Model Order Reduction for Nonlinear Systems

by

Yong Chen

Submitted to the Department of Mathematics
in partial fulfillment of the requirements for the degree of

Master of Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 1999

Author . . . .
Department of Mathematics
July 30, 1999

Certified by . . . .
Jacob White
Professor of Electrical Engineering and Computer Science
Thesis Supervisor

Accepted by . . . .
Tomasz Mrowka
Chairman, Department Committee on Graduate Studies

Model Order Reduction for Nonlinear Systems

by

Yong Chen

Submitted to the Department of Mathematics
on July 30, 1999, in partial fulfillment of the
requirements for the degree of
Master of Science

Abstract

This thesis presents some practical methods for doing model order reduction for a general type of nonlinear systems. Based on quadratic or even higher degree approximation and tensor reduction with assistance of Arnoldi type projection, we demonstrate a much better accuracy for the reduced nonlinear system to capture the original behavior than the traditional linearization method.

Thesis Supervisor: Jacob White
Title: Professor of Electrical Engineering and Computer Science

Acknowledgments

I first want to thank Professor Jacob White, the supervisor of this thesis, for his very helpful advice and guidance crucial in many stages of the research done in the thesis and a lot of academic advice and also his summer financial support. And I learned most of my numerical knowledge and intuition from his 6.336 class. I also want to thank the Department of Mathematics for the academic and financial support and especially our Graduate Coordinator Linda Okun who gave much help for the administrational procedure. I became a fan for tensors (which is widely used in the main chapters of the thesis) in the multilinear algebra class of my highly respected late advisor in math department Gian-Carlo Rota, whose influence to me is deep and enormous. Frank Wang of MIT gave me some kind technical help and Junling Ma of Princeton helped me with a question on ODE. And I also want to thank several colleagues in the RLE-VLSI group who offered help with computers. The weekly group lunch which further nourished my time there should also be mentioned.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 What is Model Order Reduction

Model order reduction is a very attractive idea in CAD area. It replaces the original large scale system model with a much smaller one, yet still retains the original behavior under investigation to high accuracy. Therefore,by simulating just the reduced small system one can still study the original system and thus make the design work much easier. With the ever increasing scale of system models appearing in the engineering design practice, model order reduction has become an indispensable tool in numerous areas such as circuit interconnect and MEMS(micro-electrical mechanical system), though the idea of model order reduction is general enough to be applicable to other areas. Model order reduction is also a very interesting and meaningful mathematical problem in its own right.

The following example, albeit linear, can easily demonstrate the meaning of model order reduction. Suppose we are given a system of equations

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{b}u(t) \tag{1.1}$$

$$y = \mathbf{c}^{\mathbf{T}}\mathbf{x} \tag{1.2}$$

This system of equations may be the mathematical model for a realistic system involving with specified input and output. Here $\mathbf{x} = \mathbf{x}(t)$ is a N-dimensional vector(variable) , $\mathbf{A}$ is an N by N (constant) matrix and $\mathbf{b}$ and $\mathbf{c}$ are N-dimensional (constant) vectors Note we will generally use bold lower case letters for vectors and bold upper case letters for matrices and tensors in the thesis. Here $u = u(t)$ is the input, $y = y(t)$ is the output. Usually,the initial value of $\mathbf{x} = \mathbf{x}(t)$ is zero ($\mathbf{x}(0) = \mathbf{0}$) unless otherwise stated.

This example is a SISO(single input-single output) example, which is the only case that we will consider in the thesis , but the approaches can can be generalized to the vector input-vector output case. In (1.1) $\mathbf{x}$ is often referred to as "state variables" and the dimension N of the original system (1.1) and (1.2) is typically much larger than 100.

