

STABLE REDUCED ORDER MODELS FOR DISCRETE TIME SYSTEMS

A DISSERTATION

*Submitted in partial fulfillment of the
requirements for the award of the degree*

of

MASTER OF TECHNOLOGY

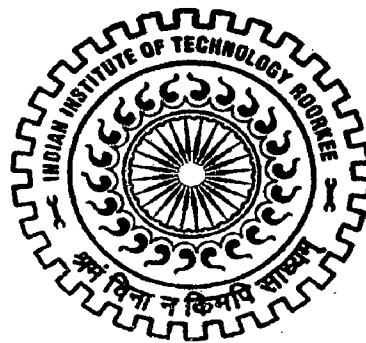
in

ELECTRICAL ENGINEERING

(With Specialization in System Engineering and Operations Research)

By

CH. SREENIVASA GUPTA



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

JUNE, 2008

CANDIDATE'S DECLARATION

I hereby declare that the work which is being presented in this dissertation entitled, "**STABLE REDUCED ORDER MODELS FOR DISCRETE TIME SYSTEM**" submitted in the partial fulfillment of the requirements for the award of the degree "**MASTER OF TECHNOLOGY**" with specialization in **SYSTEM ENGINEERING AND OPERATIONS RESEARCH**, in the department of **ELECTRICAL ENGINEERING, IIT Roorkee**, Roorkee is an authentic record of my own work carried out during the period from August 2007 to June 2008 under the supervision of **Dr. RAJENDRA PRASAD, Associative Professor** Department of Electrical Engineering, IIT Roorkee, Roorkee.


The matters embodied in this report have not been submitted by me for the award of any other degree or diploma.

Date: 30 June 2008

Place: Roorkee


(CH.SREENIVASA GUPTA)

This is to certify that above statement made by candidates is correct to the best of my knowledge.


(Dr. RAJENDRA PRASAD) 30/6/08
Associative Professor
Electrical Engg. Department
IIT Roorkee

ACKNOWLEDGEMENT

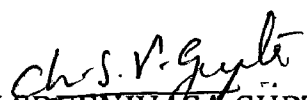
My foremost and profound gratitude goes to my guide **Dr. Rajendra Prasad, Associative Professor, Electrical Engineering, IIT Roorkee** for their proficient and enthusiastic guidance, useful encouragement and immense help. I have been deep sense of admiration for them innate goodness and inexhaustible enthusiasm.

I wish to extend my sincere gratitude to **Dr. Indra Gupta, Associative Professor** for allowing me to work in Computer & Micro processors Lab even in odd hours of the day.

My heartfelt gratitude and indebtedness goes to all teachers of SEOR group who with their encouraging ,caring words, constructive criticism and segmentation have contribute directly or indirectly in a significant way towards completion of this report.

I also express sincere thanks to all staff of Computer & Microprocessors Lab for their help in the completion of this work and special thanks to my friends whose support and encouragement has been a constant source of assurance, guidance, strength, inspiration to me.

Date: June, 2008


(CH.SREENIVASA GUPTA)

ABSTRACT

The work presented in the dissertation deals with algorithms for model reduction of single input single output discrete time systems in z-transfer function and comparison of results among algorithms for best suitable stable reduced model of given higher order system.

Reduction of higher order system transfer function to low order models has been an important area in the control engineering environment for many years. In the model reduction, finding the low order model of given higher order system transfer function, which reflects the dominant characteristics of original system, normally step responses matching are performed

There are several different approaches for the reduction of discrete time systems. Here three different such type of algorithms are discussed, namely Markov and H-parameter matching (MHM), Genetic Algorithm (direct search method), and combination of above both algorithms.

First algorithm requires error function in terms of reduced model variables, Pascal triangle. Markov parameters will dictate transient response and h-parameters will dictate the later part of response including steady state response. By making use of these variables we find out the reduced model parameters. Second algorithm is direct search optimization technique. It directs searching for reduced model variables in space in proper steps and also avoids local minima. Third algorithm is combination of both first and second algorithm. It has the advantages of both algorithms and these are explained in detail in chapter 5.

The advantages of the proposed algorithms are that characteristics of original system can be preserved in reduced models and in the reduced models are always stable provided, the original system stable. In each algorithm, general formulation, steps involved and flow charts for reducing n^{th} order system to r^{th} order are explained with numerical example also.

CONTENTS

	Page no
CANDIDATE'S DECLARATION	i
ACKNOWLEDGEMENT	ii
ABSTRACT	iii
Chapter 1: Introduction	1
1.1 General Introduction	1
1.2 Literature Review	3
1.3 Statement of problem	4
1.4 Dissertation layout	5
Chapter 2: Model Reductions	6
2.1. Introduction	6
2.2. Need for model reduction	7
2.3. Optimal model reduction	8
2.4. Different methods of reductions	8
2.4.1. Classification	9
2.4.2. Time domain Specification techniques	9
2.4.3. Frequency domain Specification techniques	9
2.5. Needs for reduction of discrete time systems	10
2.5.1. Reduction by using direct methods	10
2.5.2. Reduction by using indirect methods	10
Chapter 3: Genetic algorithms	11
3.1. Introduction	11
3.2. Methods of representation	13
3.3. Methods of approach	14
3.4. Strengths of Genetic algorithms	19
3.5. Limitations	20
Chapter 4: Implementations of Reduction Algorithms	21
4.1 Introduction	21
4.2. Statement for problem	21
4.3 Markov and H-parameters	22
4.4. Error polynomial and its implications	22
4.5. Theorems	25
4.5.1. Theorem 1	25

4.5.2. Theorem 2	26
4.6. Algorithms	28
4.6.1. Algorithm 1	28
4.6.2. Algorithm2	30
4.6.3. Algorithm 3	31
4.7. Numerical example	35
Chapter 5: Results	40
5.1. Problem 1	41
5.2. Problem 2	46
5.3. Comparisons of reduced models with given system	52
5.3.1. Problem 1	52
5.3.2. Problem 2	54
Chapter 6: Conclusion and future scope	57
6.1. Conclusion	57
6.2. Future scope	58
References	59

Chapter 1

INTRODUCTION

1.1 GENERAL INTRODUCTION

The approximation of high-order systems by low-order models is one of the important problems in system theory. The use of a reduced order model makes it easier to implement analyses, simulations and control designs. Model-reduction problems have received considerable attention for many years and have been approached in a number of ways [1-10]. Many control techniques are relatively simple to implement on a low-order system containing few parameters. High-order systems, however, generally lead to a much greater amount of effort for their analysis, especially in terms of necessary computation.

One scheme to ameliorate this is to formulate a model which is of much lower order than the system, and yet which retains certain of its properties such that it may be viewed as a reasonable approximation to the system. In the analysis of many systems for which physical laws are relatively well known, one is frequently confronted by problems arising from the high dimensions of descriptive state model, the famous curse of dimensionality. This is particularly true for the systems described by partial differential equations, for which discretization in space is generally used to obtain a system of ordinary differential equations, simpler to use but of a very high dimension.

A continuous-time transfer function can be described in terms of its Markov parameters or its time series proportional. If the Markov parameters of a low-order model are made equal to those of the system in as many cases as the model will allow, the model response to a step input will approximate that of the system for a short time period after the step change has occurred. In the same way, if time series proportional to both model and system are equated in as many cases as possible, the model will approximate as steady-state is encroached upon. One technique for obtaining an overall approximation to a high order system is to use two low order models [1]. One to cover the initial response and one for steady-state conditions.

More recently, however effort has been concentrated on finding one single low-order model to approximate. The system [2] based simply on a matching of model and system time series proportionals i.e. Pade approximation,

but more generally the model denominator can be found by means of the Routh stability criterion such that stability problems associated with Pade approximation are overcome, The model numerator parameters can then be found via Pade approximation [3, 4], although this means that fewer time series proportionals can be matched. Models can also be found whose parameters are used to match a few Markov parameters and a few time series proportionals [5, 6, and 7]. This can also mean that the error between certain remaining parameters can be minimized given sufficient degrees of freedom [8].

A review of many of these methods was given in Ashoor and Singh [9], where the performance of the schemes on several transfer functions was considered. It is apparent, however that the methods referenced involve detailed ladder networks which are computationally time consuming and prone to rounding errors. It is believed that this approach is computationally efficient and simple to comprehend. The details are described in terms of discrete time systems, due to the direct applicability of the models obtained, and follow on from previous discrete time modeling [10], involving continued fractions and, hence, ladder networks.

The approach employs what is termed the error polynomial, which is found as the difference between the system and model outputs, to an identical step input. In Section 2 the transfer function and its reduced-order model are defined, both being assumed to consist of relatively prime polynomials. The error polynomial is then detailed in Section 3, along with the effective error input for any particular step input change. The parameters contained within the error polynomial are considered in Section 4, where two theorems are introduced to show how Markov and time series proportional parameter matching can be achieved simply by equating the error polynomial coefficients with zero. Genetic algorithm using direct search method[25] for reducing higher order model also discussed here and compares this with markov parameter and h-parameter equalization algorithm. Combination of both also tested and it yields to the good results which are discussed in chapter 5.

However, the one at a time search method described above is not so computationally efficient, especially for discrete-time systems, and in the gradient based method the problem of local optima cannot be avoided, as mentioned below. Since the minimization of the many conventional quadratic cost function is a nonlinear problem with respect to the denominator parameters. A nonlinear optimization technique [18], [20] such as the gradient-based method is usually required.

However, such techniques are often very complicated and computationally demanding. Moreover, since the cost function generally has multiple local minima, the attainment of the global optimum by the nonlinear optimization techniques is difficult. In this thesis, to overcome these problems, we propose a novel L2, model-reduction algorithm [19] for single input, single output (SISO) discrete-time systems combining the least-squares (LS) into with the genetic algorithm combined with conventional method. The GA is a probabilistic search procedure based on the mechanics of natural selection and natural genetics [24]. Recently the GA has received Considerable attention in various fields [29 -30], because It has a high potential for global optimization.

1.2. LITERATUR REVIEW

Reduction of higher order systems transfer function to lower order model has been an important subject area in control engineering environments for many years. The problems in large scale systems including reduced order modeling have been a favorite area of researchers during past two decades. Reduced modeling of discrete time systems have been attempted in both in time and frequency.

Westcott, J.H. [1] are proposed a method for reduction of large scale system to lower order model in frequency domain and explained about their relation ship with transient behavior of system. Shamash.Y [2],[4],[7],[10], has done excellent work in the area of deriving stable reduction models by several method such as Routh stability criterion and Pade approximations and also proposed how to identify and estimate the system parameters. Hutton, M. F., and Friedland.B [3] proposed a method of model reductions by using the Routh approximations. Chen, T. C., Chang, C. Y., and Han, K.W [5] has proposed a new method of continuous fraction along with stability equation. Pal.J used Routh-Hurwitz array to found stable reduced approximates [8], [9].

Principal component analysis in linear time systems provided by Moore, B.C.[11] and also proposed the effects of controllability, observability on model reduction. Warwick proposed a new method of finding the stable reduced order model with matching markov and h-parameters of model and system [12].All the methods specified above also deals with stability of reduced model by routh stability array, nyquist stability criteria. Excellent books [13], [16] have detailed contributions in the reduction of higher order system to stable reduced order models by using different methods.

Skelton, R.E., and Anderson [15] are extend the work of Warwick [12] in the area of markov parameters with covariance equalent realizations.[17],[21] are entered into new dimension of introducing delay system in reducing the large scale models .[18],[19],[20],[23]are introduced new methods like extended complex curve fitting methods ,optimal and sub optimal model reduction methods.[22] proposed the hybrid method of model reduction method along with time delays.

Goldberg [24] had written excellent book to refer Genetic algorithms and also done some research in this area. [25], [28] are proposed the how to identify system parameters and how to reduce the higher order system by using genetic algorithms. It has also played roles in tuning the various controller explained in[26],[27] Lansberry, J.E., Wozniak.E, and Goldberg [27] introduce genetic algorithms to tune a hydro generator governor. Porter.B, Jones [28] used genetic algorithms in tuning the PID controller. Yang, Z.J., Hachino.T, Tsuji.T [29], [30] are extended the work proposed in [17], [21], [25] combining Genetic algorithms with conventional methods and also time delay necessity present in reduced models.

1.3. STATEMENT OF PROBLEM

The objective of the dissertation is to compare the time domain methods for model reduction of discrete time systems in z-domain and to discuss time domain approaches to derive stable reduced order model for a stable discrete time system.

In this dissertation order of discrete time system will be carried out by three methods namely Markov and H-parameters Matching (MHM), Genetic algorithms and Genetic algorithms with Markov and H-parameter matching (combination of both above algorithms).

In first algorithm, reduced model parameter variables can be found by matching markov parameters and h- parameters of given system and reduced model. In the second algorithm, reduced model parameter variables can be found by applying genetics to variables and searching in a specific manner. In algorithm 3, denominator parameters are found by alogrthm1 and numerator variables are found by genetic algorithm.

A numerical example will be included to illustrate these methods.

1.4. DISSERTATION LAYOUT

This dissertation consists of 6 chapters deals with details as given below.

Chapter-1 deals with brief introduction of model reduction and genetic algorithms along with literature review and objective of dissertation.

Chapter-2 deals with details about model reductions, its need and method of order reduction techniques.

Chapter-3 deals with details of genetic algorithms, how to handle with it and merits and demerits.

Chapter-4 deals with model reduction method and it's algorithms for solving along with flow charts in three different methods.

Chapter-5 deals with numerical examples, along with result and discussion are given this chapter to compare the responses of these methods explained in chapter 4

Chapter-6 deals with contribution made in this dissertation and future scope of work in this are given.

Chapter 2

MODEL REDUCTIONS

2.1. INTRODUCTION

The large Scale systems are all around and exist in diverse fields such as complex, chemical process, biomedical systems, social Economic systems transportation systems, ecological systems, social economic systems, electrical power systems, Aeronautics ,hydraulic pneumatic .

A system is said to be large if it can decoupled or partitioned into number of interconnected systems or small scale systems for either computational or practical reasons. Alternatively a system is large scale when its dimensions are too high, such that conventional techniques of modeling, analysis control, design and computation fails to give accurate solutions with reasonable computational fail to give accurate solutions with reasonable computational efforts.

The analyses of such physical systems start by building of a model which may be considered a faithful representation of such systems. The task of a control engineer begins with formulation of a model. The rest of analysis and design can be done with this model. In many practical situations a fairly complex and high order model is obtained from theoretical considerations.

The mathematical models of high order dynamic systems can be described either in state space form or in transfer function from which are called time domain and frequency domain representations. In the state space or time domain representation a high order differential equation is decoupled into a set of first order differential equations. Similarly in the transfer function or frequency domain representation, the Laplace transforms of high order differential equation is taken under zero initial conditionals and ,the mathematical model is represented as a rational function(in the ratio of Laplace transform of output to the Laplace transform input), called system transfer function. The exact analysis of most of high order systems is both tedious and costly; it poses a great challenge to both system analyst and control engineer.

The preliminary design and optimization of such system can often be accomplished with greater ease if a low order linear model is derived which provides a good approximation to the system. Desirable features of such model will be simplicity while preserving features of interest. Since the models may be

developed with various aims in mind/ view points, it is possible to have more than one model for a given system each satisfying some objective.

2.2. NEED FOR MODEL REDUCTION

Every physical system can be translated into mathematical model. The mathematical procedure of system modeling often leads to comprehensive description of a process in the form of high order differential equations which are very difficult to use either for analysis or controller synthesis. It is hence useful, and sometimes necessary, to find the possibility of finding some equation of the same type but of lower order that may be considered to adequately reflect the dominant characteristics of the system under consideration.

- **To have a better understanding of the system**

A system of uncomfortably high order poses difficulties in its analysis, synthesis or identification. An obvious method of dealing with such type of system is to approximate it by a low order system which reflect the characteristics of original system such as time constant, damping ratio, natural frequency etc.

