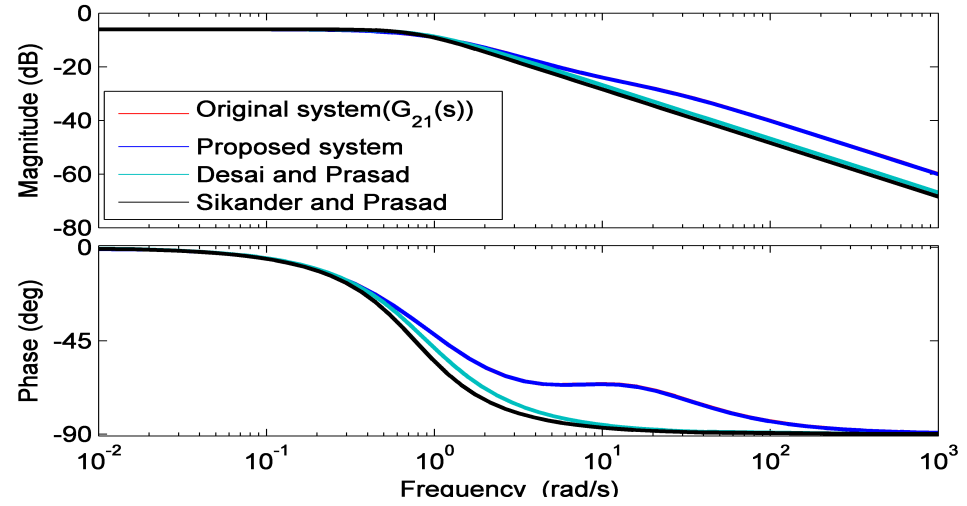
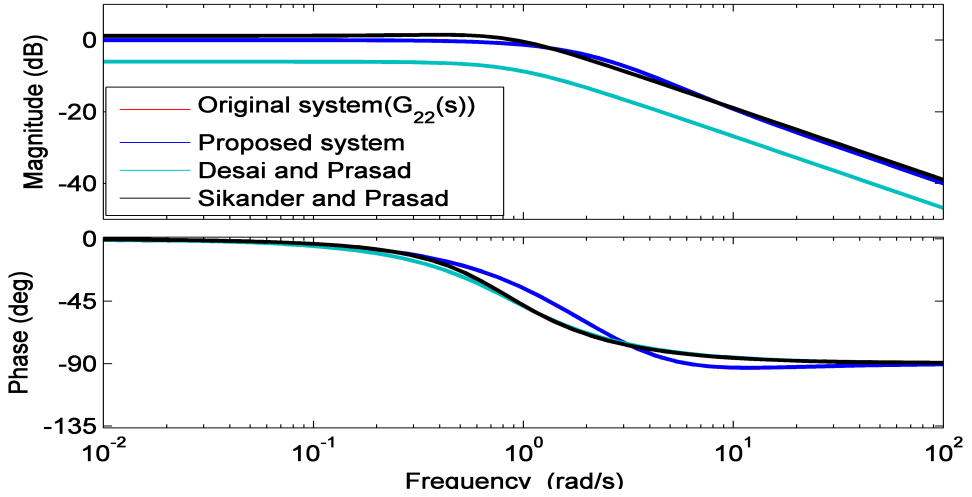


FIGURE 3.10: Frequency response comparison of (a) G_{11} (b) G_{12} for example 3.



(a)



(b)

FIGURE 3.11: Frequency response comparison of (a) G_{21} (b) G_{22} for example 3.

$$B = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1000 & 0 \\ 0 & 0.0926 & 0 & 0 & 0 & 0.4428 & 2.1179 & 2.1179 & 0 & 0 \end{bmatrix}^T$$

$$C = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.4777 & 0 & -0.0433 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, D = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \tag{3.27}$$

Given state space form is converted into transfer function form

$$\begin{bmatrix} G_{11}(s) \\ G_{21}(s) \end{bmatrix} = \frac{1}{D(s)} \begin{bmatrix} 29.08s^8 + 1868s^7 + 46100s^6 + 545900s^5 + 3185000s^4 + 8702000s^3 \\ +12060000s^2 + 7606000s + 648300 \\ -1.26s^8 - 85.17s^7 - 2089s^6 - 25680s^5 - 109800s^4 - 711800s^3 - 1083000s^2 \\ -297200s - 19430 \end{bmatrix} \tag{3.28}$$

where $D(s) = s^{10} + 64.21s^9 + 1596s^8 + 19470s^7 + 126800s^6 + 503600s^5 +$
$1569000s^4 + 3240000s^3 + 4061000s^2 + 2905000s + 253100$
and

$$G_{11}(s) = \frac{29.08s^8 + 1868s^7 + 46100s^6 + 545900s^5 + 3185000s^4 + 8702000s^3 + 12060000s^2 + 7606000s + 648300}{D(s)} \tag{3.29}$$

$$G_{21}(s) = \frac{-1.26s^8 - 85.17s^7 - 2089s^6 - 25680s^5 - 109800s^4 - 711800s^3 - 1083000s^2 - 297200s - 19430}{D(s)} \tag{3.30}$$

For $G_{11}(s)$ in (3.29), after arranging it in Routh form, we get required $c_i's$ as $c_0 = 2.561$, $c_1 = 0.652$,$c_2 = -0.9327$ $c_3 = 1.836$, $c_4 = -17.7491$, $c_5 = 179.2176$ and $c_6 = -1792.3375$. Using these value we can calculate denominator coefficient $\hat{A}_2$ and the obtained denominator is

$$\tilde{D}_{11}(s) = 2.3108s^3 + 2.7911s^2 + 10.2646s + 1$$

and the numerator coefficients are obtained using BBBC algorithm which is found out to be

$$\tilde{N}_{11}(s) = 17.42257s^2 + 24.6483s + 2.8875$$

Hence, the third-order reduced model is

$$\tilde{G}_{11}(s) = \frac{\tilde{N}_{11}(s)}{\tilde{D}_{11}(s)} = \frac{17.42257s^2 + 24.6483s + 2.8875}{2.3108s^3 + 2.7911s^2 + 10.2646s + 1} \tag{3.31}$$

The **ISE of (3.31) is 68.26931**. The only problem with above system is that dc gain obtained does not match with the original system model. The DC gain shown by original

system is $G(0) = 2.561$ whereas reduced model shows a DC gain of $\tilde{G}(0) = 2.8875$. Since, DC gain is not matching, we made slight modification in algorithm. We fix the $\tilde{G}(0)$ same as $G(0)$ and then calculate the rest unknown numerator coefficients using BBBC.

$$\tilde{N}_{11dc}(s) = 16.8003s^2 + 25.6068s + 2.561$$

And $\tilde{G}_{11}(s)$ becomes

$$\tilde{G}_{11dc}(s) = \frac{\tilde{N}_{11dc}(s)}{\tilde{D}_{11dc}(s)} = \frac{16.8003s^2 + 25.6068s + 2.561}{2.3108s^3 + 2.7911s^2 + 10.2646s + 1} \tag{3.32}$$

wherein DC gain of both original system and reduced model matches accurately but with marginal increase of performance function (ISE) which come out to be 69.1327. For the same system, we applied Desai and Prasad mixed method [18] and found the reduced order model as

$$\tilde{G}_{11DP}(s) = \frac{\tilde{N}_{11DP}(s)}{\tilde{D}_{11DP}(s)} = \frac{3.2023099s^2 + 2.081667s + 0.2056}{s^3 + 0.85591s^2 + 0.92131s + 0.08027} \tag{3.33}$$

which shows an **ISE of 147.3** and it is more than the proposed model.

()

The step and frequency response characteristics shown in Fig. 3.11 and 3.12, have considerable differences in both models but the proposed method is far better then Desai and Prasads mixed method [18]. It can be observed in frequency response characteristic (Fig. 3.12) that in working range of frequency (50-60 Hz), the proposed model imitates original model very well and hence results into more robust controller design keeping in view minor deviation but with heavily reduced numerical task.

Now, let us take $G_{21}(s)$ of (3.30) the obtained $c_i's$ are $c_0 = -0.07676$, $c_1 = -0.2931$, $c_2 = 0.31714$, $c_3 = -0.76656$, $c_4 = 7.1841$, $c_5 = -72.3491$ and $c_6 = 723.5899$. Using these value we reduce system to third- order model. And the obtained denominator is $\tilde{D}_{21}(s) = 4.8812s^3 + 7.6535s^2 + 10.672s + 1$. Now, using BBBC algorithm we find out optimal numerator which shows to be $\tilde{N}_{21}(s) = -4.00055s^2 - 0.95028s - 0.0909$ and hence

$$\tilde{G}_{21}(s) = \frac{\tilde{N}_{21}(s)}{\tilde{D}_{21}(s)} = \frac{-4.00055s^2 - 0.95028s - 0.0909}{4.8812s^3 + 7.6535s^2 + 10.672s + 1} \tag{3.34}$$

This reduced transfer function gives **ISE=0.61606**. In this case also, DC gain of original system and reduced system does not match, therefore we follow the same procedure as

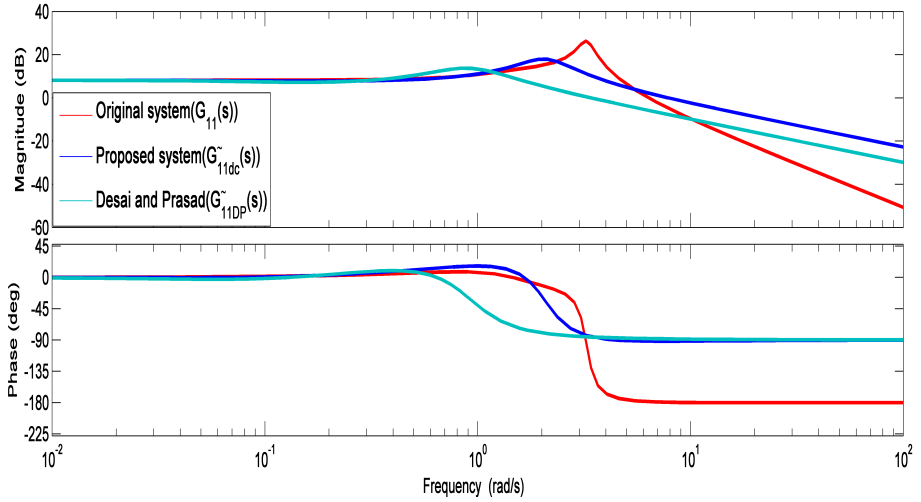


FIGURE 3.13: Frequency response comparison for Example 4

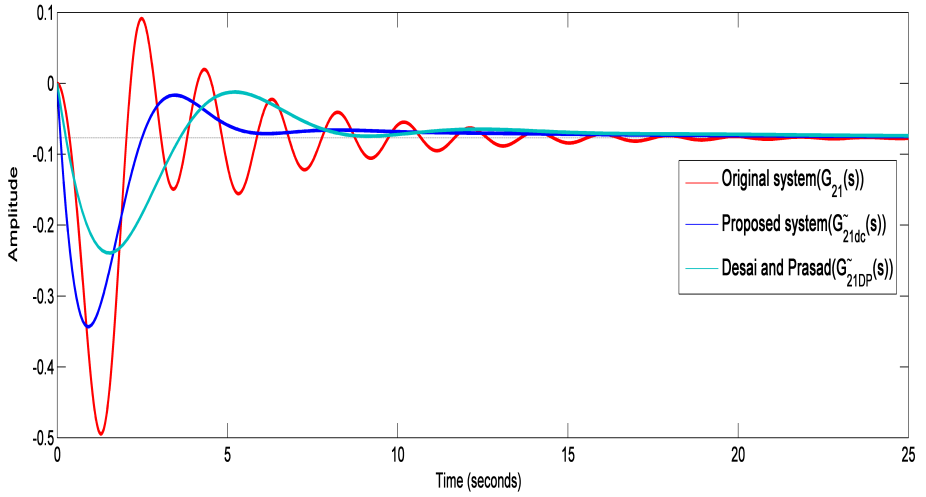


FIGURE 3.14: Step response comparison for Example 4.