Model order reduction then asks the following question: can we find another system

$$\dot{\mathbf{z}} = \mathbf{A}_r\mathbf{z} + \mathbf{b}_r u(t) \tag{1.3}$$

$$y_r = \mathbf{c}_r^T\mathbf{z} \tag{1.4}$$

where the state variables z is of dimension n, $\mathbf{A}_r$ is n by n matrix, $\mathbf{b}_r$and $\mathbf{c}_r$ are n-dimensional vectors and n is much smaller than N. For example, N could be in the order of thousands and n might be less than 10. And the "reduced" system has very close (input-output) behavior with the original system, that is, given input $u(t)$, the reduced system(equations 1.3 and 1.4) will generate an output $y_r(t)$ which is very close to $y(t)$. In more mathematical language, we can define for any system S an input-output operator or map L_S which maps any input $u(t)$ to the output $y(t)$ generated by the system, and we would require that for the reduced system Sr, we should have L_{Sr} close to L_S in certain sense (and in this thesis, we by default use the "point-wise" sense stated above that is for (any allowed) same input $u(t)$,$L_{Sr}[u(t)]$ should be close to $L_S[u(t)]$. Thus once we have the reduced system, we can study the interested behavior and property of the original system to a good precision by just studying or simulating the reduced system instead of the much larger original system. And people already have very good and mature methods to perform model order reduction on linear systems, as will be outlined in Chapter 3.

1.2 Nonlinear Model Order Reduction

In many engineering situations however, people have to deal with nonlinear models and systems. And we can ask the same question of doing model order reduction for nonlinear systems. For example a mostly encountered type (other types can often be transformed to or approximated by this representative type) is to try to reduce(with the meaning explained in the above section)

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{b}u(t) \tag{1.5}$$

$$y = \mathbf{c}^{\mathbf{T}}\mathbf{x} \tag{1.6}$$

to

$$\dot{\mathbf{z}} = \mathbf{g}(\mathbf{z}) + \mathbf{b}_r u(t) \tag{1.7}$$

$$y_r = \mathbf{c}_r^T \mathbf{z} \tag{1.8}$$

where $\mathbf{x} \in R^N$ and $\mathbf{z} \in R^n$ with $n \ll N$ as usual and both $\mathbf{f}$ and $\mathbf{g}$ are nonlinear (vector)functions (by default all the functions appearing in the thesis are assumed to have reasonably good properties such as infinite-times differentiable) .

Nonlinear model order reduction is very much desired for doing large scale CAD and simulation in many areas involving nonlinear systems but there have not been satisfactory and universally usable methods to do this. A mostly used traditional strategy is to first linearize the system then perform model order reduction on linear system; but the linearization does not always give good approximation to the original nonlinear system. This thesis will present some methods to reduce nonlinear systems (to smaller *nonlinear* systems) which are observed to offer much better approximations to the original systems compared with the traditional linearization methods as we keep the nonlinearality and which are applicable to general nonlinear systems usually encountered. These methods are based on higher degree approximation (especially quadratic) of the nonlinear systems.

1.3 Thesis Outline

In what follows,Chapter 2 studies the linear, quadratic and higher degree approximations of the nonlinear system and demonstrate their accuracies through case studies. This chapter therefore sets the goal for what the the reduced order models should match and their potential accuracies. Chapter 3 gives a self-contained review of one very successful method for linear model order reduction—the Arnoldi method. Besides being an easy and practical method for reducing those quasi-linear systems which is close to its linear approximations, some essential aspects of the method will also be important components of the nonlinear model order reduction methods presented later. Chapters 4 is the main content of the thesis. In Chapter 4 we present the "quadratic reduction" method for reducing nonlinear systems which aims to reduce the quadratic approximation of the original system (which in turn is shown earlier in Chapter 2 to usually give a quite good approximation to the nonlinear system) to a smaller quadratic system by both reducing the linear part by the traditional Arnoldi procedure and reducing the quadratic part(Hesse tensor) by applying the Arnoldi projection of state variables. We employ tensorial and Matlab notations to help express the algorithm to reduce the quadratic/Hesse tensor. The method is demonstrated to work satisfactorily on numerical verifications. We also give alternative but equivalent versions of the quadratic reduction method with no need to first get the quadratic approximation of the original system. In Chapter 5 we presented some further methods for doing nonlinear model order reductions, mainly the third order reduction method for even higher precision requirement. We also briefly describe the generalization to arbitrary degree approximation (which is not usually needed as we believe) and some other possible methods. Chapter 6 draws the conclusions.