- **To reduce Computational Complexity**

The developing of state space methods and optimal control techniques has made the design of control system for high order multivariable system quite feasible. When the order of the system becomes too high, special numerical techniques are required to permit the calculation to be done at feasible cost on fast digital computers. This saves both time and memory required by computer.

- **To reduce Hardware complexity**

A control system design for a high order system is likely to be very complicated and a high order itself. This is particularly true for controller based on optimal control theory. Controller are designed on the basis of low order model will be more reliable, less costly and easily implement and maintain.

- **To generalize results established on a particular system to comparable system.**

The results studied for a simple model can be easily generalized to other comparable system.

- **To make feasible designs**

Reduced order models, may be effectively used in special situation like,

- a) Model reference adaptive control schemes.
- b) Hierarchical control schemes.
- c) Suboptimal control.
- d) Decentralized controllers

- **To improve the methodology of computer aided control system design**

The methodology of computer aided techniques for control system design can be easily improved using simpler models.

2.3. OPTIMAL MODEL REDUCTION

The reduction methods are based on obtaining a model of specified order such that its impulse or step response matches that of original system in an optimal manner, with no restriction on the location of the Eigen values. Such techniques aim at minimizing a selected performance criterion, which is a general function of error between the response of original high order system and its reduced model system. The parameters of reduced model are obtain either from necessary conditions of optimality or from the numerical algorithm.

2.4. DIFFERENT METHODS OF REDUCTION

In the field of model reduction several methods has been proposed. The basic aim of model reduction is that reduced order approximant should reproduce the significant characteristics of the present system as closely as possible. There are wide verity of concepts and techniques which have a common goal of reducing the dimensions of mathematical model of large scale systems to facilitate their analysis and design or simulation. The model order reduction can be achieved either in the time domain or in the frequency domain or combination of both .an excellent review of the methods developed in the area is available in various references. We will use the term system to represent the original higher order system and the model for its reduced order system.

2.4.1. CLASSIFICATION

The order reduction techniques can broadly be classified as

- a) Time domain simplification techniques.
- b) Frequency domain simplification techniques.

2.4.2. TIME DOMAIN SIMPLIFICATION TECHNIQUES

In these domain techniques, the original and reduced systems are expressed in state space form. The domain techniques belong to either of the following categories.

- 1) Model analysis approach.
- 2) Subspace projection methods.
- 3) Optimal order reduction.
- 4) Hankel-norm model reduction.

2.4.3. FREQUENCY DOMAIN SIMPLIFICATION TECHNIQUES

In the frequency domain techniques, we simply reduce the order of the transfer function. Decreasing the order of the transfer function doesn't ensure the resulting transfer function is realizable. For this purpose, the term simplification is used. The frequency domain techniques belong to either of the following categories.

- 1) Continued fraction expansion and truncation [5].
- 2) Time moment matching [12], [15].
- 3) Pade approximation.
- 4) Stability criteria based reduction methods.
 - i) Routh approximations [3], [4], [7].
 - ii) Hurwitz polynomial approximation.
 - iii) Routh-Hurwitz array method [2], [6].
 - iv) Stability equation method
- 5) Polynomial differentiation.
- 6) Truncation method.
- 7) Dominant pole retention.
- 8) Simplification using Schwarz canonical form.
- 9) Reduction using minimal realization.
- 10) Frequency response matching.
- 11) Reduction using optimization [17], [24, 30].
- 12) Reduction using factor division [20].
- 13) Model reduction using Chebyshev polynomials [18].

2.5. METHODS FOR REDUCTION OF DISCRETE TIME SYSTEMS

For discrete time systems also the same arguments as for continuous systems hold as far as need for reduced order modeling is concern. Moreover the fast development and usage of small digital computers and processors in the design and implementation of control systems have increased importance of reduced order modeling methods for discrete systems.

These are broadly classified into two types.

- 1) Direct methods.
- 2) Indirect methods.

2.5.1. REDUCTION BY USING DIRECT METHODS

The bilinear transformation has been made in the indirect methods to extend model reduction techniques for continuous time systems. These methods suffer from the inherent drawbacks associated with bilinear transformation cannot only prove to be a very tedious operation it may as we produce erroneous results as some of the points in z -domain are not necessarily defined in the ' w ' domain. The reduction of discrete time systems by direct methods has been attempted by several authors [2], [10], [12], [19].

2.5.2. REDUCUTION BY USING INDIRECT METHODS

Bilinear transformations can be used to extend continued fraction method [5],[10] for continuous systems to reduce transfer function in ' w ' domain similarly many other methods for continuous time systems have been extended to reduce discrete time systems by using bilinear transformations[3],[4],[6],[7].these methods suffer from drawback that due to the nature of bilinear transformation, the initial valued step response of reduced model may not be zero.

Chapter 3

GENETIC ALGORITHM

3.1. INTRODUCTION

Genetic algorithm (GA) is a programming technique that mimics biological evolution as a problem-solving strategy. Given a specific problem to solve, the input to the GA is a set of potential solutions to that problem, encoded in some fashion, and a metric called a fitness function that allows each candidate to be quantitatively evaluated. These candidates may be solutions already known to work, with the aim of the GA being to improve them, but more often they are generated at random. Genetic algorithm is a search technique used in computing to find exact or approximate solutions to optimization and search problems.

Genetic algorithms are categorized as global search heuristics. Genetic algorithms are a particular class of evolutionary algorithms (also known as evolutionary computation) that use techniques inspired by evolutionary biology such as inheritance, mutation, selection, and crossover (also called recombination). GA then evaluates each candidate according to the fitness function. In a pool of randomly generated candidates, of course, most will not work at all, and these will be deleted. However, purely by chance, a few may hold promise - they may show activity, even if only weak and imperfect activity, toward solving the problem. These promising candidates are kept and allowed to reproduce. Multiple copies are made of them, but the copies are not perfect; random changes are introduced during the copying process.

These digital offspring then go on to the next generation, forming a new pool of candidate solutions, and are subjected to a second round of fitness evaluation. Those candidate solutions which were worsened, or made no better, by the changes to their code are again deleted; but again, purely by chance, the random variations introduced into the population may have improved some individuals, making them into better, more complete or more efficient solutions to the problem at hand. Again these winning individuals are selected and copied over into the next generation with random changes, and the process repeats.

The expectation is that the average fitness of the population will increase each round, and so by repeating this process for hundreds or thousands of rounds, very good solutions to the problem can be discovered. As astonishing

and counterintuitive as it may seem to some, genetic algorithms have proven to be an enormously powerful and successful problem-solving strategy, dramatically demonstrating the power of evolutionary principles. Genetic algorithms have been used in a wide variety of fields to evolve solutions to problems as difficult as or more difficult than those faced by human designers. Moreover, the solutions they come up with are often more efficient, more elegant, or more complex than anything comparable a human engineer would produce.

Genetic algorithms are implemented as a computer simulation in which a population of abstract representations (called chromosomes or the genotype or the genome) of candidate solutions (called individuals, creatures, or phenotypes) to an optimization problem evolves toward better solutions. Traditionally, solutions are represented in binary as strings of 0s and 1s, but other encodings are also possible.

The evolution usually starts from a population of randomly generated individuals and happens in generations. In each generation, the fitness of every individual in the population is evaluated, multiple individuals are stochastically selected from the current population (based on their fitness), and modified (recombined and possibly randomly mutated) to form a new population. The new population is then used in the next iteration of the algorithm. Commonly, the algorithm terminates when either a maximum number of generations has been produced, or a satisfactory fitness level has been reached for the population. If the algorithm has terminated due to a maximum number of generations, a satisfactory solution may or may not have been reached.

A typical genetic algorithm requires two things to be defined:

1. Genetic representation of the solution domain,
2. Fitness function to evaluate the solution domain.

A standard representation of the solution is as an array of bits. Arrays of other types and structures can be used in essentially the same way. The main property that makes these genetic representations convenient is that their parts are easily aligned due to their fixed size that facilitates simple crossover operation. Variable length representations may also be used, but crossover implementation is more complex in this case. Tree-like representations are explored in Genetic programming and graph-form representations are explored in Evolutionary programming.

The fitness function is defined over the genetic representation and measures the quality of the represented solution. The fitness function is always problem dependent. A representation of a solution might be an array of bits, where each bit represents a different object, and the value of the bit (0 or 1) represents whether or not the object. In some problems, it is hard or even impossible to define the fitness expression; in these cases, interactive genetic algorithms are used. Once we have the genetic representation and the fitness function defined, GA proceeds to initialize a population of solutions randomly, and then improve it through repetitive application of mutation, crossover, and inversion and selection operators.

3.2. METHODS OF REPRESENTATION

Before a genetic algorithm can be put to work on any problem, a method is needed to encode potential solutions to that problem in a form that a computer can process.

- ❖ One common approach is to encode solutions as binary strings: sequences of 1's and 0's, where the digit at each position represents the value of some aspect of the solution.
- ❖ Another, similar approach is to encode solutions as arrays of integers or decimal numbers, with each position again representing some particular aspect of the solution. This approach allows for greater precision and complexity than the comparatively restricted method of using binary numbers only and often "is intuitively closer to the problem space"
- ❖ A third approach is to represent individuals in a GA as strings of letters, where each letter again stands for a specific aspect of the solution. One example of this technique is Hiroaki Kitano's "grammatical encoding" approach, where a GA was put to the task of evolving a simple set of rules called a context-free grammar that was in turn used to generate neural networks for a variety of problems

3.3. METHOD OF APPROACH

Steps involved in solving problems that are related optimization and detailed discussion is given below.

INITIALIZATION

Initially many individual solutions are randomly generated to form an initial population. The population size depends on the nature of the problem, but typically contains several hundreds or thousands of possible solutions. Traditionally, the population is generated randomly, covering the entire range of possible solutions (the search space). Occasionally, the solutions may be "seeded" in areas where optimal solutions are likely to be found.

SELECTION

During each successive generation, a proportion of the existing population is selected to breed a new generation. Individual solutions are selected through a fitness-based process, where fitter solutions (as measured by a fitness function) are typically more likely to be selected. Certain selection methods rate the fitness of each solution and preferentially select the best solutions. Other methods rate only a random sample of the population, as this process may be very time-consuming.

Most functions are stochastic and designed so that a small proportion of less fit solutions are selected. This helps keep the diversity of the population large, preventing premature convergence on poor solutions. Popular and well-studied selection methods include roulette wheel selection and tournament selection.

METHODS OF SELECTION

There are many different techniques which a genetic algorithm can use to select the individuals to be copied over into the next generation, but listed below are some of the most common methods. Some of these methods are mutually exclusive, but others can be and often are used in combination.

1) ELITIST SELECTION

The fit members of each generation are guaranteed to be selected. (Most GAs does not use pure elitism, but instead use a modified form where the

single best or a few of the best, individuals from each generation are copied into the next generation just in case nothing better turns up.)

2) FITNESS-PROPORTIONATE SELECTION

More fit individuals are more likely, but not certain, to be selected.

3) ROULETTE-WHEEL SELECTION

A form of fitness-proportionate selection in which the chance of an individual's being selected is proportional to the amount by which its fitness is greater or less than its competitors' fitness. Conceptually, this can be represented as a game of roulette - each individual gets a slice of the wheel, but more fit ones get larger slices than less fit ones. The wheel is then spun, and whichever individual "owns" the section on which it lands each time is chosen.

4) SCALING SELECTION

As the average fitness of the population increases, the strength of the selective pressure also increases and the fitness function becomes more discriminating. This method can be helpful in making the best selection later on when all individuals have relatively high fitness and only small differences in fitness distinguish one from another.

5) TOURNAMENT SELECTION

Subgroups of individuals are chosen from the larger population, and members of each subgroup compete against each other. Only one individual from each subgroup is chosen to reproduce.

6) RANK SELECTION

Each individual in the population is assigned a numerical rank based on fitness, and selection is based on these ranking rather than absolute differences in fitness. The advantage of this method is that it can prevent very fit individuals from gaining dominance early at the expense of less fit ones, which would reduce the population's genetic diversity and might hinder attempts to find an acceptable solution.

7) GENERATIONAL SELECTION

The offspring of the individuals selected from each generation become the entire next generation. No individuals are retained between generations.

8) STEADY-STATE SELECTION

The offspring of the individuals selected from each generation go back into the pre-existing gene pool, replacing some of the less fit members of the previous generation. Some individuals are retained between generations.

9) HIERARCHICAL SELECTION

Individuals go through multiple rounds of selection each generation. Lower-level evaluations are faster and less discriminating, while those that survive to higher levels are evaluated more rigorously. The advantage of this method is that it reduces overall computation time by using faster, less selective evaluation to weed out the majority of individuals that show little or no promise, and only subjecting those who survive this initial test to more rigorous and more computationally expensive fitness evaluation.

REPRODUCTION

The next step is to generate a second generation population of solutions from those selected through genetic operators like crossover (also called recombination), and/or mutation. For each new solution to be produced, a pair of "parent" solutions is selected for breeding from the pool selected previously.

❖ CROSSOVER

Crossover is a genetic operator used to vary the programming of a chromosome or chromosomes from one generation to the next. It is analogous to reproduction and biological crossover, upon which genetic algorithms are based.

Many crossover techniques exist for organisms which use different data structures to store themselves.

1) ONE-POINT CROSSOVER

A single crossover point on both parents' organism strings is selected. All data beyond that point in either organism string is swapped between the two parent organisms. The resulting organisms are the children.

2) TWO-POINT CROSSOVER

Two-point crossover calls for two points to be selected on the parent organism strings. Everything between the two points is swapped between the parent organisms, rendering two child organisms.

3) CUT AND SPLICE

Another crossover variant, the "cut and splice" approach, results in a change in length of the children strings. The reason for this difference is that each parent string has a separate choice of crossover point.

4) UNIFORM CROSSOVER AND HALF UNIFORM CROSSOVER

In both these schemes, the two parents are combined to produce two new offspring. In the uniform crossover scheme (UX) individual bits in the string are compared between two parents. The bits are swapped with a fixed probability, typically 0.5. In the half uniform crossover scheme (HUX), exactly half of the non matching bits are swapped. Thus first the Hamming distance (the number of differing bits) is calculated. This number is divided by two. The resulting number is how many of the bits that do not match between the two parents will be swapped.

CROSSOVER FOR ORDERED CHROMOSOMES

Depending on how the chromosome represents the solution, a direct swap may not be possible. One such case is when the chromosome is an ordered list, such as ordered lists the cities to be travelled for the traveling salesman problem. A crossover point is selected on the parents. Since the chromosome is an ordered list, a direct swap would introduce duplicates and remove necessary candidates from the list. Instead, the chromosome up to the crossover point is retained for each parent. The information after the crossover point is ordered as it is ordered in the other parent. For example, if our two parents are ABCDEFGHI and IGAHFDBEC and our crossover point is after the fourth character, then the resulting children would be ABCDIGHFE and IGAHBCDEF.

CROSSOVER BIASES

For crossover operators which exchange contiguous sections of the chromosomes (e.g. k-point) the ordering of the variables may become important. This is particularly true when good solutions contain building blocks which might be disrupted by a non-respectful crossover operator.

❖ MUTATION

It is a genetic operator used to maintain genetic diversity from one generation of a population of chromosomes to the next. It is analogous to biological mutation.

The classic example of a mutation operator involves a probability that an arbitrary bit in a genetic sequence will be changed from its original state. A common method of implementing the mutation operator involves generating a random variable for each bit in a sequence. This random variable tells whether or not a particular bit will be modified.

The purpose of mutation in GAs is to allow the algorithm to avoid local minima by preventing the population of chromosomes from becoming too similar to each other, thus slowing or even stopping evolution. This reasoning also explains the fact that most GA systems avoid only taking the fittest of the population in generating the next but rather a random (or semi-random) selection with a weighting toward those that are fitter. By producing a "child" solution using the above methods of crossover and mutation, a new solution is created which typically shares many of the characteristics of its "parents". New parents are selected for each child, and the process continues until a new population of solutions of appropriate size is generated.