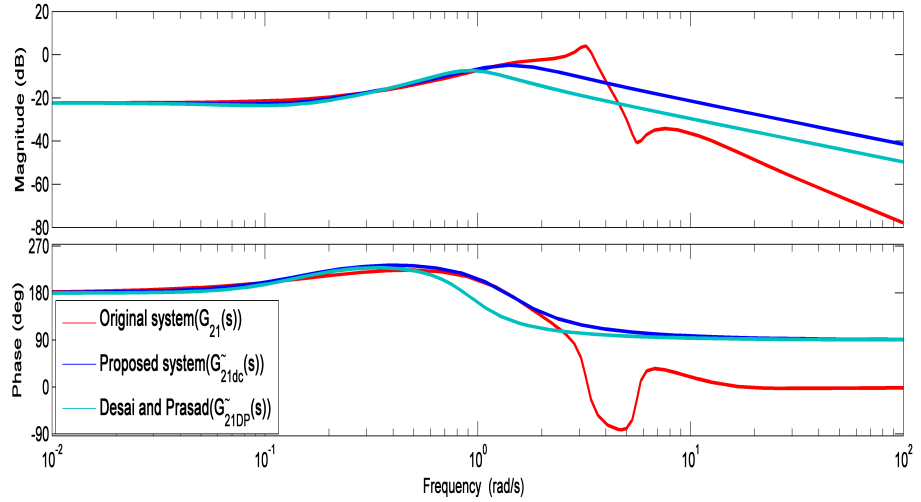


FIGURE 3.15: Frequency response comparison for Example 4

mentioned earlier and hence the realized reduced transfer function is

$$\tilde{G}_{21dc}(s) = \frac{\tilde{N}_{21dc}(s)}{\tilde{D}_{21dc}(s)} = \frac{-4.1s^2 - 0.9594s - 0.07676}{4.8812s^3 + 7.6535s^2 + 10.672s + 1} \tag{3.35}$$

The **ISE of this transfer function is 0.6278**, which is slightly more than $\tilde{G}_{21}(s)$. For the same problem, Desai and Prasad mixed method [18] is applied, the reduced order transfer function is obtained as

$$\tilde{G}_{21DP}(s) = \frac{\tilde{N}_{21DP}(s)}{\tilde{D}_{21DP}(s)} = \frac{-0.3287s^2 - 0.0716s - 0.00615}{s^3 + 0.8559s^2 + 0.92131s + 0.08027} \tag{3.36}$$

which shows an **ISE of 1.12626**.

The comparison between step response of original system ($G_21(s)$) , reduced third order model by proposed method ($\tilde{G}_{21}(s)$) and the reduced order model by Desai and Prasad mixed method ($\tilde{G}_{21DP}(s)$) are shown in Fig. 3.13. The step and frequency responses depicted have deviations but our proposed method is better then Desai and Prasads mixed method [18]. And again it can be observed that our model is emulating frequency response characteristics in working range of frequency very well, hence this model can be used successfully used keeping in mind a difference in step response.

Example 5

Time delays often appear in many real-world engineering systems either in state, the control input, or the measurements (output). Now let us consider in this example a time delayed system of order 7 as

$$G(s) = \frac{(4000s + 50000)e^{-0.3s}}{s^7 + 69s^6 + 1764s^5 + 20280s^4 + 102500s^3 + 221375s^2 + 187500s + 50000} \tag{3.37}$$

Delay element in (3.37) is Pade approximated, where the order of approximation is three, the generalized third order Pade approximant is

$$R_3(s) = \frac{120 - 60sT + 12(sT)^2 - (sT)^3}{120 + 60sT + 12(sT)^2 + (sT)^3} \tag{3.38}$$

and using (3.38), the resultant transfer function is

$$G_{pade}(s) = \frac{-4000s^4 + 110000s^3 - 666700s^2 - 15560000s + 222200000}{s^{10} + 109s^9 + 5191s^8 + 141300s^7 + 2396000s^6 + 25680000s^5 + 167500000s^4 + 610500000s^3 + 1111000000s^2 + 866700000s + 222200000} \tag{3.39}$$

The proposed method is applied to transfer function in (3.39), the obtained $c_i's$ are

$c_0 = 1$, $c_1 = -3.9705$,$c_2 = 10.484$and$c_3 = -23.7888$. Using these values, we reduce system to a second-order model. The denominator of this second order model is $\tilde{D}(s) = 2.928s^2 + 3.378s + 1$. Now, using BBBC algorithm, we find out an optimal numerator as $\tilde{N}(s) = 0.238432s^2 - 0.54329s + 1$. Using this numerator and denominator, reduced order model can be written as

$$\tilde{G}(s) = \frac{\tilde{N}(s)}{\tilde{D}(s)} = \frac{0.238432s^2 - 0.54329s + 1}{2.928s^2 + 3.378s + 1} \tag{3.40}$$

If Desai and Prasads mixed method [18] is applied on (3.39), the obtained transfer function is

$$\tilde{G}_{DP}(s) = \frac{0.0989s + 0.2328}{s^2 + 0.90802s + 0.2328} \tag{3.41}$$

The transfer function in (3.40) and (3.41) shows an **ISE of 0.0329 and 0.5928** respectively which clearly shows that proposed method is highly reliable. Fig. 3.15, 3.16 shows the comparison between original system ($G(s)$), Pad approximated model ($G_{pade}(s)$), proposed reduced model ($\tilde{G}(s)$), and Desai and Prasads reduced model ($\tilde{G}_{DP}(s)$). Note that here $G_{pade}(s)$ and $G(s)$ in Fig. 3.15 exactly imitates each other, hence distinction is not possible.

Example 6: Example considered in (3.37) has a delay of 0.3 seconds which is negligible, therefore let us consider another example having a delay of 2 seconds in output as.

$$G(s) = \frac{(4000s + 50000)e^{-2s}}{s^7 + 69s^6 + 1764s^5 + 20280s^4 + 102500s^3 + 221375s^2 + 187500s + 50000} \tag{3.42}$$

Using the Pade approximation in (3.38), the resultant transfer function is

$$G_{pade}(s) = \frac{-4000s^4 - 26000s^3 + 240000s^2 - 690000s + 750000}{s^{10} + 75s^9 + 2193s^8 + 31914s^7 + 251620s^6 + 1167000s^5 + 3357000s^4 + 6032000s^3 + 6433000s^2 + 3563000s + 750000} \tag{3.43}$$

The reduced model is obtained using proposed mixed method which is given as

$$\tilde{G}(s) = \frac{1.27s^2 - 0.898s + 1}{5.622s^2 + 4.2859s + 1} \tag{3.44}$$

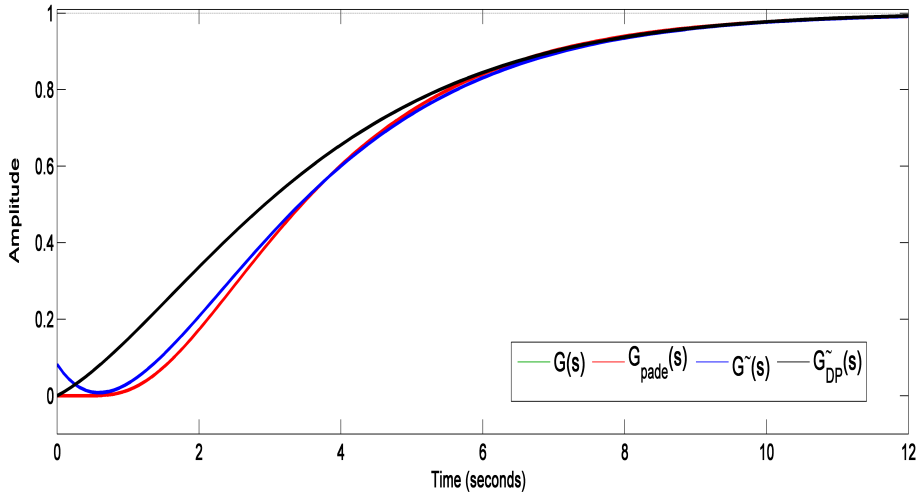


FIGURE 3.16: Step response comparison for Example 5

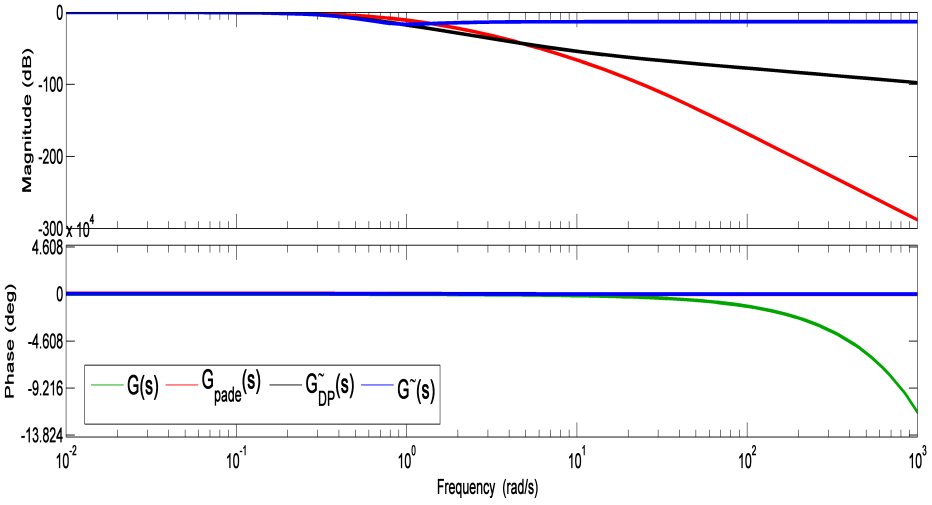


FIGURE 3.17: Frequency response comparison for Example 5

In order to show comparative results, the reduced order model is obtained using existing Desai and Prasads mixed method which is given as

$$\tilde{G}_{DP}(s) = \frac{-0.01336s + 0.14526}{s^2 + 0.6901s + 0.14526} \tag{3.45}$$

Fig. 3.17 and Fig. 3.18 show step and frequency response comparison of original system ($G(s)$), Pad approximated model ($G_{pade}(s)$), proposed reduced order model ($\tilde{G}(s)$), and Desai and Prasads reduced order model ($\tilde{G}_{DP}(s)$) respectively. The figs indicate that the proposed method is better approximating the reduced order model in comparison to exiting techniques.