These processes ultimately result in the next generation population of chromosomes that is different from the initial generation. Generally the average fitness will have increased by this procedure for the population, since only the best organisms from the first generation are selected for breeding, along with a small proportion of less fit solutions, for reasons already mentioned above.

TERMINATION

This generational process is repeated until a termination condition has been reached. Common terminating conditions are

- 1) A solution is found that satisfies minimum criteria.
- 2) Fixed number of generations reached.
- 3) Allocated budget (computation time) reached.
- 4) The highest ranking solution's fitness is reaching or has reached a plateau. Such that successive iterations no longer produce better results.
- 5) Manual inspection.
- 6) Combinations of the above.

PSEUDO CODE ALGORITHM

- 1) Choose initial population
- 2) Evaluate the fitness of each individual in the population
- 3) Repeat
 - i. Select best-ranking individuals to reproduce
 - ii. Breed new generation through crossover and mutation (genetic operations) and give birth to offspring
 - iii. Evaluate the individual fitnesses of the offspring
 - iv. Replace worst ranked part of population with offspring
- 4) Until <terminating condition>

3.4. STRENGTHS OF GA

- Genetic algorithms are intrinsically parallel

Most other algorithms are serial and can only explore the solution space to a problem in one direction at a time, and if the solution they discover turns out to be suboptimal, there is nothing to do but abandon all work previously completed and start over. However, since GAs has multiple offspring, they can explore the solution space in multiple directions at once. If one path turns out to be a dead end, they can easily eliminate it and continue work on more promising avenues, giving them a greater chance each run of finding the optimal solution.

- GA just likes as pollster on the space with the highest-fitness individuals and finds the overall best one from that group.

By evaluating the fitness of this one particular string, a genetic algorithm would be sampling each of these many spaces to which it belongs. Over many such evaluations, it would build up an increasingly accurate value for the average fitness of each of these spaces, each of which has many members. Therefore, a GA that explicitly evaluates a small number of individuals is implicitly evaluating a much larger group of individuals

- GAs are well-suited to solving problems where the space of all potential solutions is truly huge and too vast to search exhaustively in any

reasonable amount of time. GA allows it to surmount even this enormous number of possibilities, successfully finding optimal or very good results in a short period of time after directly sampling only small regions of the vast fitness landscape

- GA performs well in problems for which the fitness landscape is complex - ones where the fitness function is discontinuous, noisy, changes over time, or has many local optima.
- Genetic algorithms excel in their ability to manipulate many parameters simultaneously.

Many real-world problems cannot be stated in terms of a single value to be minimized or maximized, but must be expressed in terms of multiple objectives, usually with tradeoffs involved like one can only be improved at the expense of another.

- GA knows nothing about the problems they are deployed to solve.

One of the qualities of genetic algorithms which might at first appear to be a liability turns out to be one of their strengths. They make random changes to their candidate solutions and then use the fitness function to determine whether those changes produce an improvement.

3.5. LIMITATIONS

- GAs may have a tendency to converge towards local optima or even arbitrary points rather than the global optimum of the problem.
 - Operating on dynamic data sets is difficult
 - GAs cannot effectively solve problems in which the only fitness measure is right/wrong.
 - For specific optimization problems and problem instantiations, simpler optimization algorithms may find better solutions than genetic algorithms (given the same amount of computation time).
-

Chapter 4

IMPLEMENTATION OF REDUCTION ALGORITHMS

4.1. INTRODUCTION

In this chapter, Statement of problem, theorems used in algorithms, model reduction methods and its algorithms for solving problems along with flow charts in three different methods are discussed.

4.2. STATEMENT OF PROBLEM

The model reduction problem consists of replacing a given high order system by a low order one approximating the input/output relation of the original. Consider the linear time invariant dynamic n^{th} order system which is described by the transfer function.

$$G_n(z) = \frac{a_{n-1}z^{n-1} + a_{n-2}z^{n-2} + a_{n-3}z^{n-3} + \dots + a_1z + a_0}{z^n + b_{n-1}z^{n-1} + b_{n-2}z^{n-2} + b_{n-3}z^{n-3} + \dots + b_1z + b_0} \quad \dots\dots\dots (4.1)$$

And above transfer function can represent as

$$G_n(z) = \frac{A(z)}{B(z)} \quad \dots\dots\dots (4.2)$$

Where $A(z) = a_{n-1}z^{n-1} + a_{n-2}z^{n-2} + a_{n-3}z^{n-3} + \dots + a_1z + a_0$ and $B(z) = z^n + b_{n-1}z^{n-1} + b_{n-2}z^{n-2} + b_{n-3}z^{n-3} + \dots + b_1z + b_0$.

A reduced model of order $r < n$ is to be determined, which may be taken as:

$$R_r(z) = \frac{d_{r-1}z^{r-1} + d_{r-2}z^{r-2} + d_{r-3}z^{r-3} + \dots + d_1z + d_0}{z^n + e_{r-1}z^{r-1} + e_{r-2}z^{r-2} + e_{r-3}z^{r-3} + \dots + e_1z + e_0} \quad \dots\dots\dots (4.3)$$

And above transfer function can represent as

$$R_r(z) = \frac{D(z)}{E(z)} \quad \dots\dots\dots (4.4)$$

4.3. MARKOV PARAMETERS and H-PARAMETERS

MARKOV PARAMETERS

The transfer function $G(z)$ in equation (2.1) can also be written with regard to its power series expansion about $z=\infty$, i.e.

$$G(z) = \sum_{i=1}^{\infty} g_{-i} z^{-i} \quad \dots\dots\dots (4.5)$$

The parameters $\{ g_{-i} : i=1, 2, 3, \dots, \infty \}$ being called the markov parameters.

H-PARAMETERS

Similarly, $G(z)$ can be written in terms of power series expansion about $z=1$. this is most easily done by means of the substitution [10]

$p=z-1$ such that

$$G_n(p) = \frac{a_{n-1}p^{n-1} + a_{n-2}p^{n-2} + \dots + a_1p + a_0}{p^n + b_{n-1}p^{n-1} + \dots + b_1p + b_0} \quad \dots\dots\dots (4.6)$$

The expansion of $G(p)$ about $p = 0$ is equivalent to the expansion of $G(z)$ about $z = 1$, hence

$$G(p) = \sum_{i=1}^{\infty} h_i p^i \quad \dots\dots\dots (4.7)$$

The parameters $\{ h_i : i=1, 2, 3, \dots, \infty \}$ proportional to the system time moments [10] and these are termed the H-parameters.

4.4. ERROR POLYNOMIAL AND ITS IMPLICATIONS

The resultant error between system and model can be described the equation

$$G(z) = R(z) + \lambda(z) \quad \dots\dots\dots (4.8)$$

Where $\lambda(z)$ is a rational transfer function denoting the undesired error. Equation (4.8) can, however also, be written from equation (4.2), (4.4) as

$$A(z) * E(z) = D(z)B(z) + \lambda(z) * E(z) * B(z) \quad \dots\dots\dots (4.9)$$

Such that error polynomial may now be defined as

$$W(z) = \lambda(z) * E(z) * B(z) \quad \dots\dots\dots(4.10)$$

$$\text{Where } W(z) = w_0 + w_1z + \dots \dots\dots + w_mz^m \quad \dots\dots\dots (4.11)$$

in which $m=n+r-1$.

The usefulness of $\{ w_i: i = 0, 1, 2, \dots, m \}$ parameters and their meaning in relation to the error between model and system at any time instant will now be discussed.

If an identical input is provided to both system and model, an error will be apparent between the respective outputs when the transfer functions are not identical. Let this input be $u(t)$ at time $t \{t = 0, + 1, +2, \dots\}$, and let the outputs, at time t , be $y(t)$ and $y_m(t)$ for system and model, respectively.

Then, the error at time t is defined as

$$v(t) = y(t) - y_m(t) \quad \dots\dots\dots (4.12)$$

This may be also written as

$$v(t) = [G(z) - R(z)] * u(t) \quad \dots\dots\dots(4.13)$$

$$\text{Hence } W(z) * u(t) = E(z) * B(z) \quad \dots\dots\dots (4.14)$$

The 'm' roots of the $W(z)$ polynomial are therefore also the zeros of the transfer function relating input to error. It follows that if the system denominator polynomial, $B(z)$, is stable, it is a requirement that the model denominator, $E(z)$ is also stable, to enable the error to tend to zero under steady-state conditions. Model stability is not generally achieved with all reduction methods [2], and is shown here to be a limiting factor for a particular model choice.

ERROR POLYNOMIAL WITH A UNIT STEP INPUT

Applying a unit step, equation (4.15) input to system in equation (4.1) and model in equation (4.3)

$$u(t) = \begin{cases} 1 & t \geq 0 \\ 0 & t < 0 \end{cases} \quad \dots\dots\dots (4.15)$$

Then, if a subsidiary error signal is defined as a

$$\overline{v(t)} = W(z) * u(t) \quad \dots\dots\dots (4.16)$$

This signal equation(4.16) is filtered by the polynomial $E(z)B(z)$ to become the error $v(t)$. At time instant $t = 1$, therefore, $\tilde{v}(1) = w_m$, and $\tilde{v}(2) = w_m + w_{m-1}$ etc. This addition of errors at each time instant can be summarized by

$$\tilde{v}(t) = \begin{cases} \sum_{i=j}^m w_i & j = m + 1 - t, 0 < t \leq m + 1 \\ 0 & , t > m + 1 \end{cases} \quad \dots\dots\dots (4.17)$$

Under steady-state conditions the error $\overline{v(t)}$ fed through to the $E(z)B(z)$ filter is thus $\sum_{i=0}^m w_i$ i.e. the summation of all the $W(z)$ coefficients. There are, therefore, 'm+1=n + r' error terms $\{\overline{v(t)}: t = 1, 2, \dots, m + 1\}$, but only $2k$, the number of model parameters to be chosen, degrees of freedom. Hence, whilst $r < n$, at least one of the $\overline{v(t)}$ values will be nonzero, i.e. the model response cannot be made to fit exactly that of the system.

Time instant(t)	Error signal $\overline{v(t)}$	Representation
0	0	
1	w_n	$\overline{w_n}$
2	$w_n + w_{n-1}$	$\overline{w_{n-1}}$
3	$w_n + w_{n-1} + w_{n-2}$	$\overline{w_{n-2}}$
.
.
.
N	$w_n + w_{n-1} + \dots \dots \dots + w_1 + w_0$	$\overline{w_0}$

Table 4.1: Error signal co-efficients

There are '2k' model parameters to be chosen. Thus, there '2k' degree of freedom with respect to the 'n' number of error co-efficients $\overline{w_i}$. one further polynomial must be introduced, this being $W(p)$ obtained from equation (4.6). Hence,

$$W(p) = \widetilde{w}_m p^m + \widetilde{w}_{m-1} p^{m-1} + \dots \dots \dots + \widetilde{w}_1 p + \widetilde{w}_0 \quad \dots\dots\dots (4.18)$$

Such that \widetilde{w}_i co-efficients can be obtained simply from Pascal triangle.

For example $k=2$, then total 4 model parameters to be chosen.

$$\begin{aligned} \widetilde{w}_0 &= w_0 + w_1 + w_2 + w_3 + w_4 = \overline{w_0} \\ \widetilde{w}_1 &= w_1 + 2w_2 + 3w_3 + 4w_4 \\ \widetilde{w}_2 &= w_2 + 3w_3 + 4w_4 \\ \widetilde{w}_3 &= w_3 + 3w_4 \\ \widetilde{w}_4 &= w_4 = \overline{w_4} \end{aligned}$$

It must be noted that

$$\widetilde{w}_0 = \overline{w}_0 \quad \dots\dots\dots (4.19)$$

$$\widetilde{w}_m = \overline{w}_m \quad \dots\dots\dots (4.20)$$

4.5. THEOREMS

In this section theorems discussed about two important theorems which will be useful in algorithm 1, algorithm 3 which will be discussed in latter sections to find stable reduced models for higher order systems.

4.5. 1.THEOREM 1

Statement:

Equating the \overline{w}_i parameters to zero for $i = m - j + 1, i = 1, 2, \dots, j-1$; where $0 < j \leq 2k$ by means of correct model parameter selection, will result in the first 'i' Markov parameters of the system and the first 'j' Markov parameters of the model being equal.

Proof:

The system markov parameters can be found from $\frac{A(z)}{B(z)}$ as g_{-1}, g_{-2}, \dots etc., where as the model markov parameters, obtained from $\frac{D(z)}{E(z)}$ will be written as $\overline{g}_{-1}, \overline{g}_{-2}, \overline{g}_{-3}, \dots$ etc. then defining the function

$$\Delta g_{-i} = g_{-i} - \overline{g}_{-i-1} \quad \dots\dots\dots (4.21)$$

Where $\Delta g_i = 0$ if and only if 'ith' markov parameter of the system is equal to 'ith' markov parameter of the model. It follows that:

$$\frac{A(z)}{B(z)} - \frac{D(z)}{E(z)} = \Delta g_{-1}z^{-1} + \Delta g_{-2}z^{-2} + \dots \dots \dots = \sum_{i=1}^{\infty} \Delta g_{-i}z^{-i} \quad \dots\dots\dots (4.22)$$

And from equation (4.9)

$$\lambda(z) = \sum_{i=1}^{\infty} g_{-i}z^{-i} \quad \dots\dots\dots (4.23)$$

From equation (4.10)

$$W(z) = E(z) * B(z) * \left[\sum_{i=1}^{\infty} g_{-i}z^{-i} \right] \quad \dots\dots\dots (4.24)$$

Therefore, on condition that the term is nonzero, which must be true for a 'kth' order model of 'nth' order system, by sequentially setting the W_i terms, $\{W_m, W_{m-1}, \dots\}$ to zero. The $\Delta g_{-i} \{ \Delta g_{-1}, \Delta g_{-2}, \dots \}$ terms are consequently also set to zero. This can be seen, by equating like powers of 'z' in equation (4.20) as

$$\begin{aligned}
w_m &= e_r b_n \Delta g_{-1} \\
w_{m-1} &= e_r b_n \Delta g_{-2} + (e_r b_{n-1} + e_{r-1} b_n) \Delta g_{-1} \\
w_{m-2} &= e_r b_n \Delta g_{-3} + (e_r b_{n-1} + e_{r-1} b_n) \Delta g_{-2} + (e_{r-2} b_n + e_{r-1} b_{n-1} + e_r b_{n-2}) \Delta g_{-1} \\
&\dots\dots\dots (4.25)
\end{aligned}$$

The error coefficients \overline{w}_i , are given by

$$\overline{w}_i = \sum_{j=i}^m w_j \quad i = 0, 1, 2, \dots, m; \quad \dots\dots\dots (4.26)$$

Must then necessarily have the same effect on the parameters Δg_{-i} if they themselves are sequentially set to zero. Starting, initially with \overline{w}_m . Theorem 1 shows how Markov parameters can be matched between a model and the system which it is intended to approximate by simply equating \overline{w}_i parameters to zero. It can be seen quite clearly matching the Markov parameters will equate 'earlier' error coefficients to zero, or, in other words initial part of the system and model time responses will be identical under these conditions.

4.5.2. THEOREM 2

Statement:

Equating the \overline{w}_i parameters to zero for $i = m - j + 1, i = 1, 2, \dots, j-1$; where $0 < j \leq 2k$ by means of correct model parameter selection, will result in the first 'i' Markov parameters of the system and the first 'j' H-parameters of the model being equal.

Proof:

The system markov parameters can be found from $\frac{A(z)}{B(z)}$ as h_1, h_2, \dots etc, where as the model markov parameters, obtained from $\frac{D(z)}{E(z)}$ will be written as $\overline{h}_1, \overline{h}_2, \dots$ Etc.

Then defining the function

$$\Delta h_i = h_{-i} - \overline{h_{-i-1}} \quad \dots\dots\dots (4.27)$$

Where $\Delta h_i = 0$ if and only if 'ith' H-parameter of the system is equal to 'ith' markov parameter of the model. It follows that:

$$\frac{A(p)}{B(p)} - \frac{D(p)}{E(p)} = \Delta h_0 + \Delta h_1 p + \Delta h_2 p^2 + \dots \dots \dots = \sum_{i=1}^{\infty} h_i p^i \quad \dots\dots\dots (4.28)$$

And from equation(4.9)

$$\lambda(p) = \sum_{i=1}^{\infty} h_i p^i \quad \dots\dots\dots (4.29)$$

From equation (4.10)

$$W(p) = E(p) * B(p) * \left[\sum_{i=1}^{\infty} h_i z^i \right] \quad \dots\dots\dots (4.30)$$

Let the model transfer function be defined by

$$R_m(p) = \frac{d'_{n-1}p^{n-1} + d'_{n-2}p^{n-2} + d'_{n-3}p^{n-3} + \dots\dots\dots + d'_1p + d'_{n-1}p^{n-1} + d'_0}{p^n + e'_{n-1}p^{n-1} + e'_{n-2}p^{n-2} + e'_{n-3}p^{n-3} + \dots\dots\dots + e'_1p + e'_{n-1}p^{n-1} + e'_0} \quad \dots\dots\dots (4.31)$$

Therefore from the equation(4.6)and the above ,equation(4.31)on condition that term $e'_0b'_0$ is non zero, which must be true for reference input following,by sequentially setting the terms \widetilde{w}_i to zero, the terms Δh_i are consequently also set to zero. By equating like powers of 'p' in equation(4.30) we get

$$\begin{aligned} \widetilde{w}_0 &= e'_0b'_0\Delta h_0 \\ \widetilde{w}_1 &= (e'_1b'_0 + e'_0b'_1)\Delta h_0 + e'_0b'_0\Delta h_1 \\ \widetilde{w}_2 &= (e'_0b'_2 + e'_1b'_1 + e'_2b'_0)\Delta h_0 + (e'_1b'_0 + e'_0b'_1)\Delta h_1 + e'_0b'_0\Delta h_2 \quad \dots\dots\dots (4.32) \end{aligned}$$

Therefore, this theorem can be proved same as previous theorem 1. Matching the H-parameters have been shown to latter part of the model transient response to that of system [10]. This can be seen from the parameter, \widetilde{w}_0 which in effect, sets the steady-state error.

Using the two theorems introduced in this Section it is clear that with '2k' model parameters to be selected. They can be chosen such that a total of '2k' \widetilde{w}_i and \overline{w}_i coefficients can be equated with zero. However, because the \widetilde{w}_i and \overline{w}_i coefficients are linear combinations of the w_i parameters, they are by no means independent.

Markov and H-parameter matching by means of the error polynomial are, of course, special cases in which certain of the polynomial coefficients are set to zero. The method, however, also allows for minimizing all, rather than zeroing some, of the polynomial coefficients or indeed zeroing some and minimizing others. This latter approach links up [9] with other more complicated methods, again using ladder networks.

4.6. ALGORITHMS

4.6.1. ALGORITHM1: MARKOV AND H PARAMETER MATCHING

- STEP 1:** Initially, preserve the information of given higher order system transfer function about order of the given system and co-efficients of z that present in numerator and denominator terms along with its degree. Also the order of the reduced model transfer function.
- STEP 2:** Depends on reduced model transfer function order, calculate the number of variables required. If ' r ' is the order of the reduced model then total ' $2r+1$ ' variables are required in which ' $r+1$ ' variables to represent the denominator terms and ' r ' variables represent the denominator variable terms.
- STEP 3:** Develop the error polynomial $W(z)$ in terms of given system transfer function parameters and assumed variables of reduced model parameters. And generate the Pascal triangle matrix of order ' $r+n$ ' where ' n ' is the order of given system transfer function and ' r ' is the order of required reduced model transfer function for representation of the given system.
- STEP 4:** Find Markov parameters of the given system and model as specified in algorithm1. Matching Markov parameters will equate earlier error coefficients to zero or in other words initial part of the system and model time response will be identical under these conditions. This will give set of equations which are consists of given system parameters and reduced model parameters.
- STEP 5:** Find H-parameters of system and model as specified in algorithms 2. Matching H-parameters to match the latter part of the model transient response to that of system and model or in other words latter part of the system and model time response will be identical under these conditions. This will give set of equations which are consists of given system parameters and reduced model parameters.
- STEP 6:** With ' $2k$ ' model parameters to be selected and they can be chosen such that total of ' $2k$ ' error polynomial co-efficients to be zero. One particular choice is that to match first markov parameters and first ' $2k-1$ ' h-parameters to be zero, where $k=n+r-1$.
- STEP 7:** By solving the equations that are obtained from step6 and step 7 we will get the reduced model numerator and denominator variables. Apply

basic step input to both given system and reduced model, compare the results.

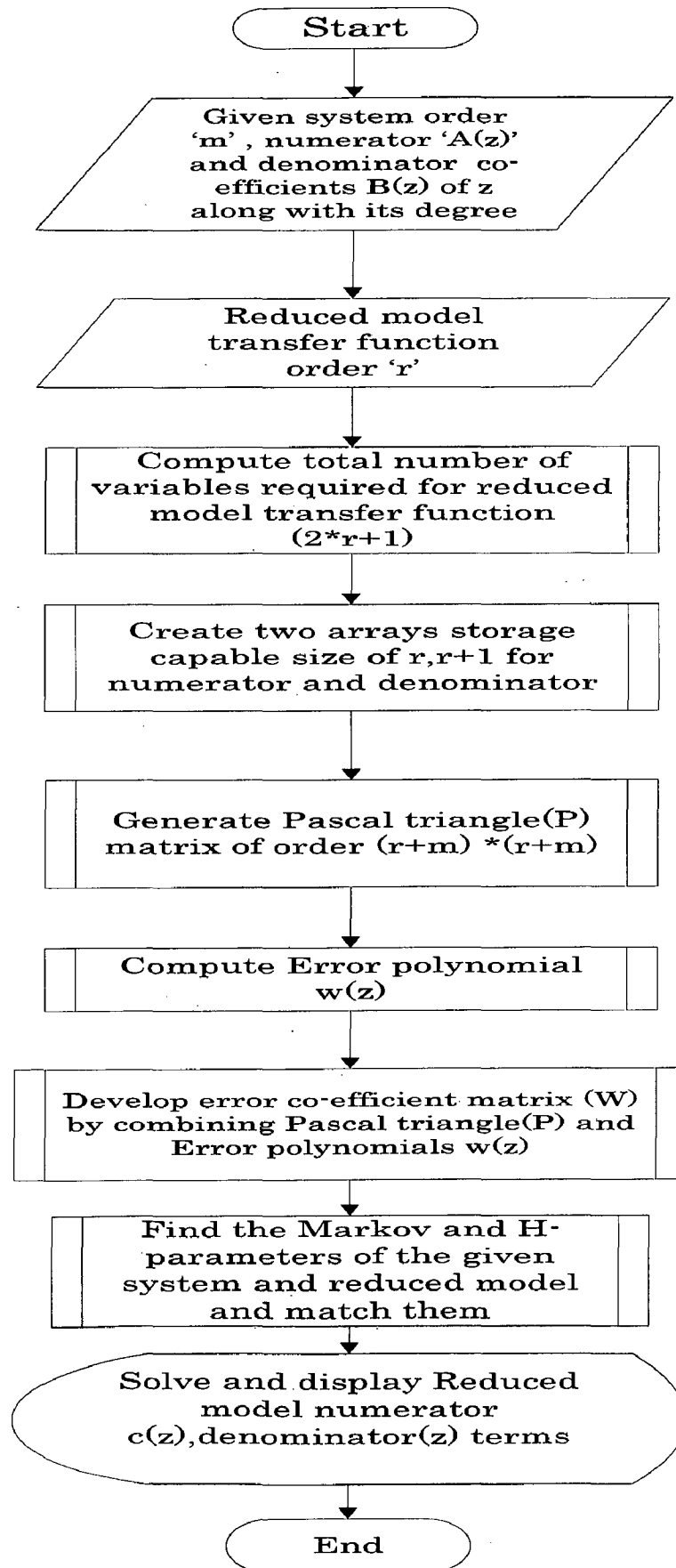


Figure 4.1.Flowchart for Algorithm 1

4.6.2. ALGORITHM2: GENETIC ALGORITHMS (DIRECT SEARCH)

- STEP 1:** Initially, preserve the information of given higher order system transfer function about order of the given system and co-efficients of z that present in numerator and denominator terms along with its degree. Also the order of the reduced model transfer function.
- STEP 2:** Depends on reduced model transfer function order, calculate the number of variables required. If 'r' is the order of the reduced model then total '2r+1' variables are required in which 'r+1' variables to represent the denominator terms variables represent the denominator variable terms.
- STEP 3:** Initialization to Genetic algorithms. For total number of variables generate binary strings or assign genotype to each variable. And it generates the random population for each variable or it takes predefined data also.
- STEP 4:** Decode them to decimal number system by using formulae. If initial range of the population lies in $[a_{min} a_{max}]$.
- If 'a1' is decimal value of binary string and it can be decoded as $a_1=10^q$
- Where $q = \frac{\log a_{max} - \log a_{min}}{2^r - 1} A_1 + \log a_{min}$ and A_1 is decimal value of binary string.
- STEP5:** Substitute randomly generated values in reduced model variables and take frequency response samples of reduced model and given system transfer function and note down the time sampling also. Samples are taken such away that no information regarding system is lost. Sampling time is very very small so that it seems to be continuous.
- STEP 6:** Calculate the mean square error between these two systems. This is fitness function of the problem. Genetic algorithm is going to searches the values for variables such that it leads to as minimum as possible i.e. optimization of fitness function.
- STEP 7:** Verify whether fitness function has reached absolutely low value or numbers of iterations are increased more than specified or numbers of generations are more than specified. Here checking of condition for termination of optimization will go on.

STEP 8: If the conditions specified in the above step are not met, optimization process will progress in specified manner explained in below steps. Firstly, we have to select population from randomly generated pool and methods are specified in chapter 3.

STEP 9: Reproduction to generate a second generation population of solutions from those selected through genetic operators like crossover (also called recombination), and/or mutation. Eliminate any repeated strings in newly generated population. The methods for crossover and mutation are specified in chapter 3, go to step5 and repeat.

4.6.3. ALGORITHM3: COMBINATION OF MARKOV AND H- PARAMETER MATCHING and GA (DIRECT SEARCH METHOD)

STEP1: Initially, preserve the information of given higher order system transfer function about order of the given system and co-efficients of z that present in numerator and denominator terms along with its degree. Also the order of the reduced model transfer function.

STEP2: Depends on reduced model transfer function order, calculate the number of variables required. If ' r ' is the order of the reduced model then total ' $2r+1$ ' variables are required in which ' $r+1$ ' variables to represent the denominator terms variables represent the denominator variable terms.

STEP 3: Apply Algorithm1 and find the denominator parameters and substitute these variables into reduced model transfer function.

STEP 4: Apply Algorithm 2 and find the numerator parameters and Obtain total reduced model transfer function.

STEP 5: Apply step input to both given system and reduced model, compare the results.