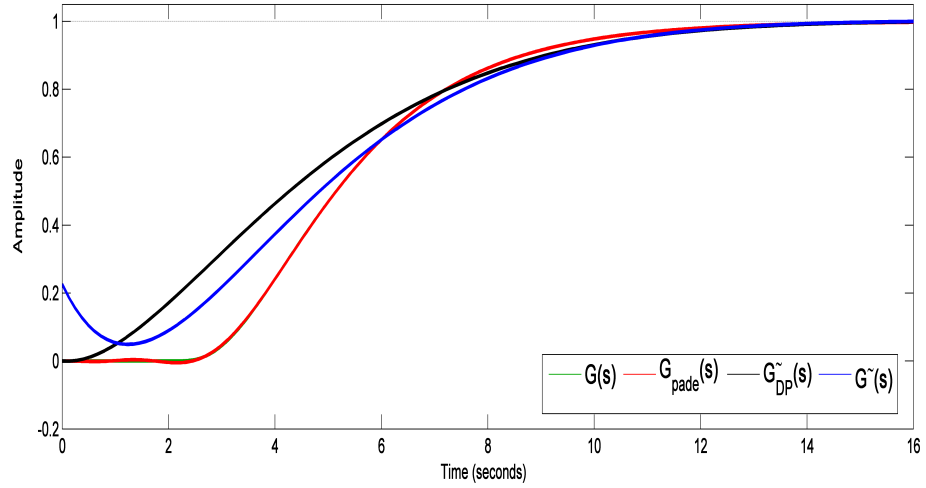


FIGURE 3.18: Step response comparison for Example 6

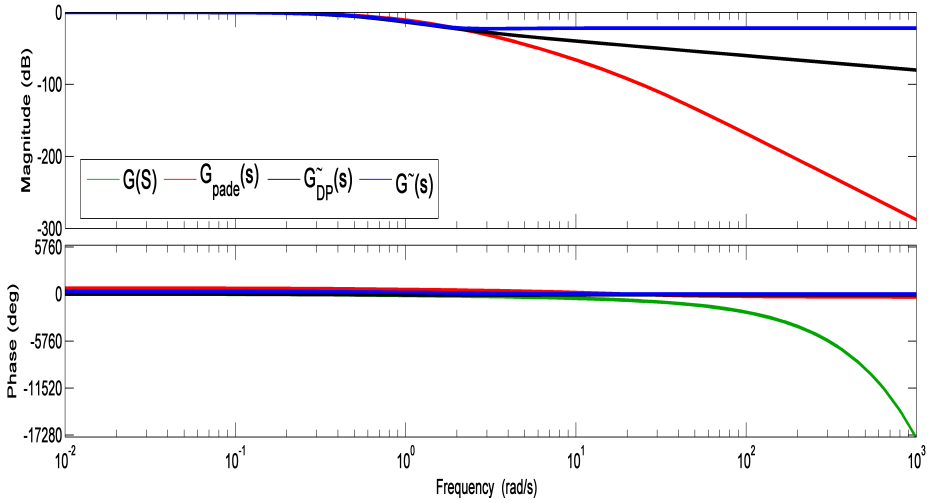


FIGURE 3.19: Frequency response comparison for Example 6

3.7 Conclusion

This chapter addresses the model-order reduction problem for LTI-SISO, MIMO and time delayed systems. We proposed a unifying framework for developing reduced-order model using the recently developed optimization scheme of BBBC. The performance and efficiency of the proposed method is demonstrated through application to several high-order systems and compared with the well known and recently developed methods. We observe that the proposed algorithm performs well in both time and frequency domain testing.

Chapter 4

Optimial PID Controller Design in Automatic Voltage Regulator (AVR) System Using BBBC

4.1 Introduction

The automatic voltage regulator (AVR) controls the output of the exciter so that generated voltage and reactive power changes in desired manner. In most modern power systems, the AVR is a controller that senses the generator output voltage and initiates the corrective action by changing the exciter control in the desired direction [33]. Over the past few decades, various control techniques have been formulated but the classical proportional-integral-derivative (PID) controller is considered to be outstanding among them. Moroever, the PID controllers are widely used as they induce stability and robustness into the systems [34], [35], [36]. Hence, PID is still a favored choice for AVR systems. It is observed that the PID tuning is a time consuming task especially in industrial systems which generally possesses non linearities, delay time and systems are of higher-order. In literature, many methods have been proposed for tuning of PID contoller. First paper on PID tuning is based on Ziegler-Nichols technique [35], [36]. But it suffers from major drawbacks such as (a) it is challenging to achieve best performance of system, (b) forces the process into the condition of marginal stability and (c) more importantly this method is not applicable for open loop unstable systems. Furthermore, many artificial intelligence techniques such as neuro-fuzzy system, neural network and fuzzy logic [37], [38], [39] have been schemed for fine tuning PID

controllers parameters. However, these techniques impose high convergence time and lengthy training process. Likewise numerous analytical techniques like lambda tuning method [40], Lyapunov thoerm [41] and stability boundary locus [42] are being used for efficient tuning of the PID controller parameter. With strong development in the field of evolutionary computation, various tuning techniques are being proposed using meta-heuristic optimization algorithms such as particle swarm optimization (PSO), Genetic Algorithm (GA) [43],[44], [45], [46], [47], [48]. Recently, Gaing [46] worked on a Particle Swarm optimization approach for optimum design of PID controller in AVR System as these algorithm are able to handle highly non-linear, mixed integer optimization problems of complex engineering systems however these algorithm requires huge memory to carry out computationally intensive task. On various occasions, these metaheuristc techniques are blended with other mathematical approaches for ease of computation. On the same line, Hasanien [49] proposed a mixed method which was combination of Taguchi method [50] and GA for tuning of AVR systems and demonstrated improved results.

BBBC [1] is utilized for efficient tuning of the PID controller for AVR systems. The results obtained shows exciting improvement in various performance time domain parameters.

4.2 Performance criterion

The main purpose of fitness function is to optimize various performance parameters. There are various performance criterion widely used in control systems such as integral of squared error (ISE), the integral of absolute error (IAE), integral of time weighted squared error (ITSE) and integral of time weighted absolute error (ITAE). Among which ISE and IAE weigh the error equally over the entire defined time interval whereas ITSE and ITAE gives higher weightage to error at later times in interval. The ISE, IAE, ITAE, and ITSE in time domain are illustrated below.

$$ISE = \int_0^\infty e^2(t)dt \tag{4.1}$$

$$IAE = \int_0^\infty |e(t)|\, dt \tag{4.2}$$

$$ITSE = \int_0^\infty te^2(t)dt \tag{4.3}$$

$$ITAE = \int_0^\infty t|e(t)|dt \tag{4.4}$$

Here,a fitness function has been suggested in time domain for BBBC-PID controller design. This performance criterion comprises of peak overshoot (M_p), rise time (t_r), settling time (t_s), steady state error (e_{ss}) and ISE, which is presented in discrete form in expression (9)

$$f_i = \sum_{i=0}^{M}[e^2(i\Delta t)] + w_1M_p + w_2t_r + w_3t_s + w_4e_{ss} \tag{4.5}$$

where, $M = \frac{5T}{\Delta t}$, $e(i\Delta t)$ is difference between reference voltage and output voltage at $(i\Delta t)$ instant, T is the time constant of process, Δt is taken as 0.1 sec for computational convenience and weight matrix $W = \{w_1, w_2, w_3, w_4\}$ decides relative significance of described parameters and is designed to realize required specification for controller design.

4.3 Linearised model of AVR system

The prominent components of AVR system are amplifier, exciter, the generator and the sensor as arranged in Fig. 4.1. The output voltage of generator V_t is regularly sensed by the voltage sensor and this voltage signal is conditioned and evaluated to see the fuctuation with respect to reference signal in the comparator. The resulting error voltage is fed as input to controller. The control signal emanating from controller is fed to amplifier and gradually to generator field winding through exciter. AVR system by nature is a nonlinear system with nonlinearity such as saturation, hysteresis, dead time etc. We know that the system analysis becomes highly complex when these nonlinearities are considered. Hence, this system is linearised taking into consideration dominant time constants and avoiding nonlinearities. The linearized model of AVR system as shown in Fig. 4.1. comprises of

- **Amplifier model**

The first order amplifier model is characterized by Amplifier gain K_a and Time constant τ_a. The transfer function of this system is given as:

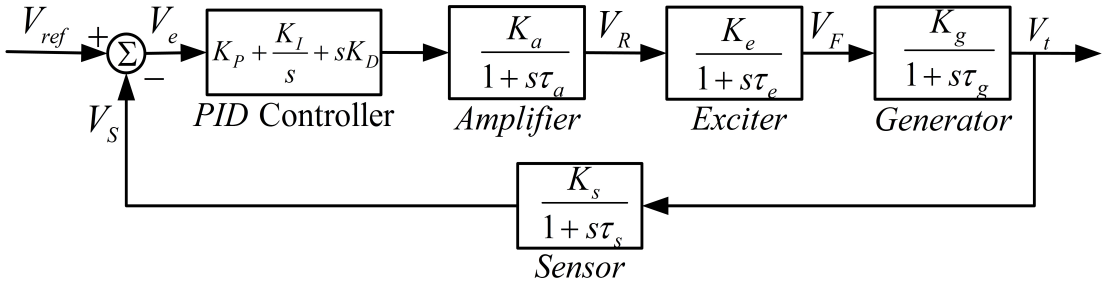


FIGURE 4.1: Block diagram of an AVR system with PID controller

$$\frac{V_R(s)}{V_e(s)} = \frac{K_a}{1 + s\tau_a} \tag{4.6}$$

where V_R and V_e corresponds to input voltage to exciter system and error voltage respectively and quintessential values of K_a and τ_a lies in the range of 10 to 400 and 0.02 to 0.1s respectively.

• Exciter model

The first order exciter model is characterized by gain K_e and Time constant τ_e. The transfer function for this system can be expressed as:

$$\frac{V_F(s)}{V_R(s)} = \frac{K_e}{1 + s\tau_e} \tag{4.7}$$

where V_F and V_R corresponds to input voltage to generator system and input voltage to exciter system respectively, the quintessential values of K_e and τ_e lies in the range of 1 to 10 and 0.4 to 1s respectively.

• Generator model

The first order Generator model is characterized by gain K_g and Time constant τ_g. The transfer function for this system can be written as:

$$\frac{V_t(s)}{V_F(s)} = \frac{K_g}{1 + s\tau_g} \tag{4.8}$$

V_t and V_F corresponds to terminal voltage of generator and input voltage to generator respectively. The quintessential values of K_g and τ_g lies in the range of 0.7 to 1 and 1 to 2s respectively.

• Sensor model

The first order Sensor model is characterized by gain K_s and Time constant τ_s. The transfer function is given as:

$$\frac{V_S(s)}{V_t(s)} = \frac{K_s}{1 + s\tau_s} \tag{4.9}$$

V_S and V_t are input voltage to sensor system and terminal voltage of generator system respectively. Here $K_s = 1$ and the sensor time constant τ_s ranges from 0.001 to 0.06s.

4.4 PID controller

The ideal version of PID controller is given by formula.

$$u(t) = K_P e(t) + K_I \int_0^t e(\tau)d\tau + K_D \frac{de}{dt} \tag{4.10}$$

This can be put in laplace domain as:

$$u(s) = (K_P + \frac{K_I}{s} + sK_D)E(s) \tag{4.11}$$

where u is the control signal and e is the control error ($e = r - y$). The reference value r, is also called as set-point value. The control signal is thus a sum of three terms: a proportional term that is proportional to the error, an integral term that is proportional to the integral of the error, and a derivative term that is proportional to the derivative of the error. The controller parameters are proportional gain k_p, integral gain k_i and derivative gain k_d. The controller in (10) can also be parameterized as

$$u(t) = K_P \left(e(t) + \frac{1}{T_i} \int_0^t e(\tau)d\tau + T_d \frac{de}{dt} \right) \tag{4.12}$$

where T_i is the integral time constant and T_d the derivative time constant. The proportional part acts on the present value of the error, the integral represents an average of past errors and the derivative can be interpreted as a prediction of future errors based on linear extrapolation [51].

The designer's challenge is to choose a set of controller gains that produces a system response within specifications. In (equation 4.11) it can be seen that this is equivalent

to choosing one gain and a pole-zero pattern - staying within the allowable configurations. The pole-zero pattern comprise of a zero that can be controlled and a pole that is constrained to be at the origin.