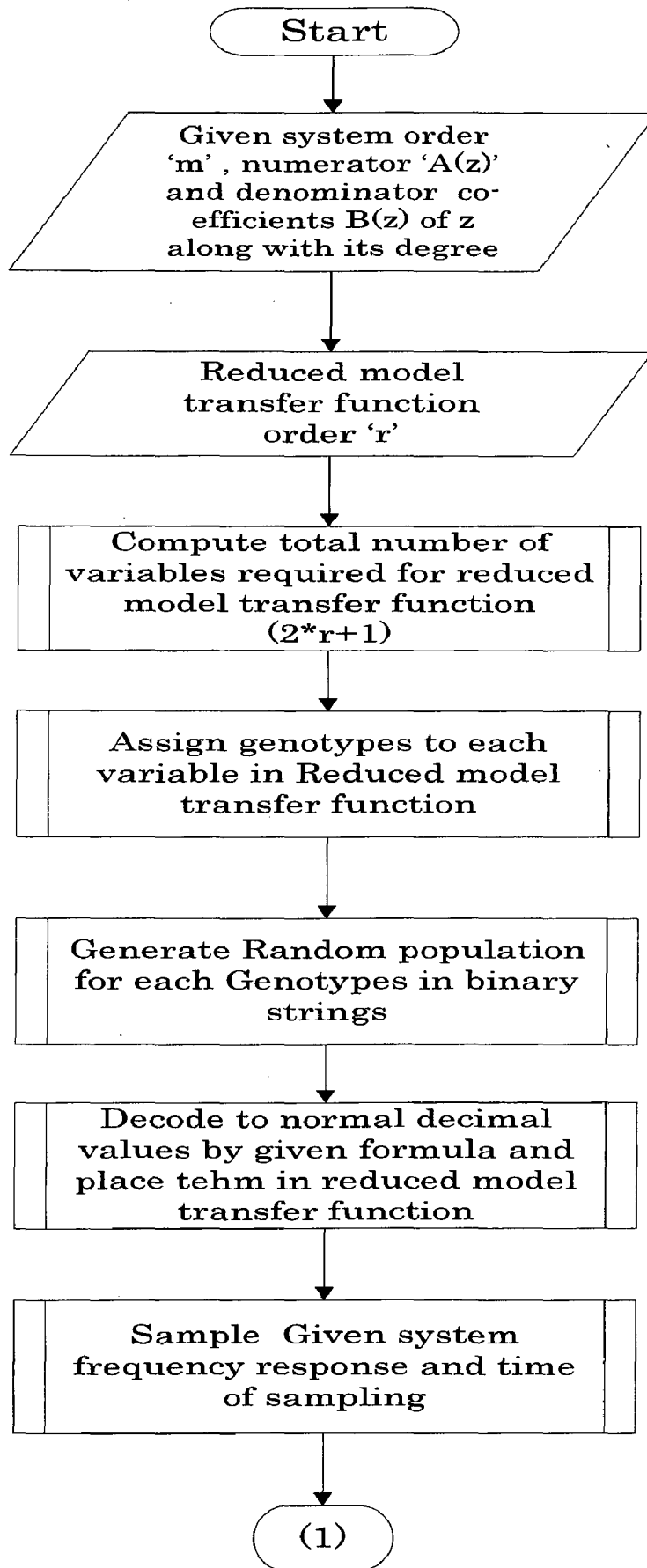


Figure 4.2.a. Flowchart for Algorithm 2

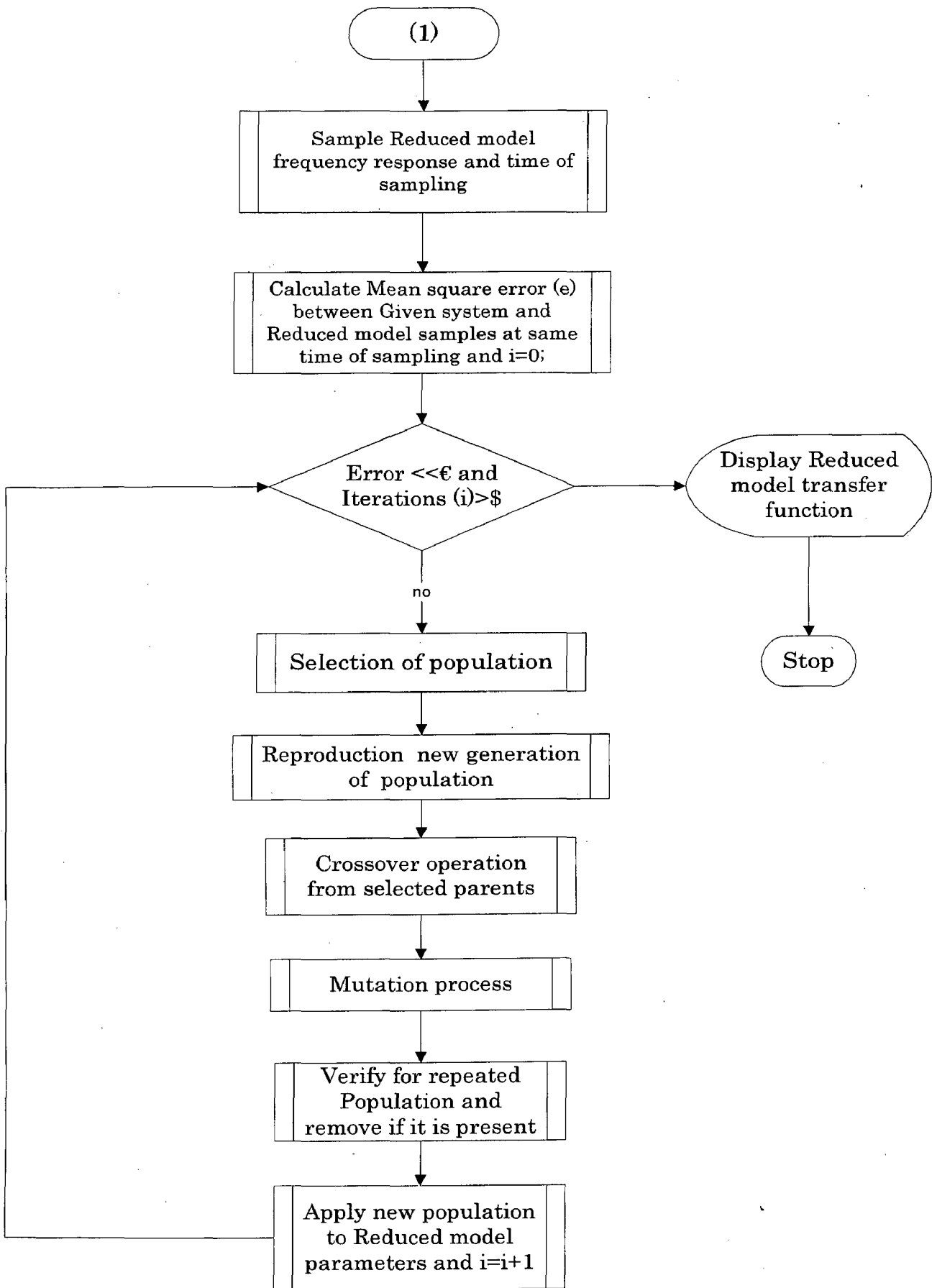


Figure 4.2.b. Flowchart for Algorithm 2

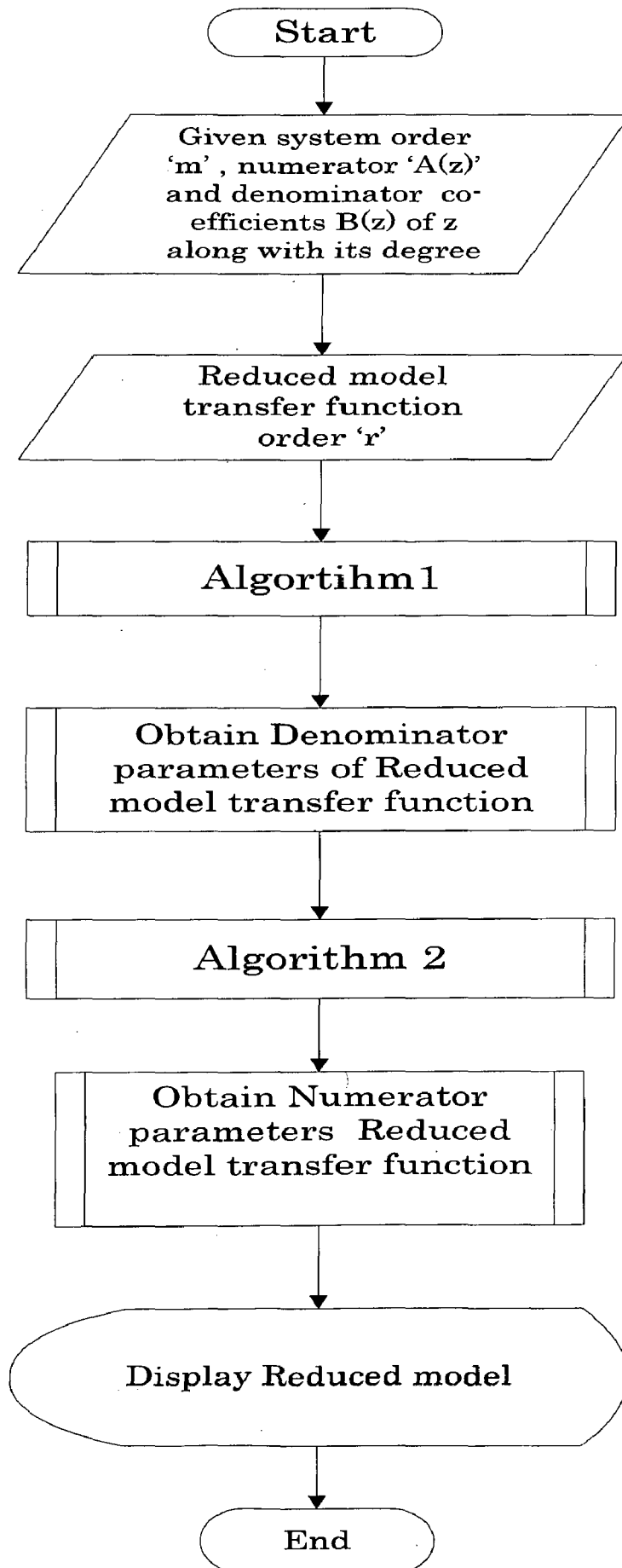


Figure 4.3. Flowchart for Algorithm 3

4.7. NUMERICAL EXAMPLE

$$G(z) = \frac{0.3124 z^3 - 0.5743 z^2 + 0.3879 z - 0.0089}{z^4 - 3.233z^3 + 3.9869z^2 - 2.2209z + 0.4723}$$

ALGORITHM 1

Given system is $\frac{A(z)}{B(z)} = \frac{0.3124 z^3 - 0.5743 z^2 + 0.3879 z - 0.0089}{z^4 - 3.233z^3 + 3.9869z^2 - 2.2209z + 0.4723}$

Here order of the system $n=4$. It is desired to use a second order model to approximate the above transfer function. Therefore, order of the model $r=2$. This can be represented as

$$R_r(z) = \frac{d_1 z + d_0}{e_2 z^2 + e_1 z + e_0} \quad \dots\dots\dots (4.33)$$

The polynomial $W(z)$ can now be formed from

$$W(z) = A(z)B(z) - D(z)B(z)$$

And error polynomial terms subscripts from 0 to $m=n+r-1=5$ i.e. $w_0, w_1, w_2, w_3, w_4, w_5$ are error polynomial co-efficients of 'z' and total 6 terms are present in error polynomial. Error co-efficient matrix in terms of original system and reduced model parameters given as below.

$$\begin{bmatrix} w_5 \\ w_4 \\ w_3 \\ w_2 \\ w_1 \\ w_0 \end{bmatrix} = \begin{bmatrix} 0.3124 & 0 & 0 & -1 & 0 \\ -0.5743 & 0.3124 & 0 & 3.233 & -1 \\ 0.3879 & -0.5743 & 0.3124 & -3.9869 & 3.233 \\ -0.0889 & 0.3879 & -0.5743 & 2.2209 & -3.9869 \\ 0 & -0.0889 & 0.3879 & -0.4723 & 2.2209 \\ 0 & 0 & -0.0889 & 0 & -0.4723 \end{bmatrix} * \begin{bmatrix} e_2 \\ e_1 \\ e_0 \\ d_1 \\ d_0 \end{bmatrix} \quad \dots\dots\dots (4.34)$$

As the model is of order $k=2$ and, a total Δg_{-i} and Δh_i of $2k=4$ can be set to zero. One particular choice is then to make, i.e. to match the first markov parameter and first three h-parameters. From the definition given

$$\bar{w}_5 = 0 = w_5 \text{ as } m = n + r - 1 = 5 \quad \dots\dots\dots (4.35)$$

$$\widetilde{w}_0 = 0 = w_0 + w_1 + w_2 + w_3 + w_4 + w_5 \quad \dots\dots\dots (4.36)$$

$$\widetilde{w}_1 = 0 = w_1 + 2w_2 + 3w_3 + 4w_4 + 5w_5 \quad \dots\dots\dots (4.37)$$

$$\widetilde{w}_2 = 0 = w_2 + 3w_3 + 6w_4 + 10w_5 \quad \dots\dots\dots (4.38)$$

Equation (4.37) represents the first markov parameter of error polynomial said to be zero. Equation (4.38), (4.39), (4.40) are representing the first three h-

parameters of error polynomial function and these are equating to zero.(since from Theorem 1 and Theorem 2 specified in section 4.5,4.6)

These equations can re written as

$$w_5 = 0$$

$$w_0 = -w_3 - 3w_4$$

$$w_1 = 3w_3 + 8w_4$$

$$w_2 = -3w_3 - 6w_4 \quad \dots\dots\dots (4.39)$$

By substituting the equations (4.39) in matrix in (4.34) we get the parameters of reduced model for specified order r =2; then final reduced model will be

$$R_2(z) = \frac{0.3124z-0.0298}{z^2-1.7369z+0.7773}$$

ALGORITHM 2

Given system is $\frac{A(z)}{B(z)} = \frac{0.3124 z^3 - 0.5743 z^2 + 0.3879 z - .0089}{z^4 - 3.233z^3 + 3.9869z^2 - 2.2209z + 0.4723}$

Here order of the system n=4.It is desired to use a second order model to approximate the above transfer function. Therefore, order of the model r=2.This can be represented as

$$R_r(z) = \frac{d_1z+d_0}{e_2z^2+e_1z+e_0} \quad \dots\dots\dots (4.40)$$

Here total number of variables are (2*r+1) =5.so Genetic algorithm assigns 5 genotypes to variables.

First step it assigns the random population size of 100, a matrix of (population size × no. of variables), i.e. 100×6.and follows algorithm 2 specified in section 4. Secondly, sample both given system and reduced system and at same instants so that calculate fitness function and select best population and apply modification to the existing population through mutation and crossover using different methods present in them and try to minimize the error. Take limit on the number of iteration, error obtained, and population generation. To avoid local minima we have to search from different initial points.

The options taken in genetic algorithm for solving above problem are specified below.

OPTIONS:

Population Type	double Vector
Population Initial Range	[0 1]
Population Size	[100×5]
Population spreading	Passions distribution
Elite Count	2
Crossover Fraction	0.8100
Migration Direction	Forward and backward
Migration Interval	20
Migration Fraction	0.2000
Generations	100
Time Limit	Infinite
Fitness Limit	-Infinite
Stall Generation Limit	50
Stall Time Limit	100
Selection Function	Roulette function
Crossover Function	Two point crossover method
Crossover fraction	0.81
Mutation	0.05

Table 4.1. Genetic algorithms options for algorithm 2

Reduced order model obtained after too many iterations or generations of population. The information regarding that is present below table (4.2).

Generation	f-count	Best fitness f(x)	Average value of fitness function F(x)	Stall iteration
1	100	9226	49000	0
2	200	9226	46710	1
3	300	7793	38530	0
4	400	7567	315300	0
.....
.....
50	5000	5386	8517	4

Table 4.2. Genetic algorithm diagnostics display for algorithm 2

$$R_2(z) = \frac{0.8174z - 0.7333}{z^2 - 1.848z + 0.857}$$

ALGORITHM 3

Given system is
$$\frac{A(z)}{B(z)} = \frac{0.3124 z^3 - 0.5743 z^2 + 0.3879 z - 0.0089}{z^4 - 3.233z^3 + 3.9869z^2 - 2.2209z + 0.4723}$$

Here order of the system $n=4$. It is desired to use a second order model to approximate the above transfer function. Therefore, order of the model $r=2$. This can be represented as

$$R_r(z) = \frac{d_1 z + d_0}{e_2 z^2 + e_1 z + e_0} \quad \dots\dots\dots (4.41)$$

It will obtain by combine method of algorithm 1 and algorithm 2 specified in sections 4.7.

From the matrix in equation (4.34) and set of equations in (4.39), we obtain the denominator parameters

By the normalization $e_2 = 1$,

and $e_1 = -1.732$ and $e_0 = 0.773$;

Equation (4.41) further reduced to

$$R_2(z) = \frac{d_1 z + d_0}{z^2 - 1.732z + 0.7773} \quad \dots\dots\dots (4.42)$$

Here total numbers of variables are $r=2$; so Genetic algorithm assigns 2 genotypes to variables. First step it assigns the random population size of 100, a matrix of (population size \times no. of variables), i.e. 100×2 . and follows algorithm 2 specified in section 4. Secondly, sample both given system and reduced system and at same instants so that calculate fitness function and select best population and apply modification to the existing population through mutation and crossover using different methods present in them and try to minimize the error. Take limit on the number of iteration, error obtained, and population generation. To avoid local minima we have to search from different initial points.

The options taken in genetic algorithm for solving above problem are specified below.

OPTIONS:

Population Type	double Vector
Population Initial Range	[0 1]
Population Size	[100×2]
Population spreading	Passions distribution
Elite Count	2
Crossover Fraction	0.8100
Migration Direction	Forward and backward
Migration Interval	20
Migration Fraction	0.2000
Generations	100
Time Limit	Infinitive
Fitness Limit	-Infinitive
Stall Generation Limit	50
Stall Time Limit	100
Selection Function	Roulette function
Crossover Function	Two point crossover method
Crossover fraction	0.81
Mutation	0.05

Table 4.3.Genetic algorithms options for algortihm3

Then finally, obtained reduced model transfer function of order =2 is given by

$$R_2(z) = \frac{0.4399z - 0.8220}{z^2 - 1.7369z + 0.7773}$$

The detailed results are discussed in next chapter with comparisons and plots of different models obtained from different algorithms and different given systems.

Chapter 5

RESULTS

5.1.PROBLEM 1:

$$G(z) = \frac{0.3124 z^3 - 0.5743 z^2 + 0.3879 z - .0089}{z^4 - 3.233z^3 + 3.9869z^2 - 2.2209z + 0.4723}$$

ALGORITHM 1

$$R_3(z) = \frac{0.3124 z^2 - 0.2461 z + 0.233}{z^3 - 1.859 z^2 + 1.003z - 0.4723}$$

$$R_2(z) = \frac{0.3124 z - 0.02985}{z^2 - 1.737z + 0.7773}$$

$$R_1(z) = \frac{0.3124}{z - 0.9554}$$

Characteristics	4 th order Given System	3 rd order Reduced model	2 rd order Reduced model	1 st order Reduced model
Peak Magnitude (M_p)	1.15	1.13	1.12	-
%Overshoot ($\%M_p$)	14.7	12.9	11.7	-
%Overshoot time(T_{M_p})	1.62	1.66	1.67	-
Settling Time (T_s)	3.34	2.52	2.58	8.57
Rise Time (T_r)	0.813	0.78	0.789	4.82
Steady state time (T_{ss})	4.5	4.5	4.5	15
Final Value (FV)	1	1	1	1

Table 6.1. Comparison of time domain characteristics of reduced models of problem 1 obtained from Algorithm1 with given system.

From table 6.1, second and third order transfer function reduction models are having approximate time domain characteristics similar to given higher order system in algorithm 1. Step responses of reduced models are shown in figure 6.1 and compared with step response e of given system.