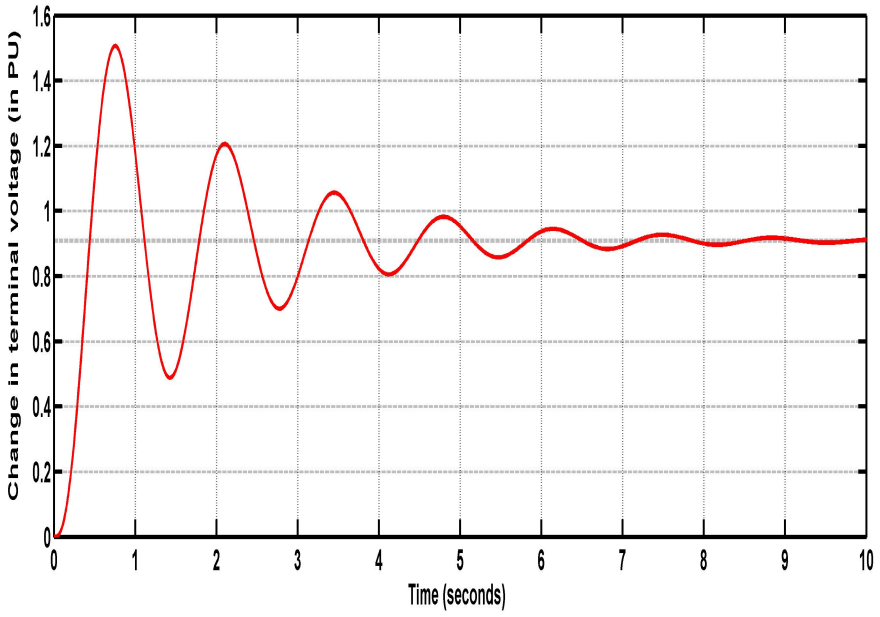


FIGURE 4.2: Step response of change in the terminal voltage without PID controller.

4.5 Implementation of BBBC-PID controller

This paper employs BBBC [1] optimization algorithm for optimal tuning of PID controller parameters i.e. $K = [K_P, K_I, K_D]$, such that desired transient response is obtained for AVR system.

4.5.1 Parameter initialisation

In every meta-heuristic optimization technique, first step is to generate population for unknown parameters. In the same way, in this proposed approach using BBBC, population is to be generated for unknown parameters, i.e,K_P, K_I and K_D, in vector form and it is interpreted as $K = [K_P, K_I, K_D]$. Each parameter here acts as an *individual* and elements in these individual parameter are *candidate*. These individuals are elaborated as given below.

$$\mathrm{K}_{Pi} \in K_P = \{K_{Pi} \,|\, K_{Pi} \in \mathbb{R}^{1\times N}, K_{P\min} \leq K_{Pi} \leq K_{P\max}\}$$

$\mathrm{K}_{Ii} \in K_I = \{K_{Ii} \,|K_{Ii} \in \mathbb{R}^{1\times N}, K_{I\,\min} \leq K_{Ii} \leq K_{I\,\max}\}$

$\mathrm{K}_{Di} \in K_D = \{K_{Di} \,|K_{Di} \in \mathbb{R}^{1\times N}, K_{D\,\min} \leq K_{Di} \leq K_{D\,\max}\}$

where $i = 1 \ldots N$ and K_{Pi}, K_{Ii} and K_{Di} are candidates of individual parameters, and hence size of matrix K is $N \times 3$.

4.5.2 BBBC-PID controller design algorithm

In this section, we are going to introduce an algorithm which facilitates in finding an optimal solution for individual parameters. The details of all the variables are mentioned in section IV-A. Exploration procedure of the BBBC-PID controller is given below.

Step 1: Define the lower and upper bounds of the three individual parameters (controller parameters) and initialize randomly the candidates of the population.

Step 2: For each initial candidate of the population, utilize the Routh-Hurwitz criterion to evaluate the closed-loop system stability and determine the values of the four performance criteria in the time domain, namely $M_P, E_{SS}, t_r,$ and $\mathrm{t_s}$.

Step 3: Calculate the fitness value of each candidate in the population using the performance criterion given by (9).

Step 4: Calculate the Center of mass as defined in (1) using fitness values obtained in step 3.

Step 5: Compare each candidates fitness value with defined least value(ϵ) that is $leastf_i = \min(f_k^{iter})$, $if(leastf_i \leq \varepsilon)$. Then jump to step (8), otherwise go to step (6).

Step 6: Generate new candidates around center of mass as defined in section (2) under pseudocode step (7).

Step 7: Go to step (3).

Step 8: Return optimal center of mass i.e. K_P, K_I, K_D.

4.6 Simulation results and discussion

Using above algorithm, PID controller is designed for AVR system using MATLAB. AVR system has following parameters: the gain components of AVR system are chosen

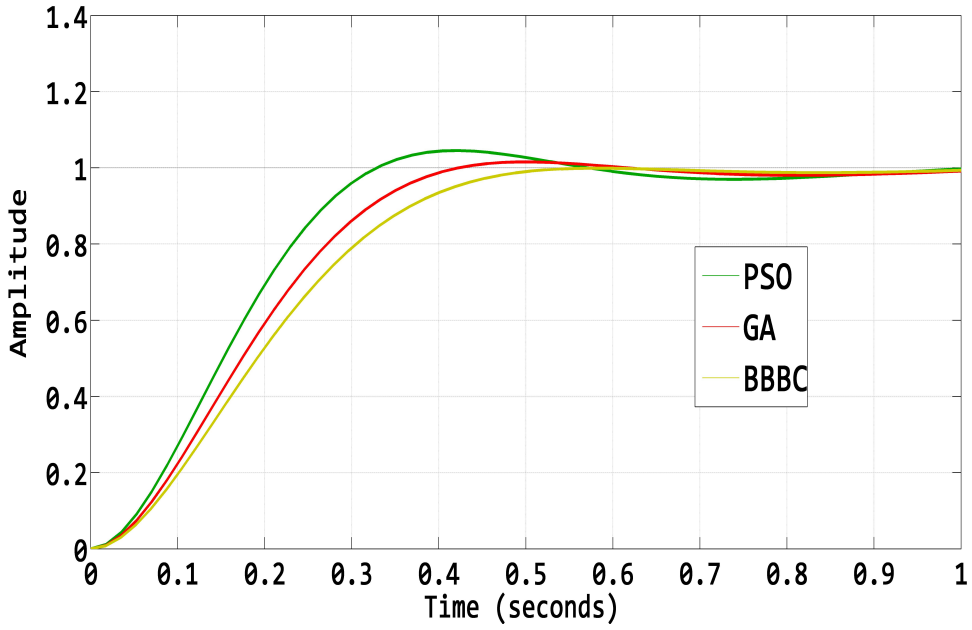


FIGURE 4.3: Response of change in the terminal voltage with PID controller.

to be $K_a = 10$, $K_e = 1$, $K_g = 1$ and $K_s = 1$. The time constant of the components are set as $\tau_a = 0.1s$, $\tau_e = 0.4s$, $\tau_g = 1s$ and $\tau_s = 0.01s$. The values of gain and time constant are taken from [46].

As seen from Fig. 4.2, the original model of AVR system has considerably high peak overshoot and large settling time. The rise time equals to 0.2607, the settling time equals to 6.9865 sec, the maximum peak overshoot (MPOS) equals to 50.69%, and the E_{SS} equals to 8.81%.

The proposed optimal controller design using BBBC gives best result. This tuned controllers response is compared with optimal PSO, optimal GA. Optimum models GA is carried out in matlab toolbox. In GA optimization model the optimal parameters are $K_{GA} = [0.772, 0.72, 0.319]$. In PSO modeling technique optimal values are $K_{PSO} = [0.67, 0.59, 0.26]$, and optimal values obtained using BBBC optimization technique is $K_{BBBC} = [0.599, 0.53, 0.2599]$. The simulation results are shown in Table 4.1. It can be observed from its time response parameter results (Table 4.1) that rise time, settling time and peak overshoot are lesser then all the modeling techniques mentioned in Table 4.1.

The test results obtained using PSO are far superior than GA, as PSO is computationally efficent then GA. In PSO candidates in swarm have knowledge of both its position and remaining candidates position, it does not loose this information and search is

TABLE 4.1: Comparision of optimum models

Modelling Techique	MPOS(%)	Rise time	Settling Time	E_{ss}(%)
Optimum GA	4.74	0.333	0.86	1
Optimum PSO	17	0.42	0.81	0.9
Optimum BBBC	1.7	0.3	0.46	0.48

carried by sharing this information all the way till solution is obtained. Where as in GA unfit candidates are abandoned and only healthy candidates are preserved, consequently population include share of finest individual. Moreover result obtained using BBBC technique are better then that obtained using PSO and GA. In BBBC each candidate in search space is weighted with respect to values returned by objective function. Importance of each candidate is measured and this information is used to form a solution (center of mass). More refined solution is obtained by creating population around the present solution (center of mass).

4.7 Conclusion

In this chapter a new optimization technique i.e. BBBC was used to optimally design a PID controller in automatic voltage regulator system for improvising its response to step input. The proportional gain [K_P], derivative gain [K_D] and Integral gain [K_I] are parameters in search space.The optimized value obtained yeilds better step response compared to earlier PSO and GA. As a result of this proposed BBBC-PID approach better decision can be actuated when disturbance occurs on exciter to maintain proper voltage profile across line.

Chapter 5

Novel Optimial PID Controller Design in Automatic Voltage Regulator (AVR) System Using Taguchi Combined BBBC (TC-BBBC)

5.1 Introduction

In previous chapter we discussed about controller design for AVR sustem using BBBC and a proposed objective function. During this problem it was noticed that computational effort to optimize this problem was very high due to complex objective function. And hence there was a need to reduce this computational effort. Basically, reduction in computational effort here means that reduction in number of variable. So, there was a need to find a method in which dimension of the problem can be reduced. During this search we came across a statistical method called as "Taguchi method", which is an optimization method or more appropriately here we use this for dimensionality reduction of an problem. This method has been applied to many practical electrical engineering system which is used to optimize system parameters [52], [53], [54], [55],[56]. This method is a very easy statistical technique which uses no sophisticated software package, we have used Microsoft excel and Matlab 2015.

This chapter presents a novel PID design technique using TC-BBBC method. A multi-objective function is formed which optimizes system parameters i.e. the maximum percentage overshoot (M_p), the rise time (t_r), the settling time (t_s) and the steady state error (e_{ss}). Whereas in optimization algorithm, k_p, k_i and k_d will form a search space. The optimum value of the design variable i.e. k_p, k_i and k_d will be obtained using Taguch method using Analysis of Mean (ANOM). We are also using Analysis of Variance (ANOVA) to choose two most dominant design variable. Then BBBC with the help of Matlab is used to obtain optimized value of two design variable. The efficacy of this method is then compared with PSO and GA optimization technique.

5.2 Linearised model of AVR system

The model has been explained in section 4.3.

5.3 Optimization by The Taguchi Method

The Taguchi method [57], [58] was proposed based on formation of orthogonal array, [59] this is a basis of dimensionality reduction i.e. reduction of design variable. The Orthogonal array in Taguchi method depends on number of factors and number of levels which are used to study variation in parameters. This is highly advantageous because of less number of experiments. For example if we have four factor and each factor has three levels then total experiments to totally describe the variation would be $3^4 = 81$. Whereas, Taguchi method only needs 9 experiment ot totally describe the experiment.

5.3.1 Orthogonal Array

Orthogonal arrays are employed in industrial experiment to study the effect of various control factor on the industrial system. The specialty of these array is that the columns for the independent variables are 'orthogonal' to each other.

Benefits

1) Valid conclusion can be retrieved over the entire region spanned by design variable.

2) Saving in computational effort.

3) Easy analysis of system behavior.

Data required for forming an Orthogonal Array

1) Number of factors to be studied.

2) Number of levels for each factor.

3) 2 factor interaction to be studied.

4) Special difficulties in running experiment.

Notation of Matrix Experiments $= L_{N_e}({N_l}^{N_f})$

where

L is an orthogonal array with following specification as given below.

N_e=Number of experiments.

N_l=Number of levels.

N_f=Number of factors.

All these variable are related to each other and satisfies following relation

$$N_e = (N_l - 1) \times N_f + 1 \tag{5.1}$$

Example of typical L_9 ***Orthogonal Array***

There are standard orthogonal arrays available, and each arrays represents number of independent design variables and levels . For example, if we have 4 independent variable and each variable having 3 level values. With this information, if we conduct an experiment to analyze the impact of 4 independent design variable, then for such case L_9 orthogonal array is a right choice. This L_9 orthogonal array assumes that there is no interaction between any two factor. Assumption of no interaction is valid for many cases. But, there are cases where interaction is inevitable and can be observed clearly.

TABLE 5.1: $L_9(3^4)$ Orthogonal Array

$L_9(3^4)$ **Orthogonal Array**					
	Independent Varaibles				
Experiment	**Variable 1**	**Variable 2**	**Variable 3**	**Variable 4**	**Performance Parameter Values**
1	**1**	**1**	**1**	**1**	**P1**
2	**1**	**2**	**2**	**2**	**P2**
3	**1**	**3**	**3**	**3**	**P3**
4	**2**	**1**	**2**	**3**	**P4**
5	**2**	**2**	**3**	**1**	**P5**
6	**2**	**3**	**1**	**2**	**P6**
7	**3**	**1**	**3**	**2**	**P7**
8	**3**	**2**	**1**	**3**	**P8**
9	**3**	**3**	**2**	**1**	**P9**

5.3.2 Properties of Orthogonal Array [60]

Following properties of Orthogonal Array helps in reducing total number of experiments to be carried out.

1) The vertical column under each independent variables of the above table has a special combination of level settings. All the level settings appears an equal number of times. For L9 array under variable 4 , level 1 , level 2 and level 3 appears thrice. This is called the balancing property of orthogonal arrays.

2) All the level values of independent variables are used for conducting the experiments.

3) The sequence of level values for conducting the experiments shall not be changed. This means one can not conduct experiment 1 with variable 1, level 2 setup and experiment 4 with variable 1 , level 1 setup. The reason for this is that the array of each factor columns are mutually orthogonal to any other column of level values. The inner product of vectors corresponding to weights is zero. If the above 3 levels are normalized between -1 and 1, then the weighing factors for level 1, level 2 , level 3 are -1 , 0 , 1 respectively. Hence the inner product of weighing factors of independent variable 1 and independent variable 3 would be

$$(-1 \times -1 + -1 \times 0 + -1 \times 1) + (0 \times 0 + 0 \times 1 + 0 \times -1) + (1 \times 0 + 1 \times 1 + 1 \times -1) = 0.$$

Coming back to our problem of optimizing PID values for controller design of AVR system. Now we will form an orthogonal array which will have four factors i.e. A, B, C and D. Here, factor A is propotional gain (k_p), B is integral gain (k_i), C is derivative gain (k_d) and D is saturation limit, saturation limit is fixed to 100 for the AVR system stability.

TABLE 5.2: Design Variables and Levels

Design Variable	**Level 1**	**Level 2**	**Level 3**
k_p	**0.6**	**0.7**	**0.8**
k_i	**0.5**	**0.6**	**0.7**
k_d	**0.2**	**0.25**	**0.3**
D	**100**	**100**	**100**

5.3.3 Performing the Experiment

After forming an orthogonal array as shown in Table 5.3, which describes the reduced number of experiments to be carried out. Now, using these values in Table 5.3 the experiment is carried out to calculate the values of $M_p(\%)$, $t_r(s)$, $t_s(s)$ and $E_{ss}(\%)$ as

TABLE 5.3: L_9 Orthogonal Array

Exp. No.	k_p	k_i	k_d	**D**
1	**0.6**	**0.5**	**0.2**	**100**
2	**0.6**	**0.6**	**0.25**	**100**
3	**0.6**	**0.7**	**0.3**	**100**
4	**0.7**	**0.5**	**0.25**	**100**
5	**0.7**	**0.6**	**0.3**	**100**
6	**0.7**	**0.7**	**0.2**	**100**
7	**0.8**	**0.5**	**0.3**	**100**
8	**0.8**	**0.6**	**0.2**	**100**
9	**0.8**	**0.7**	**0.25**	**100**

shown in Table 5.4. To obtain the values of all the system parameters described above we have used Matlab.

After performing the experiment the next task is to analyse the experimental data. This

TABLE 5.4: Result of Experiment performed

Exp. No.	$M_p(\%)$	$t_r(s)$	$t_s(s)$	$E_{ss}\%$
1	**1.79**	**0.52**	**0.47**	**1.45**
2	**0**	**1**	**1.78**	**2.84**
3	**0**	**1**	**1.89**	**3.86**
4	**2.34**	**0.41**	**1.03**	**0**
5	**1.8**	**0.37**	**0.985**	**1.35**
6	**9.34**	**0.42**	**1.739**	**1.42**
7	**4.45**	**0.34**	**1.158**	**1**
8	**11.5**	**0.393**	**0.949**	**0**
9	**8.25**	**0.36**	**0.785**	**1**

analysis is carried out using two important parameters i.e. ANOM and ANOVA. These statistical measures are used to assess the effect of variation in three design variable that are k_p, k_i and k_d, and to find relative importance of design variable respectively.

5.3.3.1 ANOM

1) *Overall Mean*: Average value all the performance parameter.

$$m = \frac{1}{9}\sum_{i=1}^{9} P \tag{5.2}$$

Where, P=[M_p t_r t_s E_{ss}], is performance parameters of the system. Overall mean is calculated for all the system parameters and is listed below in Table 5.5.

2) *Average Effect of a Design Variable at One Setting*: In this calculation one setting is chosen and effect of this setting on the design variable is calculated. For example, we

TABLE 5.5: Mean of system parameters P

	$M_p(\%)$	$t_r(s)$	$t_s(s)$	$E_{ss}\%$
m (overall mean)	**4.39**	**0.536**	**1.205**	**1.441**

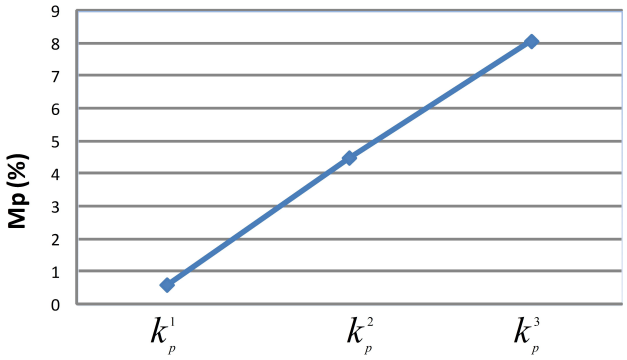
chose k_p of level 3 from Table 5.2, which is represented as k_p^3. Then, this calculation represents the effect of k_p^3 on performance parameter. Let us consider that we need effect of k_p^3 on performance parameter M_p, which is represented using below equation 5.3

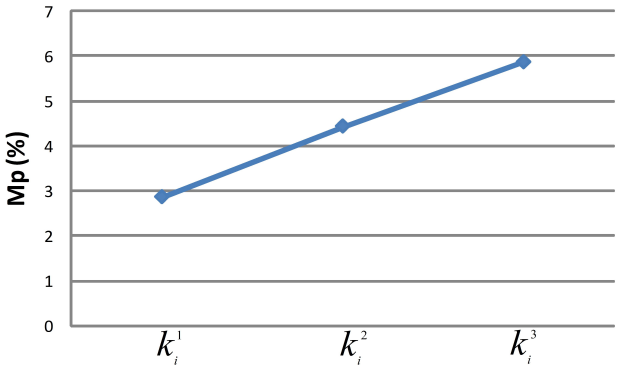
$$m_{k_p^3}(M_p) = \frac{1}{3}(M_p(7) + M_p(8) + M_p(9)) \tag{5.3}$$

Expected effect of all the performance parameter are given below in table

TABLE 5.6: M_p for all factors

Setting Factor	M_p **of** k_p	M_p **of** k_i	M_p **of** k_d
1	**0.595**	**2.85**	**7.542**
2	**4.493**	**4.433**	**3.53**
3	**8.065**	**5.862**	**2.085**





5.3.4 Analysis of Variance (ANOVA)

This test is also known by other name i.e. Fisher anaysis of variance. This test measures variance among groups, when there are more than two groups. More precisely this test

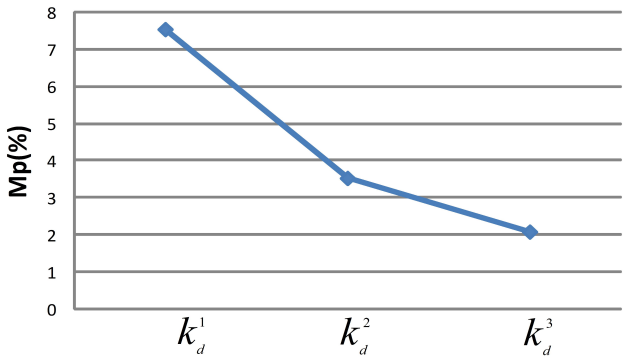
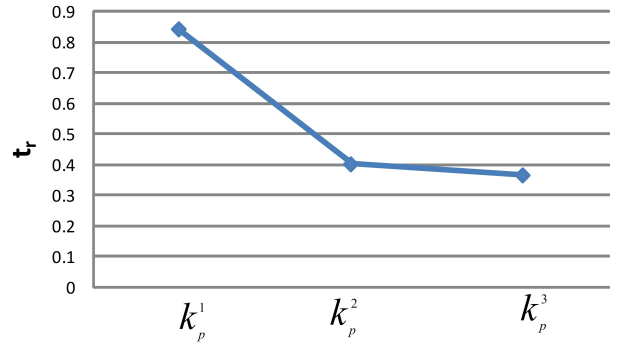
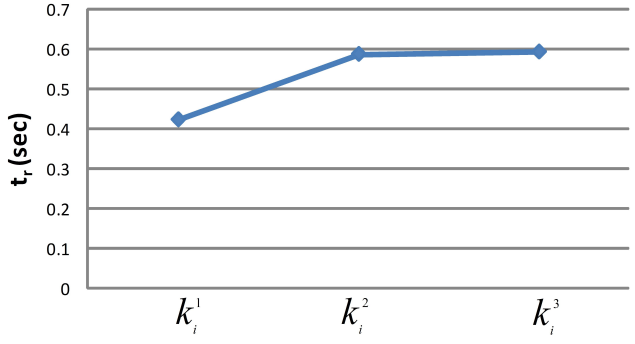


FIGURE 5.1: Setting parameters of k_p, k_i and k_d

TABLE 5.7: t_r for all factor

Setting Factor	t_r **of** k_p	t_r **of** k_i	t_r **of** k_d
1	**0.84**	**0.423**	**0.444**
2	**0.4**	**0.587**	**0.59**
3	**0.364**	**0.593**	**0.57**





measures the statistical significance of design variables. This is calculated here using equation given below.