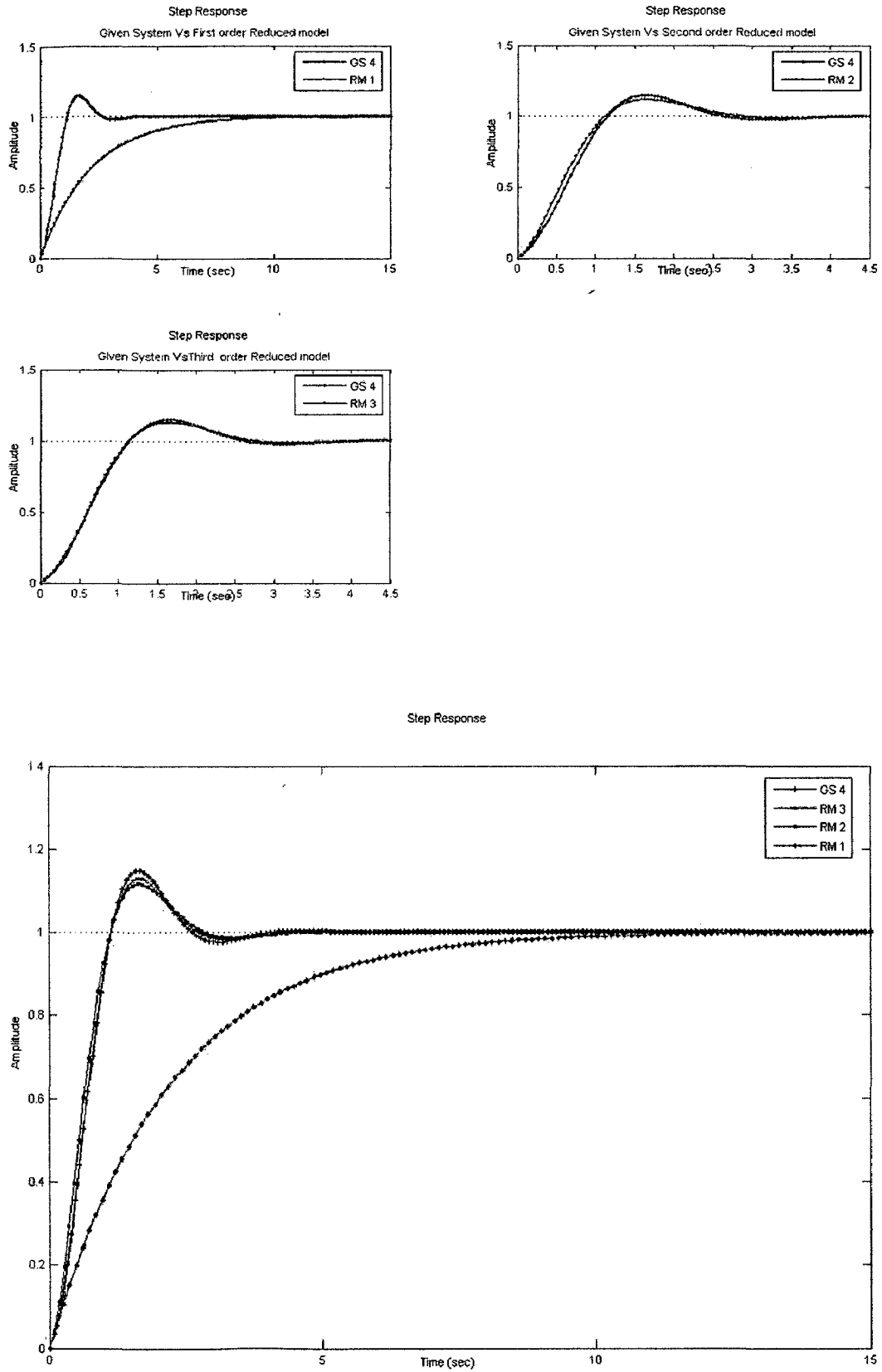


Figure 6.1. Comparison of step responses of given system Vs Reduced models of problem 1 obtained in Algorithm 1.

ALGORITHM 2

$$R_3(z) = \frac{0.1186 z^2 - 0.1512 z + 0.04081}{z^3 - 2.804 z^2 + 2.629z - 0.8236}$$

$$R_2(z) = \frac{0.8714 z - 0.7333}{z^2 - 1.848z + 0.857}$$

$$R_1(z) = \frac{1.748}{z - 0.8076}$$

Characteristics	4 th order Given System	3 rd order Reduced model	2 rd order Reduced model	1 st order Reduced model
Peak Magnitude (M_p)	1.15	1.4	1.05	-
%Overshoot ($\%M_p$)	14.7	37.7	4.86	-
%Overshoot Time (T_{Mp})	1.62	2.76	3.36	-
Settling Time (T_s)	3.34	10	5.08	1.83
Rise Time (T_r)	0.813	1.17	1.54	1.03
Steady state time (T_{ss})	4.5	18	8	4.5
Final Value (FV)	1	1	1	1

Table 6.2. Comparison of time domain characteristics of reduced models of problem 1 obtained from Algorithm2 with given system.

From table 6.2, none of reduction model obtained by Algorithm 2 is closely representing the given system. So it is difficult to express any reduced model with Genetic algorithms alone. Algorithm 3 results shown in table 6.3 and figure 6.3. will clarify this fact. Step responses of reduced models obtained from algorithm 2 and step response of given system are shown in figure 6.2.

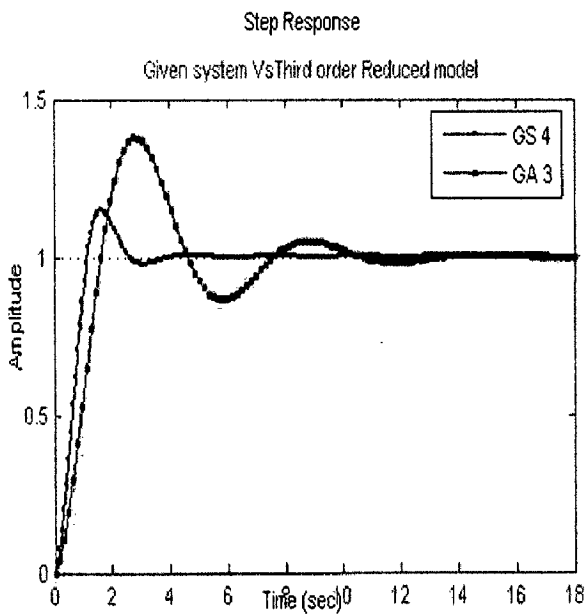
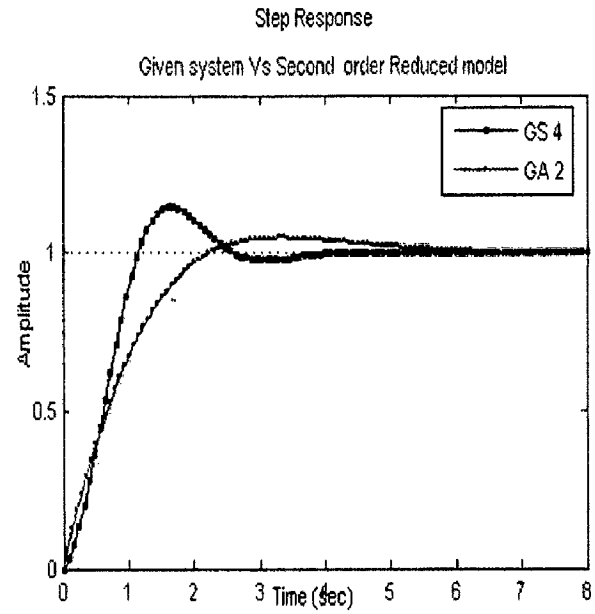
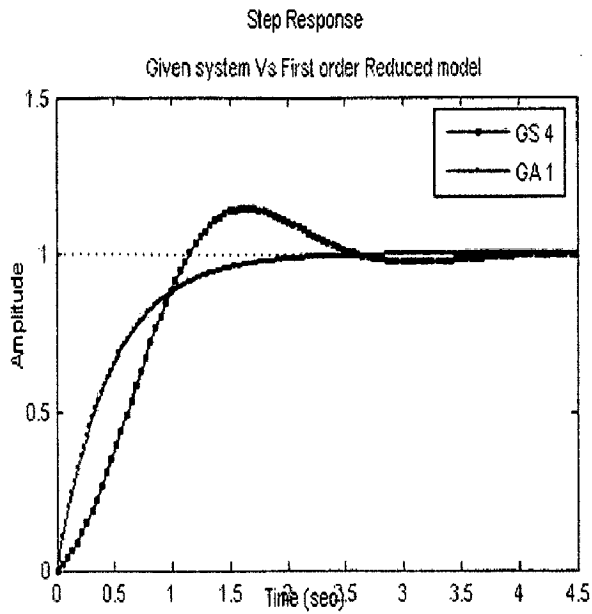


Figure 6.2. Comparison of step responses of given system Vs Reduced models of problem 1 obtained in Algorithm 2.

ALGORITHM 3

$$R_3(z) = \frac{0.85 z^2 + 1.2212 z + 0.0203}{z^3 - 1.859 z^2 + 1.003z - 0.4723}$$

$$R_2(z) = \frac{0.4399 z - 0.8220}{z^2 - 1.737z + 0.7773}$$

$$R_1(z) = \frac{0.3343}{z - 0.9554}$$

Characteristics	4 th order Given System	3 rd order Reduced model	2 rd order Reduced model	1 st order Reduced model
Peak Magnitude (M_p)	1.15	1.13	1.12	-
%Overshoot ($\%M_p$)	14.7	13.5	11.6	-
%Overshoot time(T_{M_p})	1.62	1.62	1.71	-
Settling Time (T_s)	3.34	2.5	2.66	8.57
Rise Time (T_r)	0.813	0.734	0.798	4.82
Steady state time (T_{ss})	4.5	4.5	4.5	15
Final Value (FV)	1	1	1	1

Table 6.3. Comparison of time domain characteristics of reduced models of problem 1 obtained from Algorithm2 with given system

From table 6.3. Second and Third order transfer function reduction models are having approximate similar time domain characteristics as that of given higher order system. So second and third order transfer function reduction models are best suitable for given system representation from Algorithm 3. Step responses of reduced models and given system step response are shown in figure6.3.

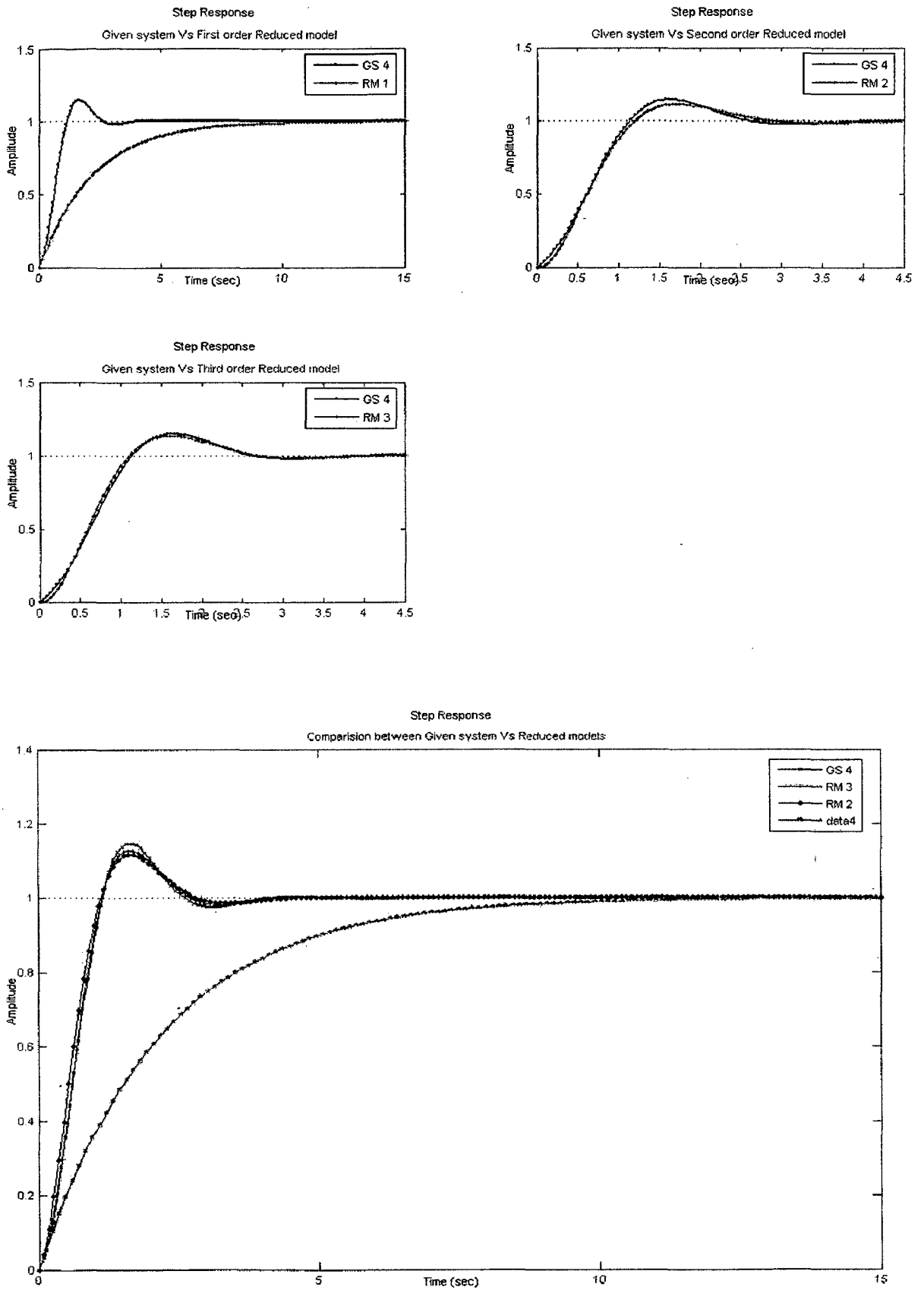


Figure 6.3. Comparison of step responses of given system Vs Reduced models of problem 1 obtained in Algorithm 3.

5.2.PROBLEM2

$$G(z) = \frac{3z^4 - 8.886z^3 + 10.0221z^2 - 5.092z + 0.9811}{z^5 - 3.7z^4 + 5.47z^3 - 4.037z^2 + 1.486z - 0.2173}$$

ALGORITHM 1

$$R_4(z) = \frac{3z^3 - 6.6261z^2 + 5.008z - 1.2974}{z^4 - 2.9560z^3 - 3.2803z^2 - 1.6343z + 0.3518}$$

$$R_3(z) = \frac{3z^2 - 4.462z + 1.679}{z^3 - 2.184z^2 + 1.49z - 0.2915}$$

$$R_2(z) = \frac{3z - 2.6644}{z^2 - 1.769z + 0.7916}$$

$$R_1(z) = \frac{3}{z - 0.7976}$$

Characteristics	5 th order Given system	4 th order Reduced model	3 rd order Reduced model	2 rd order Reduced model	1 st order Reduced model
Peak Magnitude (M_p)	1.26	1.27	1.25	1.17	-
%Overshoot ($\%M_p$)	26	26.9	25.4	16.7	-
%Overshoot time (T_{M_p})	1.71	1.65	1.68	1.44	-
Settling Time (T_s)	4.94	4.91	4.94	3.11	1.73
Rise Time (T_r)	0.71	0.695	0.705	0.577	0.972
Steady state time (T_{ss})	9	9	8	5	2.5
Final Value (FV)	1	1	1	1	1

Table 6.4. Comparison of time domain characteristics of reduced models of problem 2 obtained from Algorithm 1 with given system.

From table 6.4, Fourth and third order transfer function reduction models are having best approximate similar time domain characteristics as that of given higher order system in Algorithm 1. Second order system also giving good characteristics representation. The step responses are shown in figure 6.4. and compared with step response of given system.

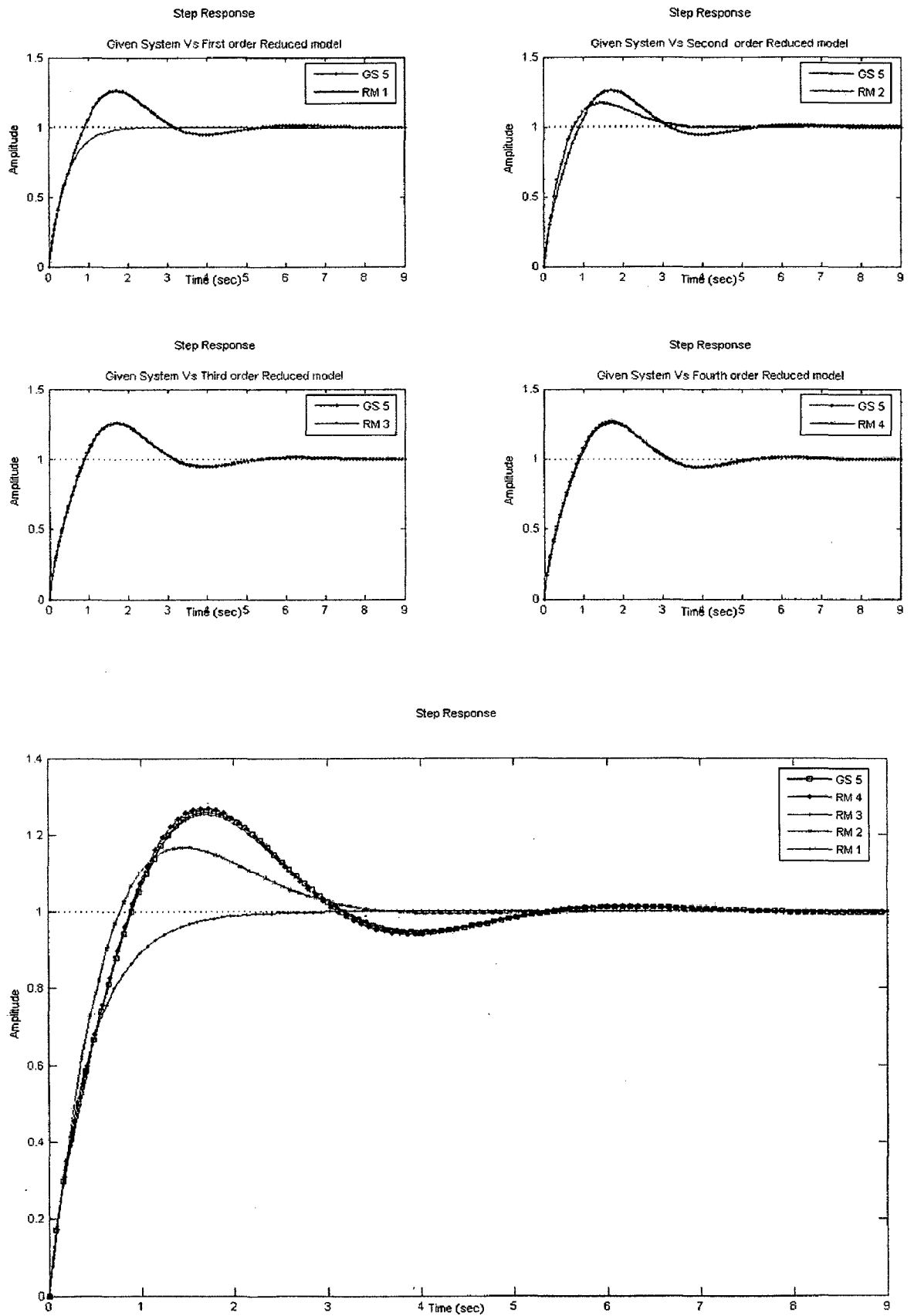


Figure 6.4. Comparison of step responses of given system Vs Reduced models of problem 2 obtained in Algorithm 1.

ALGORITHM 2

$$R_4(z) = \frac{0.2565 z^3 - 0.6797 z^2 + 0.5966 z - 0.1729}{z^4 - 3.699z^3 + 5.173 z^2 - 3.249 z + 0.7743}$$

$$R_3(z) = \frac{0.1272 z^2 - 0.1628 z + 0.04416}{z^3 - 2.184 z^2 + 2.642z - 0.8272}$$

$$R_2(z) = \frac{0.8386 z - 0.7633}{z^2 - 1.901 z + 0.9068}$$

$$R_1(z) = \frac{0.8963}{z - 0.9425}$$

Characteristics	5 th order Given system	3 rd order Reduced model	2 rd order Reduced model	1 st order Reduced model
Peak Magnitude (M_p)	1.26	1.34	1.14	-
%Overshoot ($\%M_p$)	26	33.6	13.6	-
%Overshoot time (T_{Mp})	1.71	3.95	3.61	-
Settling Time (T_s)	4.94	13.4	6.52	6.61
Rise Time (T_r)	0.71	1.83	1.63	3.71
Steady state time (T_{ss})	9	20	12	10
Final Value (FV)	1	1	1	1

Table 6.5. Comparison of time domain characteristics of reduced models of problem 2 obtained from Algorithm 2 with given system.

From table 6.5. None of reduction model obtained by Algorithm 2 is closely representing the given system. So it is difficult to express any reduced model with Genetic algorithms alone. Algorithm 3 results shown in table 6.6 and figure 6.6. will clarify this fact. Step responses of reduced models obtained from algorithm 2 and step response of given system are shown in figure 6.5.

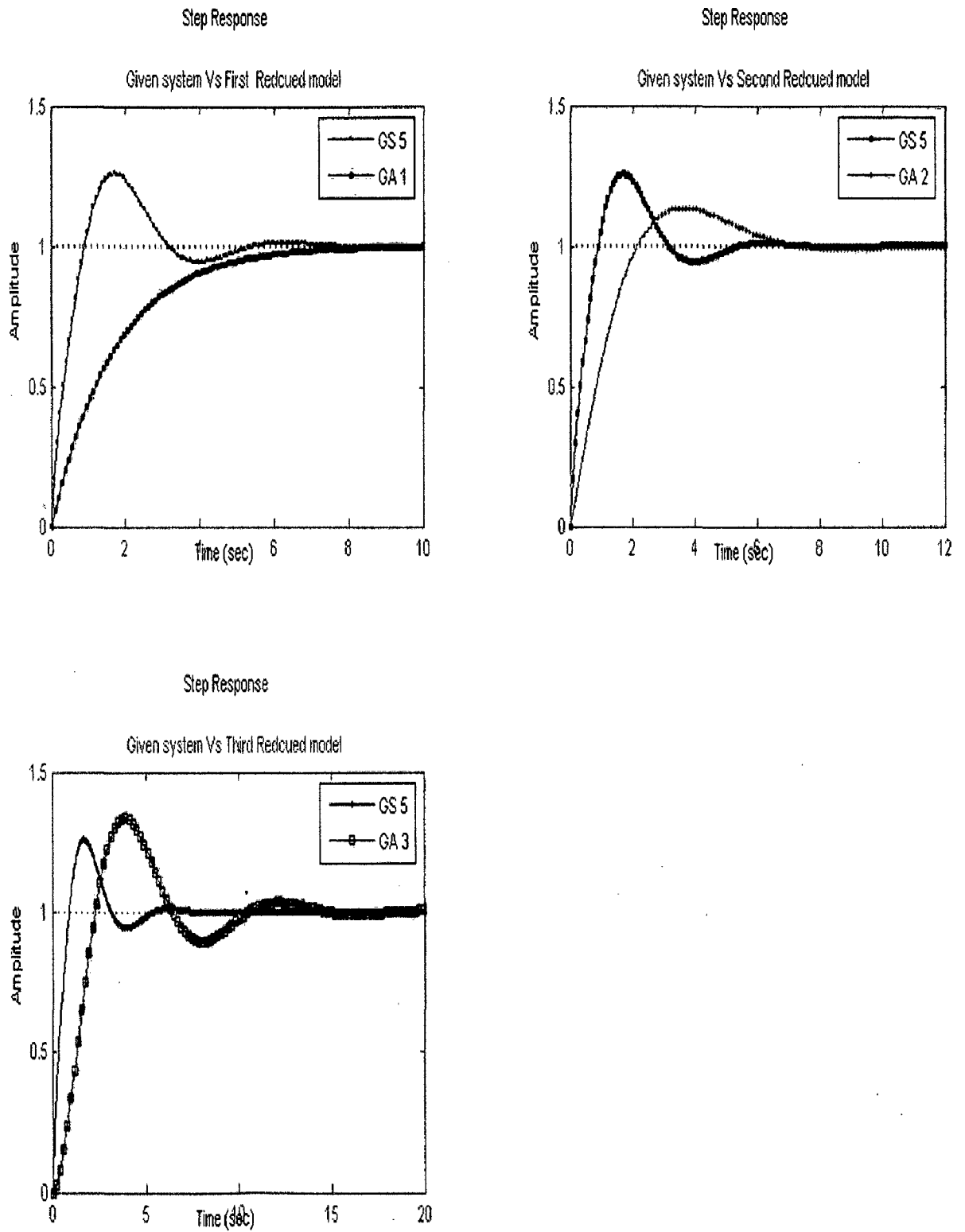


Figure 6.5. Comparison of step responses of given system Vs Reduced models of problem 2 obtained in Algorithm 2.

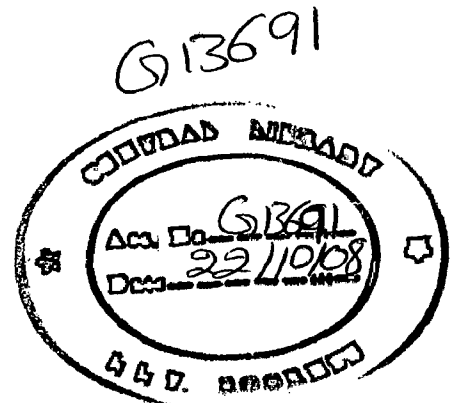
ALGORITHM 3

$$R_4(z) = \frac{1.095 z^3 + 0.09146 z^2 + 0.8386 z + 1.45}{z^4 - 2.9560z^3 - 3.2803 z^2 - 1.6343 z + 0.3518}$$

$$R_3(z) = \frac{0.07168 z^2 + 0.07546 z + 1.56}{z^3 - 2.184 z^2 + 1.49z - 0.2915}$$

$$R_2(z) = \frac{2.93 z - 2.67}{z^2 - 1.769 z + 0.7916}$$

$$R_1(z) = \frac{2.6576}{z - 0.7976}$$



Characteristics	5 th order Given system	4 th order Reduced model	3 rd order Reduced model	2 nd order Reduced model	1 st order Reduced model
Peak Magnitude (M _p)	1.26	1.22	1.25	1.28	-
%Overshoot (%M _p)	26	21.7	25.4	26.1	-
%Overshoot time(T _{Mp})	1.71	2.55	1.68	1.26	-
Settling Time (T _s)	4.94	5.67	4.94	3.06	1.73
Rise Time (T _r)	0.71	1.03	0.705	0.427	0.972
Steady state time (T _{ss})	9	9	9	9	9
Final Value (FV)	1	1	1	1	1

Table 6.6. Comparison of time domain characteristics of reduced models of problem 2 obtained from Algorithm 3 with given system.

From table 6.6. Fourth and Third order transfer function reduction models are having approximate similar time domain characteristics as that of given higher order system. Second order transfer function reduction models also suitable for given system representation from Algorithm 3. Step responses of reduced models and given system step response are shown in figure 6.6.

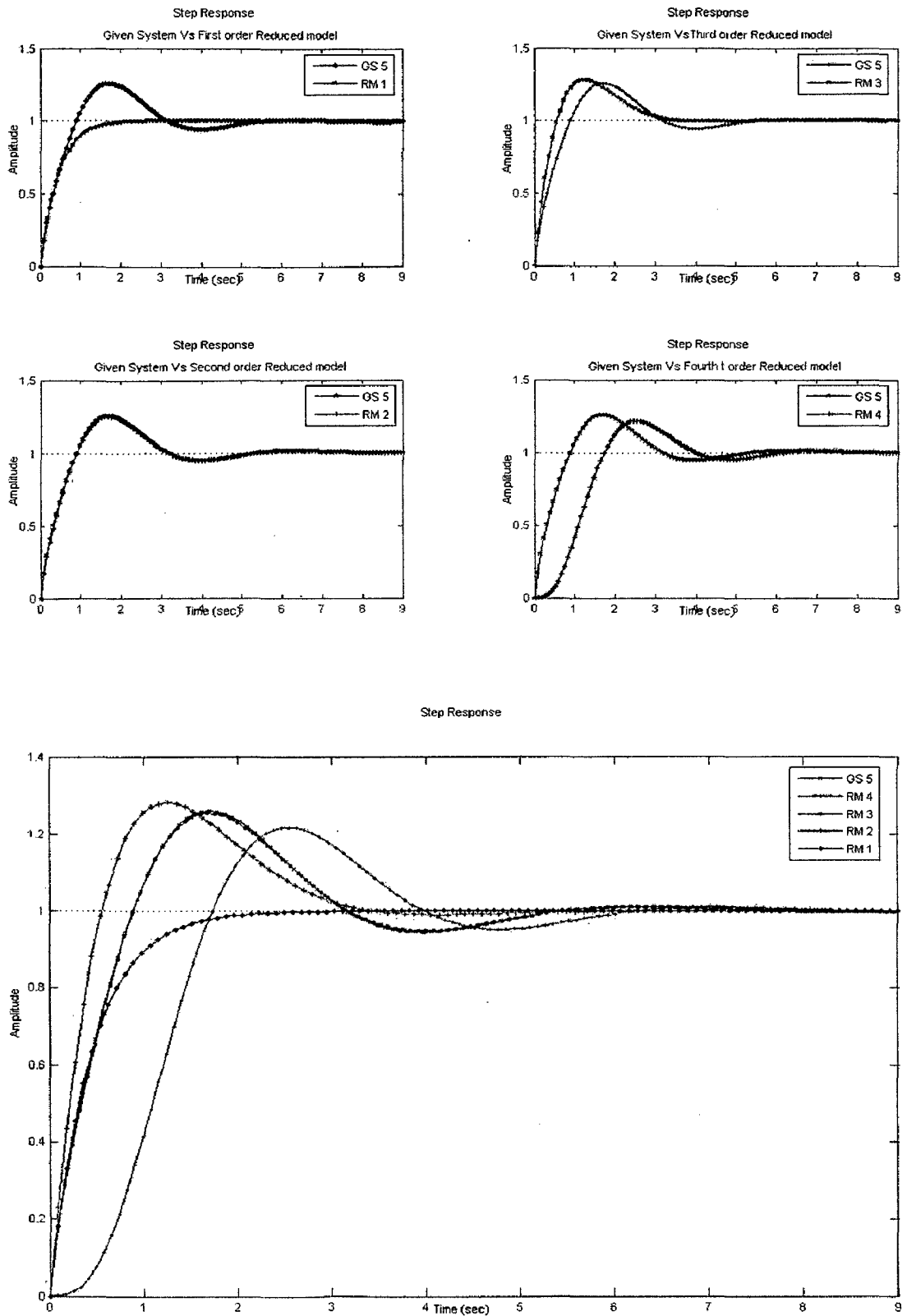


Figure 6.6. Comparison of step responses of given system Vs Reduced models of problem 2 obtained in Algorithm 3.