$$SSFA = 3\sum_{i=1}^{3}(m_{Ai} - m)^2. \tag{5.4}$$

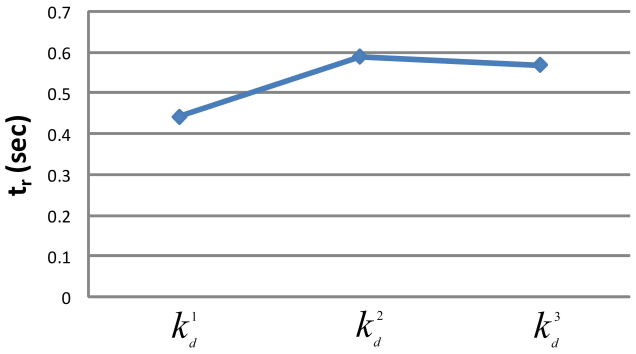
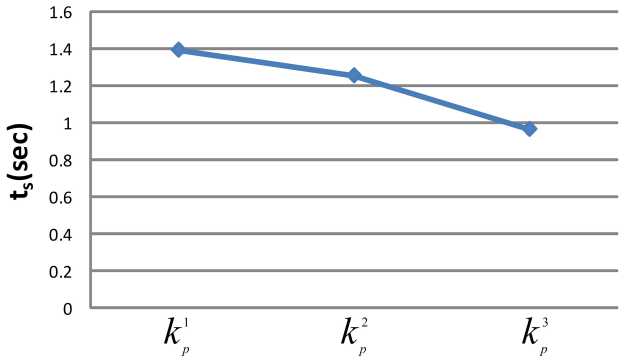
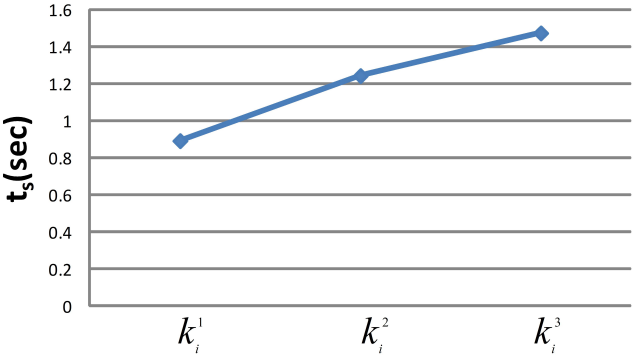


FIGURE 5.2: Setting parameters of k_p, k_i and k_d

TABLE 5.8: t_s for all factors

Setting Factor	t_s **of** k_p	t_s **of** k_i	t_s **of** k_d
1	**1.392**	**0.892**	**1.056**
2	**1.254**	**1.244**	**1.208**
3	**0.965**	**1.475**	**1.348**





$$SSFB = 3\sum_{i=1}^{3}(m_{Bi} - m)^2. \tag{5.5}$$

$$SSFC = 3\sum_{i=1}^{3}(m_{Ci} - m)^2. \tag{5.6}$$

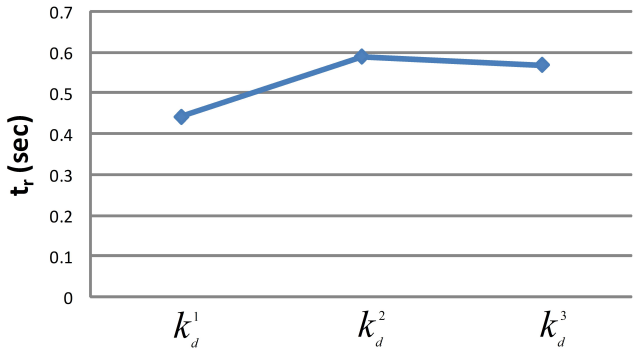
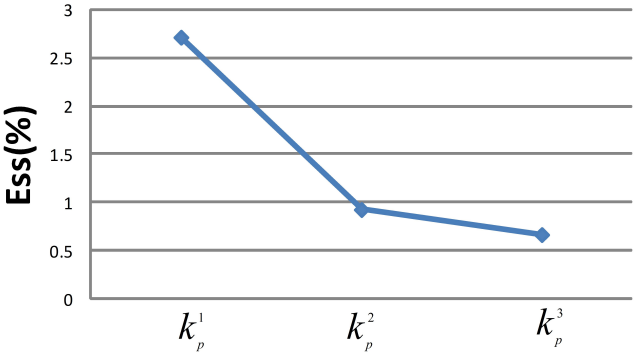
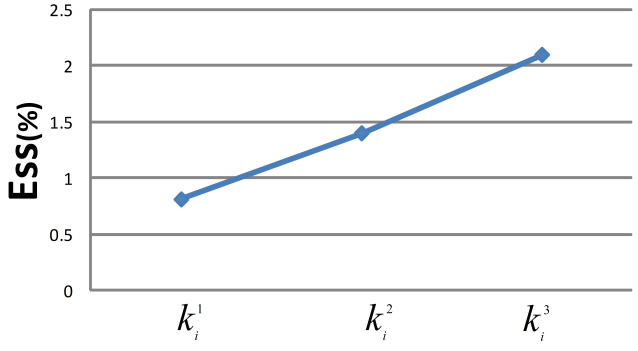


FIGURE 5.3: Setting parameters of k_p, k_i and k_d

TABLE 5.9: E_{ss} for all factors

Setting Factor	E_ss of k_p	E_ss of k_i	E_ss of k_d
1	**2.716**	**0.816**	**0.96**
2	**0.926**	**1.4**	**1.283**
3	**0.666**	**2.093**	**2.066**





5.3.5 Discussion

From Table 5.5-5.9 it can be seen that design variable k_i is minimizing M_p, t_r, t_s and E_{ss}. Observation tells us that k_i^1 is optimal value which reduces all the factor simultaneously. Whereas, K_p reduces M_p for first factor and for other settings, parameters that are t_r, t_s and E_{ss} increases. Again, same is the case with k_d does not have a great effect

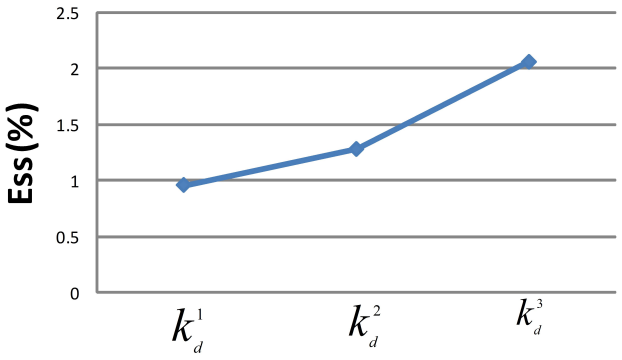


FIGURE 5.4: Setting parameters of k_p, k_i and k_d

TABLE 5.10: Effect of various factors on the dynamic response

	M_p		t_r		t_s		E_{ss}	
	SSF	**Factor Effects (%)**	**SSF**	**Factor Effects (%)**	**SSF**	**Factor Effects (%)**	**SSF**	**Factor Effects (%)**
A	**83.7**	**57.64**	**0.42**	**81.85**	**0.285**	**30.75**	**7.47**	**62.97**
B	**13.5**	**9.32**	**0.055**	**10.87**	**0.516**	**55.49**	**2.45**	**20.65**
C	**48.1**	**33.04**	**0.037**	**7.28**	**0.128**	**13.76**	**1.94**	**16.38**
Sum	**145.3**	**100**	**0.514**	**100**	**0.929**	**100**	**11.86**	**100**

on design variable. Therefore, design variable K_p and k_d needs to be optimized for better results. For optimization of these design variable BBBC optimization algorithm is used in order to obtain the optimal values. The optimal of k_i is $k_i^1 = 0.5$. Hence, as it was contemplated that number of variables will decrease, leading to reduction in computational burden and lesser usage of memory.

5.4 Optimization by BBBC

History and details of BBBC are detailed in chapter 1. BBBC has been used in various branches of engineering such as in civil engineering for optimal truss design [61], [62], skeletal or frame design [63], least-cost design of water distribution system [64], it is also used in data clustering [65], optimal economic dispatch of power [66], optimal reconfiguration and distributed generation power allocation in distribution systems [67], reactive power dispatch [68], optimal preventive control action on power systems [69], adaptive fuzzy model based inverse controller design [70], power system stabilizer design [71] in electrical engineering.

In any optimization process, we need to have an objective function. Here, we have system parameters P=[M_p t_r t_s E_{ss}] which needs to be optimized. After application of Taguchi

method, we found that $k_i = k_i^1 = 0.5$. And hence, now there are only two parameters k_p and k_d which are to be optimized such that system parameter P get minimized.

5.4.1 Creation of Objective Function

All the details of system parameters P are available and using this data and Matlab curve fitting toolbox a polynomial model with respect to M_p, t_r, t_s and E_{ss} is formed. These polynomial models are given below

$$M_p = 4.29 + 3.735k_p - 2.73k_d - 1.315k_pk_d - 0.1617k_p^2 + 1.2833k_d^2 \tag{5.7}$$

$$t_r = 0.11 - 0.2378k_p + 0.0628k_d - 0.1332k_pk_d + 0.2022k_p^2 - 0.0828k_d^2 \tag{5.8}$$

$$t_s = 0.14 - 0.2139k_p + 0.146k_d - 0.303k_pk_d - 0.0758k_p^2 - 0.0063k_d^2 \tag{5.9}$$

$$E_{ss} = 0.003 - 0.025k_p + 0.001k_d - 0.0015k_pk_d + 0.004k_p^2 + 0.001k_d^2. \tag{5.10}$$

The constraints are $0.5 \leqslant k_p \leqslant 1$ and $0 \leqslant k_d \leqslant 0.3$.

5.4.2 BBBC approach

Now, BBBC optimizatin algorithm is used to optimize the above objective function in equation 5.7-5.10. The optimal values are $k_p = 0.56$ and $k_d = 0.2$. At these optimal values, the optimized value of system parameters are $M_p = 0.006\%$, $t_r = 0.6322s$, $t_s = 0.5238s$ and $E_{ss} = 0.03\%$.

5.5 Simulation Result

This method i.e. TC-BBBC is applied here to fine tune the PID values. The fine tuned PID values obtained using TC-BBBC has to be compared with other optimum PID value obtained using PSO, GA and TC-BBBC. The optimium value obtained using GA is $k_p = 0.772$, $k_i = 0.72$ and $k_d = 0.319$. In PSO, the optimum values are $k_p = 0.67$, $k_i = 0.59$ and $k_d = 0.26$. Whereas, optimal values from TC-BBBC are $k_p = 0.56$, $k_i = 0.5$ and $k_d = 0.2$. The simulation results for the obtained optimal values are shown in From Table 5.11, it can be observed that peak overshoot of the TCBBBC is has lowest possible peak overshoot as compared to all optimum models. The rise time TCBBBC is bit higher as compared to other models. The settling time is better as compared to

TABLE 5.11: Simulation result for all optimum models

Model	$M_p(\%)$	$t_r(s)$	$t_s(s)$	$E_{ss}(\%)$
GA	**4.74**	**0.33**	**0.86**	**1**
PSO	**1.7**	**0.42**	**0.81**	**0.9**
BBBC	**1.175**	**0.3**	**0.46**	**0.48**
TC-BBBC	**0.006**	**0.63**	**0.52**	**0.03**

GA and PSO, but is slightly higher as compared to BBBC. The steady state error for TCBBBC model is the lowest among all the models.

Figure 5.5 compares the step response of change in terminal voltage of AVR system with PID controller tuned using GA, PSO, BBBC, TCBBBC

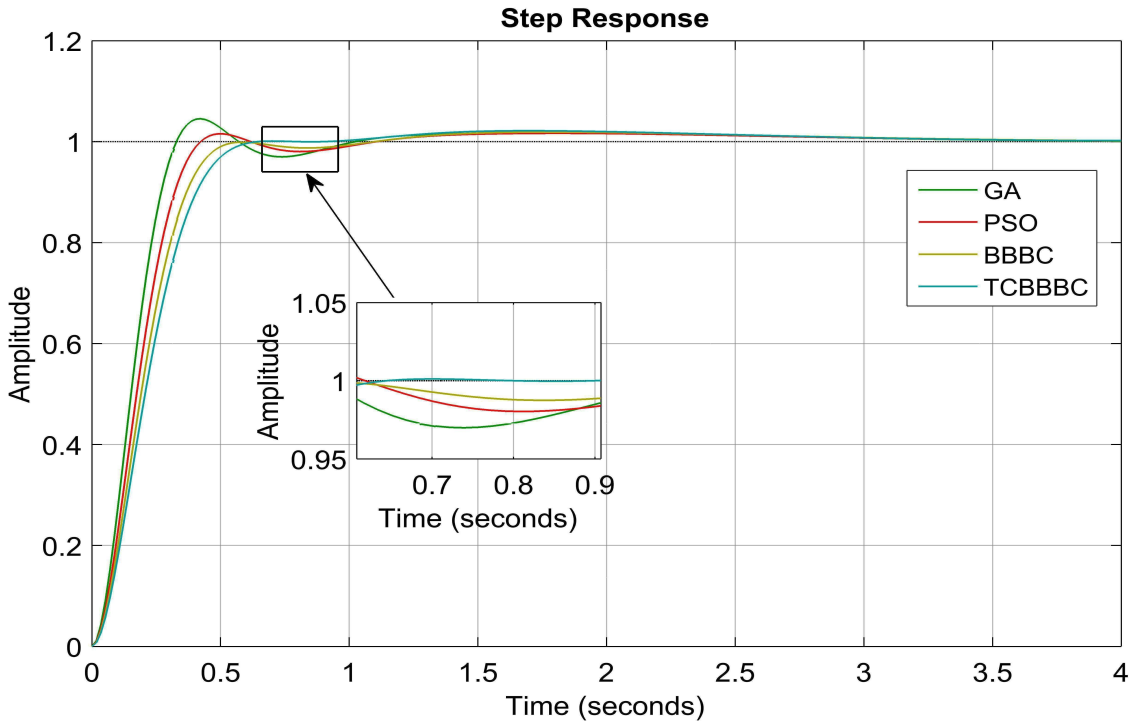


FIGURE 5.5: Change in terminal voltage of AVR with optimum models

It must be noted that M_p has been reduced by 99.64%, the t_s reduced by 35.8% and E_{ss} is also reduced by 96.6%. Therefore TCBBBC technique for controller design gives us an optimum design.

5.6 Conclusion

This chapter provides a glimpse of application of statistical technique in controller design. Here, we have combined Taguchi method and BBBC optimization for controller design in automatic voltage regulator. The Taguchi method has helped in reduction of

design variable in search space, and hence increasing computational and memory management efficiency. One more point worth noting is that this method is best suited for fine tuning already available design variables value. The values of design variable obtained using this technique is near optimal, which are better than other technique.

Chapter 6

Conclusion and Future Scope

6.1 Conclusion

The research work presented in this thesis is concerned with the detailed study of recently developed algorithm i.e. BBBC, and to demonstrate it's utility in field of System and Control engineering. This report also focuses on utility of statistical techniques like hypothesis testing and Taguchi method in Control engineering.

Our investigation into BBBC algorithm revealed various shortcomings with respect to its concept and working structure. Therefore, to mitigate these short comings various modification has been proposed into this algorithm. Along with this work, it was also found that BBBC and MBBBC, like many algorithm get stuck in local optima.And hence, this problem was overcome by introduction of chaos in BBBC and Modified BBBC. Proposed MBBBC and CMBBBC has been compared with PSO in term of there convergence rate and its effectiveness. It was found that MBBBC is far better than BBBC. CMBBBC never get stuck in local optima and has very high convergence rate as well as effectiveness.

A novel model order reduction technique has been proposed which is combination of BBBC and time moment matching method. This techniques utility is shown for various linear time invariant system of order ranging from fourth order to tenth order system and time delayed system of tenth order system. It is observed that proposed algorithm performs well in both time and frequency domain.

This dissertation also presents utility of BBBC algorithm for controller design in automatic voltage regulator system (AVR). In this work, we have formulated a multi-objective function and BBBC algorithm has been used as an optimization tool. This work has shown highly improved results as compared to other techniques.

During this work question was raised regarding the efficiency of BBBC with respect to well known algorithms like PSO and GA. Therefore, to find out concrete conclusion regarding efficiency and effectiveness of BBBC compared to PSO and GA, we have used a technique in inferential statistics i.e. hypothesis testing. And results shows that effectiveness of BBBC is on par with PSO and GA, whereas computational efficiency of BBBC is far better than GA.

During work on controller design of AVR system using BBBC algorithm, we found that there was still some scope of improvement in terms of its maximum peak overshoot and steady state error. This challenge of fine tuning the design variables was achieved by using a statistical technique i.e. Taguchi method in combination with BBBC. Observation shows an improvement in both of these parameters.

6.2 Future work

While this thesis shows a potential of MOR of linear time variant system and time delayed system using optimization technique in combination of analytical technique. Many opportunities exist to further extend the scope of this work.

Extending the proposed MOR technique for higher order high frequency switching systems.

In chapter two, the proposed technique for order reduction has shown very promising results for system with low frequency switching. But, this algorithm shows weakness in the case of high frequency switching systems. For example, the system given below equation 6.1 shows high frequency variations. Future work might be to extended proposed work for this type of systems.

$$G(s) = \frac{2s\ e^{-s}}{(s+1)^2 + 10000} \tag{6.1}$$

Extending proposed multi-objective function for AVR system controller design to more complex model (higher order) of AVR including various non-linearities.

In chapter 4, we have proposed a multi-objective function (Equation 4.5) which has shown a very promising result in case of generalized model (Figure 4.1). There are various complex models of AVR system in literature like [72], [73], [74].

Extending hypothesis testing method on various other powerful optimization technique and forming a mathematical proof regarding there effectiveness and computational efficiency.

In chapter 2, we have demonstrated a method of finding a conclusion regarding effectiveness and computational efficiency of BBBC, PSO and GA. This method can be extended to various other powerful optimization technique like ant colony optimization technique, artificial bee colony, firefly algorithm etc.. The whole elaborated list of optimization technique has been provided in [75].

Extending the work of searching various method in statistics for fine tuning design variables (PID values) in controller design.

In last chapter, we have used Taguchi method in combination with BBBC algorithm to fine tune various system parameters in AVR controller design. This work can be extended by finding out various others statistical technique to fine tune controller design variable, like various methods described in [76], [77], [78].

Extending the work of comparing PSO, GA and BBBC using ANOVA technique.

In chapter 2, we have compared the efficiency of of PSO, GA, BBBC using t-testing method. This test can also be carried out using ANOVA technique. Hence, the work of comparing the algorithms using ANOVA technique can be a next milestone.

List of Publications

1. S. Biradar, Y. V. Hote and S. Saxena, "Reduced-order modeling of linear time invariant systems using big bang big crunch optimization and time moment matching method," *Applied Mathematical Modelling, Elsevier*, 2016.

2. S. Biradar, S. Saxena and Y. V. Hote, "Simplified model identification of automatic voltage regulator using model-order reduction," *Power and Advanced Control Engineering (ICPACE), 2015 International Conference on*, Bangalore, 2015, pp. 423-428.

To Be Submitted

3. S. Biradar, Y. V. Hote, "Accelerated Modified Big Bang Big Crunch Optimization based on evolution of universe".

4. S. Biradar, Y. V. Hote, "Design Optimization of PID Controller in Automatic Voltage Regulator System Using Big Bang Big Crunch Optimization Algorithm".

5. S. Biradar, Y. V. Hote, "Comparision of PSO, GA and BBBC using Inferential Statistics".

Bibliography

[1] O. K. Erol and I. Eksin, "A new optimization method: big bang–big crunch," *Advances in Engineering Software*, vol. 37, no. 2, pp. 106–111, 2006.

[2] B. Liu, L. Wang, Y.-H. Jin, F. Tang, and D.-X. Huang, "Improved particle swarm optimization combined with chaos," *Chaos, Solitons & Fractals*, vol. 25, no. 5, pp. 1261–1271, 2005.

[3] Wikipedia, "Logistic function." https://en.wikipedia.org/wiki/Logistic_function.

[4] L. Wang, "Intelligent optimization algorithms with applications," *Tsinghua University & Springer Press, Beijing*, 2001.

[5] R. Hassan, B. Cohanim, O. De Weck, and G. Venter, "A comparison of particle swarm optimization and the genetic algorithm," in *Proceedings of the 1st AIAA multidisciplinary design optimization specialist conference*, pp. 18–21, 2005.

[6] D. E. Goldberg, *Genetic algorithms*. Pearson Education India, 2006.

[7] W. Volk, "Applied statistics for engineers.," tech. rep., 1969.

[8] V. Feoktistov, *Differential evolution*. Springer, 2006.

[9] C. A. Floudas, P. M. Pardalos, C. Adjiman, W. R. Esposito, Z. H. Gümüs, S. T. Harding, J. L. Klepeis, C. A. Meyer, and C. A. Schweiger, *Handbook of test problems in local and global optimization*, vol. 33. Springer Science & Business Media, 2013.

[10] V. Picheny, T. Wagner, and D. Ginsbourger, "A benchmark of kriging-based infill criteria for noisy optimization," *Structural and Multidisciplinary Optimization*, vol. 48, no. 3, pp. 607–626, 2013.

[11] A. R. Hedar, "Global optimization test problems," *URL http://www-optima. amp. i. kyoto-u. ac. jp/member/student/hedar/Hedar_files/TestGO. htm*, 2013.

[12] W. H. Schilders, H. A. Van der Vorst, and J. Rommes, *Model order reduction: theory, research aspects and applications*, vol. 13. Springer, 2008.

[13] G. Obinata and B. D. Anderson, *Model reduction for control system design*. Springer Science & Business Media, 2012.

[14] R. Eid, *Time domain model reduction by moment matching*. PhD thesis, Universität München, 2009.

[15] L. Fortuna, G. Nunnari, and A. Gallo, *Model order reduction techniques with applications in electrical engineering*. Springer Science & Business Media, 2012.

[16] P. Benner, V. Mehrmann, and D. C. Sorensen, *Dimension reduction of large-scale systems*, vol. 45. Springer, 2005.

[17] R. Hassan, B. Cohanim, O. De Weck, and G. Venter, "A comparison of particle swarm optimization and the genetic algorithm," in *Proceedings of the 1st AIAA multidisciplinary design optimization specialist conference*, pp. 1–13, 2005.

[18] S. Desai and R. Prasad, "A novel order diminution of linear time invariant systems using big bang big crunch optimization and routh approximation," *Applied Mathematical Modelling*, vol. 37, no. 16, pp. 8016–8028, 2013.

[19] V. Zakian, "Simplification of linear time-invariant systems by moment approximants," *International Journal of Control*, vol. 18, no. 3, pp. 455–460, 1973.

[20] A. Sikander and R. Prasad, "Linear time-invariant system reduction using a mixed methods approach," *Applied Mathematical Modelling*, vol. 39, no. 16, pp. 4848–4858, 2015.

[21] Y. Smamash, "Truncation method of reduction: a viable alternative," *Electronics Letters*, vol. 17, no. 2, pp. 97–99, 1981.

[22] V. Krishnamurthy and V. Seshadri, "Model reduction using the routh stability criterion," *IEEE Transactions on Automatic Control,*, vol. 23, no. 4, pp. 729–731, 1978.

[23] T. Chen, C. Chang, and K. Han, "Stable reduced-order padé approximants using stability-equation method," *Electronics Letters*, vol. 16, no. 9, pp. 345–346, 1980.

[24] P. Kokotovic, H. K. Khalil, and J. O'reilly, *Singular perturbation methods in control: analysis and design*, vol. 25. SIAM, 1999.

[25] B. Moor, "Principle component analysis in linear system," *IEEE Trans Autom Control*, vol. 11, pp. 17–32, 1981.

[26] G. Parmar, R. Prasad, and S. Mukherjee, "Order reduction of linear dynamic systems using stability equation method and genetic algorithm," *International Journal of Computer, Information, and Systems Science, and Engineering*, vol. 1, no. 1, pp. 26–32, 2007.