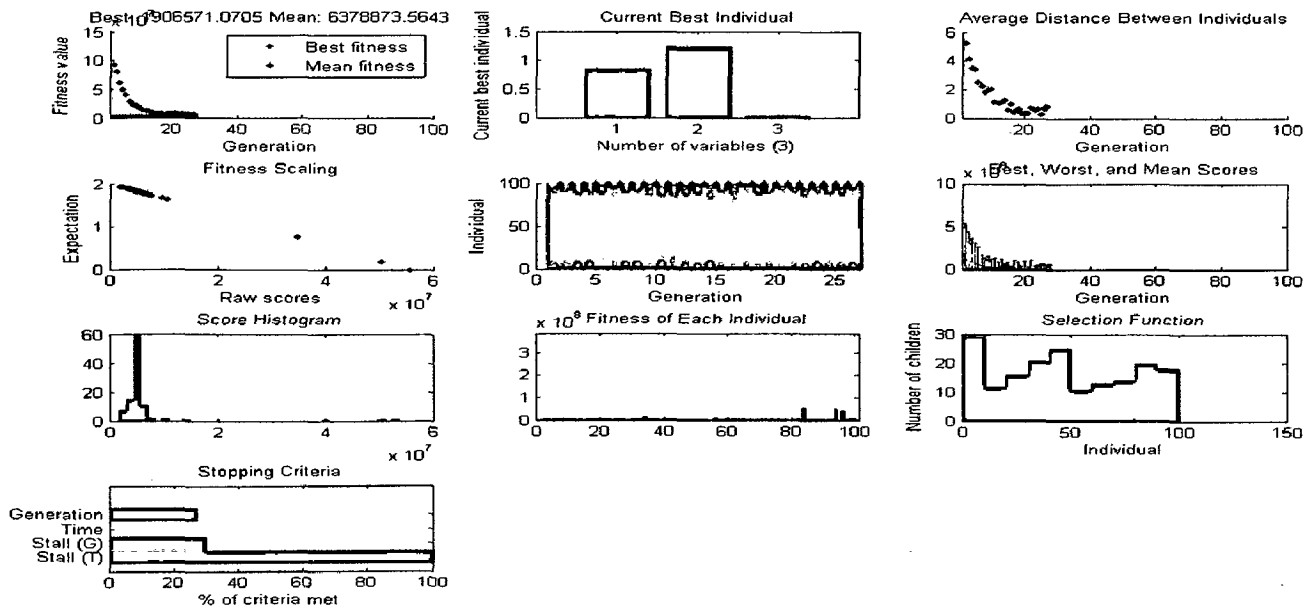


Figure 6.7. Genetic Algorithm plots for different parameters specified in program

5.3. COMPARISON OF REDUCTION MODELS OBTAINED IN DIFFERENT ALGORITHMS WITH GIVEN SYSTEM

5.3.1. PROBLEM 1

Characteristics	4 th order Given System	3 rd order Reduced model (Algorithm1)	3 rd order Reduced model (Algorithm2)	3 rd order Reduced model (Algorithm3)
Peak Magnitude (M_p)	1.15	1.13	1.4	1.13
%Overshoot ($\%M_p$)	14.7	12.9	37.7	13.5
%Overshoot Time (T_{Mp})	1.62	1.66	2.76	1.62
Settling Time (T_s)	3.34	2.52	10	2.5
Rise Time (T_r)	0.813	0.78	1.17	0.734
Steady state time (T_{ss})	4.5	4.5	18	4.5
Final Value (FV)	1	1	1	1

Table 6.7. Comparison of time domain characteristics of 3rd order Reduced models of problem-1 Vs Given System.

Characteristics	4 th order Given System	2 nd order Reduced model (Algoritihm1)	2 nd order Reduced model (Algoritihm2)	2 nd order Reduced model (Algoritihm3)
Peak Magnitude (M_p)	1.15	1.12	1.05	1.12
%Overshoot ($\%M_p$)	14.7	11.7	4.86	11.6
%Overshoot Time (T_{Mp})	1.62	1.67	3.36	1.71
Settling Time (T_s)	3.34	2.58	5.08	2.66
Rise Time (T_r)	0.813	0.789	1.54	0.798
Steady state time (T_{ss})	4.5	4.5	8	4.5
Final Value (FV)	1	1	1	1

Table 6.8. comparison of time domain characteristics of 2nd order Reduced models of problem 1 Vs Given System.

Characteristics	4 th order Given System	1 st order Reduced model (Algoritihm1)	1 st order Reduced model (Algoritihm2)	1 st order Reduced model (Algoritihm3)
Peak Magnitude (M_p)	1.15	-	-	-
%Overshoot ($\%M_p$)	14.7	-	-	-
%Overshoot Time (T_{Mp})	1.62	-	-	-
Settling Time (T_s)	3.34	1.83	8.57	8.57
Rise Time (T_r)	0.813	1.03	4.82	4.82
Steady state time (T_{ss})	4.5	4.5	15	15
Final Value (FV)	1	1	1	1

Table 6.9. comparison of time domain characteristics of 1st order Reduced models of problem 1 Vs Given System.

Tables 6.7., 6.8., 6.9., brings overall comparison of time domain characteristics of all reduced models in all different algorithms proposed in this thesis with respect to their order of reduced model. Among comparisons, the completion is between Algorithm 1 and algorithm3.redduced models obtained from algorithm 2 characteristics are not proper representation of original system. Among all reduced models obtained from three different algorithms algorithm 3 represents best compared with algorithm 1.Third and second order models from algorithm 1 and algorithm 2 are representing good models. But none of the first order models from any algorithm can represent the original system, because first order system cannot obtain overshoot characteristics of under damped systems. Given system in problem 1 consists of overshoot characteristics, so first order systems cannot applicable here. Overall, by increasing the order of the reduced model, then the probability of representing and matching of characteristics that of original system can be made higher.

5.3.2. PROBLEM 2

Characteristics	5 th order Given system	4 th order Reduced model (Algorithm1)	4 th order Reduced model (Algorithm3)
Peak Magnitude (M_p)	1.26	1.27	1.22
%Overshoot ($\%M_p$)	26	26.9	21.7
%Overshoot Time(T_{Mp})	1.71	1.65	2.55
Settling Time (T_s)	4.94	4.91	5.67
Rise Time T_r)	0.71	0.695	1.03
Steady state time (T_{ss})	9	9	9
Final Value(FV)	1	1	1

Table 6.10. Comparison of time domain characteristics of 4th order Reduced models of problem 2 Vs Given System

Characteristics	5 th order Given system	3 rd order Reduced model (Algorithm1)	3 rd order Reduced model (Algorithm 2)	3 rd order Reduced model (Algorithm 3)
Peak Magnitude (M_p)	1.26	1.25	1.34	1.25
%Overshoot ($\%M_p$)	26	25.4	33.6	25.4
%Overshoot time(T_{Mp})	1.71	1.68	3.95	1.68
Settling Time (T_s)	4.94	4.94	13.4	4.94
Rise Time (T_r)	0.71	0.705	1.83	0.705
Steady state time (T_{ss})	9	8	20	9
Final Value (FV)	1	1	1	1

Table 6.11. Comparison of time domain characteristics of 3rd order Reduced models of problem 2 Vs Given System

Characteristics	5 th order Given system	2 nd order Reduced model (Algorithm1)	2 nd order Reduced model (Algorithm 2)	2 nd order Reduced model (Algorithm 3)
Peak Magnitude (M_p)	1.26	1.17	1.14	1.28
%Overshoot ($\%M_p$)	26	16.7	13.6	26.1
%Overshoot time(T_{Mp})	1.71	1.44	3.61	1.26
Settling Time (T_s)	4.94	3.11	6.52	3.06
Rise Time (T_r)	0.71	0.577	1.63	0.427
Steady state time (T_{ss})	9	5	12	9
Final Value (FV)	1	1	1	1

Table 6.12. Comparison of time domain characteristics of 2nd order Reduced models of problem 2 Vs Given System

Characteristics	5 th order Given system	1 st order Reduced model (Algorithm1)	1 st order Reduced model (Algorithm 2)	1 st order Reduced model (Algorithm 3)
Peak Magnitude (M_p)	1.26	-	-	-
%Overshoot ($\%M_p$)	26	-	-	-
%Overshoot time(T_{Mp})	1.71	-	-	-
Settling Time(T_s)	4.94	1.73	6.61	1.73
Rise Time (T_r)	0.71	0.972	3.71	0.972
Steady state time (T_{ss})	9	2.5	10	9
Final Value (FV)	1	1	1	1

Table 6.13. Comparison of time domain characteristics of 1st order Reduced models of problem 2 Vs Given System.

Tables 6.10., 6.11., 6.12., 6.13, brings overall comparison of time domain characteristics of all reduced models in all different algorithms proposed in this thesis with respect to their order of reduced model. Reduced models obtained from algorithm 2 characteristics are not proper representation of original system. Among all reduced models obtained from three different algorithms algorithm 3 represents best compared with algorithm 1. Fourth, third, second order models from algorithm 1 and algorithm 2 are representing good models for problem 2. But none of the first order models from any algorithm can represent the original system, because first order system cannot obtain the overshoot characteristics of under damped systems. Given system in problem 2 consists of overshoot characteristics, so first order systems cannot applicable here. Overall, from the above discussion, increasing the order of the reduced model, then the probability of representing and matching of characteristics that of original system can be made higher.

Chapter 6

CONCLUSION and FUTURE SCOPE

6.1 CONCLUSION

In this thesis, three different algorithms are explained in detail to reduce given higher order transfer function to find the stable reduced model transfer function for single input and single output systems. Algorithm 1, Markov and h-parameters matching give good approximated reduced model to represent given system as compared with Algorithm 2, Genetic algorithms (GA). Algorithm 3, combination both Markov and H-parameter matching and Genetic algorithms (GA) give better approximated reduced model as compared with Algorithm 1.

Two common numerical examples are taken and tried to find the best suitable reduced order models by using three algorithms. Results are shown in chapter 5 and tabulated the specification of time domain properties of given system and reduced model, and compare the results. All methods are using common philosophy of step response matching. Algorithm 2, Algorithm 3 were tried with uniform sampling rate for order reduction of discrete time systems. Algorithm 1 does not need any samples of either given system or reduced system.

As we are using Genetic algorithms in Algorithm 2, Algorithm 3 the obtained models are very sensitive to parameter of genetic algorithm properties such as method of selection, type of mutation and crossover, distribution of population, etc. Genetic algorithms are parallel search optimization method, so each time it starts with different starting point and ends with different one. Algorithm 1, Markov and H-parameter matching requires a lot of computations and complexity while finding the stable reduce model.

6.2. SCOPE OF FUTURE WORK

Development of three algorithms of order reduction and its applications in designing a controller, leads to following possibilities, which can be tried in future.

- The choice of constant sampling rate can be replaced with varying sampling rate for trying these ideas as a faster sampling rate can give better result in transient part of the response and slower rate may be not enough.
- The design of a controller using reduced order model can be examined more rigorously by comparing with other existing techniques.
- This work can be improved over for MIMO systems.
- Case study of Genetic Algorithms with other conventional methods for giving still better approximated reduced models may be possible.

Motivation for working with discrete time systems technology leads to a radically new approach in control system design. It will be practicable to design a controller for very complicated digital systems using reduced order model.

REFERENCES

- [1].WESTCOTT, J.H., "The frequency response method: its relationship to transient behavior in control system design", Trans. on Instrum. Tech., vol.4, pp. 113-124, 1952.
- [2].SHAMASH.Y, "Critical review of methods for deriving stable reduced order models" Proc. 6th. IF AC Symposium on identification and system parameter Estimation, Washington DC, pp.1355-1359, June 1982.
- [3].HUTTON, M. F., and FRIEDLAND.B, "Routh approximations for reducing the order of linear time-invariant systems", IEEE Trans., AC20, pp. 329-337, 1975.
- [4].SHAMASH. Y, "Model reduction using the Routh stability criterion and Pade approximation technique", Int. J. Control, vol. 21, pp.475-484, 1975,
- [5].CHEN, T. C., CHANG, C. Y., and HAN, K.W., "Model reduction using the stability equation method and the continued fraction method", Int. J. Control, vol.32, pp. 81-94, 1980.
- [6].PAL.J, "Stable reduced order approximants using the Routh- Hurwitz array", Electron. Lett, IS, pp. 225-226, 1979.
- [7].SHAMASH.Y, "Stable biased reduced order models using the Routh method of reduction", Int. J. Syst. Sci., pp. 641-654, 1980.
- [8].SINGH.V. , "Stable approximants for stable systems", Proc. IEEE, vol. 69, pp. 1155-1156, 1981.
- [9].ASHOOR.N, and SINGH.V, "A note on low order modeling", IEEE Trans. AC-27, pp. t 124-1126, 1982.
- [10]. SHAMASH.Y. "Continued fraction methods for the reduction of discrete-time dynamic systems", Int. J. Control. vol.20, pp. 267-275, 1974.
- [11].MOORE, B.C., "Principal component analysis in linear systems: controllability, observability and model reduction", IEEE Trans., AC-26, pp. 17-32, 1981.
- [12].K.WARWICK, "A new approach to reduced order modeling", IEE proc., vol.131, pp. 74-79, mar 1984.
- [13].JAMSHIDI.M, "large scale systems modeling and control", series vol.9, north Holland, amsterdam oxford 1983.

- [14].GLOVER.K, "All optimal hankel norm approximations of linear multivariable systems and their L_{∞} error bounds", Int. J.Control, vol. 39, pp. 1115-1193, 1984.
- [15].SKELTON, R.E., and ANDERSON, "Q-Markov covariance equivalent realizations", Int. J. Control, vol.44, pp. 1477-1490, 1986.
- [16].MAHMOUD, M.S. and SINGH, M.G., "Large scale systems modeling", permon press, Oxford 1981.
- [17].HALEVI.Y, "Optimal reduced-order models with delay". Proc. of the 30th conference on Decision and Control, Brighton, pp. 602-607, 1991.
- [18].LILLY, and J.H., "Efficient DFT based model reduction for continuous systems", IEEE Trans., AC-36, pp. 1188-1193, 1991.
- [19].SPANOS, J.T., MILMAN, M.H., and MINGORI, D.L., "A new algorithm for L2 optimal model reduction", IEE Trans., vol.28, pp. 897-909, 1992.
- [20].GONG, M., MURRAY and SMITH, "Model reduction by an extended complex curve fitting approach", IEEE Trans. Inst. Meas.Control, vol.15, pp. 188-198, 1993.
- [21].GRUCA, A., and BERTRAND.P, "Approximation of high order systems by low-order models with delays", Int. Journal on Control, vol.28, pp. 953-965, 1978.
- [22].ZHENG, W.X. and FENG.C, "Hybrid method for model reduction with time delay", Int. J. Systems Sci., vol. 21, pp. 755-763, 1990.
- [23].XUE.D, ATHERTON, and D.P. "A suboptimal reduction algorithm for linear systems with a time delay", Int. Journal on Control, vol.60, pp. 181-196, 1994.
- [24].GOLDBERG, and D.E., "Genetic algorithms in search, optimization, and machine learning." (Addison-Wesley, 1989)
- [25].KRISTINSSON.K, and DUMONT.G, "System identification and control using genetic algorithms', IEEE Transactions, SMC-22, pp.1033-1046, 1992.
- [26].LANSBERRY, J.E., WOZNIAK.E, and GOLDBERG, "Optimal hydro generator governor tuning with a genetic algorithm", IEEE Trans., vol.7, pp. 623-630, 1992.

- [27].PORTER.B, JONES, and A.H., "Genetic tuning of digital PID controllers", *Electron. Lett.* , vol.28, pp. 843-844, 1992.
- [28].CAPONETTO, R., FORTUNA, L., GRAZIANI, S., XIBILIA, and M.G. "Genetic algorithms and applications in system engineering a survey", *Trans. Inst. Meas. Control*, vol.15, pp.143-156, 1993.
- [29].YANG, Z.J., HACHINO.T, TSUJI.T, and SAGARA.S, "Identification of parameters and time delay of continuous systems using the genetic algorithm", 10th Symposium on Identification and System Parameter Estimation, Copenhagen, Vol. 3, pp. 657-662, 1994.
- [30].Z.J.YANG, T.HACHINO, T.TSUJI, "Model Reductions with time delay combining the least squares method with genetic algorithm", *IEE Proc. on control theory and applications*, Vol. 143, No.3, may 1996.