[27] T. Chen, C. Chang, and K. Han, "Stable reduced-order padé approximants using stability-equation method," *Electronics Letters*, vol. 16, no. 9, pp. 345–346, 1980.

[28] J. Pal, "Improved padé approximants using stability equation method," *Electronics Letters*, vol. 19, no. 11, pp. 426–427, 1983.

[29] R. Parthasarathy and K. Jayasimha, "System reduction using stability-equation method and modified cauer continued fraction," *Proceedings of the IEEE*, vol. 70, no. 10, pp. 1234–1236, 1982.

[30] E. J. Davison, "A method for simplifying linear dynamic systems," *IEEE Transactions on Automatic Control,*, vol. 11, no. 1, pp. 93–101, 1966.

[31] L. Shieh and Y. Wei, "A mixed method for multivariable system reduction," *IEEE Transactions on Automatic Control,*, vol. 20, no. 3, pp. 429–432, 1975.

[32] M. Safonov and R. Chiang, "Model reduction for robust control: A schur relative error method," *International Journal of Adaptive Control and Signal Processing*, vol. 2, no. 4, pp. 259–272, 1988.

[33] D. Abu-Al-Nadi, O. M. Alsmadi, and Z. S. Abo-Hammour, "Reduced order modeling of linear mimo systems using particle swarm optimization," in *7th International Conference on Autonomic and Autonomous Science*, vol. 7077, pp. 278–286, 2011.

[34] A. Ula and A. R. Hasan, "Design and implementation of a personal computer based automatic voltage regulator for a synchronous generator," *IEEE Transactions on Energy Conversion,*, vol. 7, no. 1, pp. 125–131, 1992.

[35] K. J. Åström and T. Hägglund, "The future of pid control," *Control engineering practice*, vol. 9, no. 11, pp. 1163–1175, 2001.

[36] A. Visioli, "Tuning of pid controllers with fuzzy logic," *IEE Proceedings-Control Theory and Applications*, vol. 148, no. 1, pp. 1–8, 2001.

[37] Y. Li, K. H. Ang, *et al.*, "PID control system analysis and design," *IEEE Control Systems,*, vol. 26, no. 1, pp. 32–41, 2006.

[38] S.-R. Qi, D.-F. Wang, P. Han, W. Li, *et al.*, "Grey prediction based RBF neural network self-tuning pid control for turning process," in *Proceedings of 2004 International Conference on Machine Learning and Cybernetics.*, vol. 2, pp. 802–805, 2004.

[39] A. Rubaai, J. Castro-Sitiriche, and A. R. Ofoli, "Design and implementation of parallel fuzzy PID controller for high-performance brushless motor drives: an integrated environment for rapid control prototyping," *IEEE Transactions on Industry Applications,*, vol. 44, no. 4, pp. 1090–1098, 2008.

[40] A. Rubaai and P. Young, "EKF-based PI-PD-like fuzzy-neural-network controller for brushless drives," *IEEE Transactions on Industry Applications,*, vol. 47, no. 6, pp. 2391–2401, 2011.

[41] B. Lennartson and B. Kristiansson, "Evaluation and tuning of robust PID controllers," *IET Control Theory & Applications,*, vol. 3, no. 3, pp. 294–302, 2009.

[42] X. H. Li, H. Yu, M. Yuan, and J. Wang, "Design of robust optimal proportional–integral–derivative controller based on new interval polynomial stability criterion and lyapunov theorem in the multiple parameters' perturbations circumstance," *IET Control Theory & Applications,*, vol. 4, no. 11, pp. 2427–2440, 2010.

[43] N. Tan, I. Kaya, C. Yeroglu, and D. P. Atherton, "Computation of stabilizing PI and PID controllers using the stability boundary locus," *Energy Conversion and Management*, vol. 47, no. 18, pp. 3045–3058, 2006.

[44] R. Krohling, J. P. Rey, *et al.*, "Design of optimal disturbance rejection PID controllers using genetic algorithm," *IEEE Transactions on Evolutionary Computation,*, vol. 5, no. 1, pp. 78–82, 2001.

[45] D. Devaraj and B. Selvabala, "Real-coded genetic algorithm and fuzzy logic approach for real-time tuning of proportional-integral-derivative controller in automatic voltage regulator system," *IET Generation, Transmission & Distribution,*, vol. 3, no. 7, pp. 641–649, 2009.

[46] J. Zhang, J. Zhuang, H. Du, *et al.*, "Self-organizing genetic algorithm based tuning of PID controllers," *Information Sciences*, vol. 179, no. 7, pp. 1007–1018, 2009.

[47] Z.-L. Gaing, "A particle swarm optimization approach for optimum design of PID controller in AVR system," *IEEE Transactions on Energy Conversion,*, vol. 19, no. 2, pp. 384–391, 2004.

[48] V. Mukherjee and S. Ghoshal, "Intelligent particle swarm optimized fuzzy PID controller for AVR system," *Electric Power Systems Research*, vol. 77, no. 12, pp. 1689–1698, 2007.

[49] J.-S. Chiou and M.-T. Liu, "Numerical simulation for fuzzy-PID controllers and helping EP reproduction with PSO hybrid algorithm," *Simulation Modelling Practice and Theory*, vol. 17, no. 10, pp. 1555–1565, 2009.

[50] H. M. Hasanien, "Design optimization of PID controller in automatic voltage regulator system using Taguchi combined genetic algorithm method," *IEEE Systems Journal,*, vol. 7, no. 4, pp. 825–831, 2013.

[51] A. M. Omekanda, "Robust torque and torque-per-inertia optimization of a switched reluctance motor using the Taguchi methods," *IEEE Transactions on Industry Applications,*, vol. 42, no. 2, pp. 473–478, 2006.

[52] K. J. Aström and R. M. Murray, *Feedback systems: an introduction for scientists and engineers*. Princeton university press, 2010.

[53] A. M. Omekanda, "Robust torque and torque-per-inertia optimization of a switched reluctance motor using the Taguchi methods," *IEEE Transactions on Industry Applications,*, vol. 42, no. 2, pp. 473–478, 2006.

[54] C.-C. Hwang, L.-Y. Lyu, C.-T. Liu, and P.-L. Li, "Optimal design of an SPM motor using genetic algorithms and Taguchi method," *IEEE Transactions on Magnetics,*, vol. 44, no. 11, pp. 4325–4328, 2008.

[55] M. Ashabani, Y. A.-R. I. Mohamed, and J. Milimonfared, "Optimum design of tubular permanent-magnet motors for thrust characteristics improvement by combined Taguchi–neural network approach," *IEEE Transactions on Magnetics,*, vol. 46, no. 12, pp. 4092–4100, 2010.

[56] F. D. Zahlay, K. R. Rao, and T. B. Ibrahim, "A new intelligent autoreclosing scheme using artificial neural network and Taguchi's methodology," in *Industrial*

and Commercial Power Systems Technical Conference (I&CPS), 2010 IEEE, pp. 1–8, IEEE, 2010.

[57] Y.-H. Liu and Y.-F. Luo, "Search for an optimal rapid-charging pattern for Li-ion batteries using the Taguchi approach," *IEEE Transactions on Industrial Electronics,*, vol. 57, no. 12, pp. 3963–3971, 2010.

[58] G. Taguchi, Y. Yokoyama, *et al.*, *Taguchi methods: design of experiments*, vol. 4. Amer Supplier Inst, 1993.

[59] R. K. Roy, *A primer on the Taguchi method*. Society of Manufacturing Engineers, 2010.

[60] G. Taguchi and G. Taguchi, "System of experimental design; engineering methods to optimize quality and minimize costs," tech. rep., 1987.

[61] "Introduction to taguchi method." http://www.ecs.umass.edu/mie/labs/mda/fea/sankar/chap2.html. Accessed: 2016-05-18.

[62] A. Kaveh and S. Talatahari, "Size optimization of space trusses using Big Bang–Big Crunch algorithm," *Computers & structures*, vol. 87, no. 17, pp. 1129–1140, 2009.

[63] C. V. Camp, "Design of space trusses using Big Bang–Big Crunch optimization," *Journal of Structural Engineering*, vol. 133, no. 7, pp. 999–1008, 2007.

[64] A. Kaveh and S. Talatahari, "A discrete big bang-big crunch algorithm for optimal design of skeletal structures," *Asian Journal of Civil Engineering*, vol. 11, no. 1, pp. 103–122, 2010.

[65] A. Tahershamsi, A. Kaveh, R. Sheikholeslami, and S. Talatahari, "Big bang–big crunch algorithm for least-cost design of water distribution systems," 2012.

[66] A. Hatamlou, S. Abdullah, and M. Hatamlou, "Data clustering using big bang–big crunch algorithm," in *Innovative Computing Technology*, pp. 383–388, Springer, 2011.

[67] Y. Labbi and D. Attous, "Big bang-big crunch optimization algorithm for economic dispatch with valve-point effect.," *Journal of Theoretical & Applied Information Technology*, vol. 16, 2010.

[68] M. Sedighizadeh, M. Esmaili, and M. Esmaeili, "Application of the hybrid Big Bang-Big Crunch algorithm to optimal reconfiguration and distributed generation power allocation in distribution systems," *Energy*, vol. 76, pp. 920–930, 2014.

[69] Z. Zandi, E. Afjei, and M. Sedighizadeh, "Reactive power dispatch using big bang-big crunch optimization algorithm for voltage stability enhancement," in *IEEE International Conference on Power and Energy (PECon)*, pp. 239–244, IEEE, 2012.

[70] C. F. Kucuktezcan and V. M. I. Genc, "Big bang-big crunch based optimal preventive control action on power systems," in *3rd IEEE PES International Conference and Exhibition on Innovative Smart Grid Technologies (ISGT Europe)*, pp. 1–4, IEEE, 2012.

[71] T. Kumbasar, I. Eksin, M. Guzelkaya, and E. Yesil, "Adaptive fuzzy model based inverse controller design using BB-BC optimization algorithm," *Expert Systems with Applications*, vol. 38, no. 10, pp. 12356–12364, 2011.

[72] E. Dincel and V. Gene, "A power system stabilizer design by big bang-big crunch algorithm," in *IEEE International Conference on Control System, Computing and Engineering (ICCSCE)*, pp. 307–312, IEEE, 2012.

[73] M. Zamani, M. Karimi-Ghartemani, N. Sadati, and M. Parniani, "Design of a fractional order PID controller for an AVR using particle swarm optimization," *Control Engineering Practice*, vol. 17, no. 12, pp. 1380–1387, 2009.

[74] W.-D. Chang and S.-P. Shih, "PID controller design of nonlinear systems using an improved particle swarm optimization approach," *Communications in Nonlinear Science and Numerical Simulation*, vol. 15, no. 11, pp. 3632–3639, 2010.

[75] K. Law, D. Hill, and N. Godfrey, "Robust controller structure for coordinated power system voltage regulator and stabilizer design," *IEEE Transactions on Control Systems Technology,*, vol. 2, no. 3, pp. 220–232, 1994.

[76] I. Fister Jr, X.-S. Yang, I. Fister, J. Brest, and D. Fister, "A brief review of nature-inspired algorithms for optimization," *arXiv preprint arXiv:1307.4186*, 2013.

[77] T. Harris, C. Seppala, and L. Desborough, "A review of performance monitoring and assessment techniques for univariate and multivariate control systems," *Journal of Process Control*, vol. 9, no. 1, pp. 1–17, 1999.

[78] J. Yu and S. J. Qin, "Statistical mimo controller performance monitoring. part i: Data-driven covariance benchmark," *Journal of Process Control*, vol. 18, no. 3, pp. 277–296, 2008.

[79] G. Box and T. Kramer, "Statistical process monitoring and feedback adjustmenta discussion," *Technometrics*, vol. 34, no. 3, pp. 251–267, 1992.