Chapter 2

Nonlinear Systems and Their Approximation

2.1 A Typical Form of Nonlinear System

The nonlinear systems (with single input and output) that we are concerned with in this thesis are of the following form already encountered in the introductory chapter:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{b}u(t) \tag{2.1}$$

$$y = \mathbf{c}^{\mathbf{T}}\mathbf{x} \tag{2.2}$$

Here we can call the first equation "state equation" and the second one "output equation" and we will refer this type of nonlinear systems as of "standard form" in the thesis.

This type of nonlinear systems (with the nonlinearality focused on the state variables and with linear input and output appearance) is encountered often in the practical situations and we actually have not much loss of generality by restricting our considerations to it with the following two remarks:

Remark 1: When there is a nonlinear dependence on the input $u(t)$ in the state equation too it is often in the following form:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{b}p(u(t))$$

where $p(u)$ is a nonlinear function on u. In this case the nonlinear dependence on input can often be bypassed with the trick commonly used in practical simulation by treating the whole term $p(u(t))$ as the input, thus transforming the state equation to

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{b}v(t)$$

where $v = p(u)$. Even for the most general case where the state equation reads

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \beta\mathbf{h}(u(t))$$

where $\mathbf{h}(u) = (h_1(u), h_2(u), \cdots, h_N(u))^T$ is some nonlinear vector function and β is some constant(can be absorbed into $\mathbf{h}$ if needed); we can still similarly set $\mathbf{v}(t) = \mathbf{h}(u(t))$ as the new (vector) input and $\mathbf{B} = \beta\mathbf{I}_N$ then transforming the state equation to

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{B} \cdot \mathbf{v}(t)$$

with the multi-input $\mathbf{v} = (v_1, v_2, \cdots, v_n)^T = (h_1(u(t)), h_2((u(t)), \cdots, h_N(u(t)))^T$.

Remark.2: Although for convenience we will develop our nonlinear model order reduction methods for the standard type of system above, it will be seen in Chapter 4 that our main method which is based on projecting high dimensional state variables onto low dimensional state variables is also easily extended to the case where the output is nonlinearly depending on the state variables($y = g(\mathbf{u})$ where g is a nonlinear function). Besides this, for some systems with nonlinear output equations we may also use some tricks to transform the system to the stand form. As an simple illustrating example, for the system

$$\dot{x}_1 = e^{x_1} + x_2^2 + u(t)$$

$$\begin{aligned}\dot{x}_2 &= x_1 + \sin(x_2)\\ y &= x_1 + 2x_2 + x_2^2\end{aligned}$$

we can introduce a new state variable $x_3 = x_2^2$ thus $\dot{x}_3 = 2\dot{x}_2 x_2$ (with the initial value $x_3(0)$ still being 0) and therefore we can transform the original system to

$$\begin{aligned}\dot{x}_1 &= e^{x_1} + x_2^2 + u(t)\\ \dot{x}_2 &= x_1 + \sin(x_2)\\ \dot{x}_3 &= 2x_1x_2 + 2x_2\sin(x_2)\\ y &= x_1 + 2x_2 + x_3\end{aligned}$$

which is of the standard form. (This sort of trick will work especially fine when there are not too many auxiliary state variables introduced; although we increase the system scale a bit at the transformation stage, the model order reduction step can usually reduce the scale tremendously to generate a reduced order model with scale still much smaller than the original one)

2.2 Approximations to Nonlinear System

All the nonlinear model order reduction methods presented in this thesis is based on the idea of approximating the nonlinear system by a polynomial system through the Taylor expansion (to certain degree, most easily two) which approximate the original system to a satisfactory precision then trying to generate a reduced polynomial system for it which matches it to high precision and in turn matches the original system to satisfactory precision. The effectiveness and error of the nonlinear model order reduction also come from the corresponding two pieces with the title of this section being the first piece.

We observe that for the nonlinear system 1.5 and 1.6,

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{b}u(t)$$

$$
y = \mathbf{c}^{\mathbf{T}}\mathbf{x}
$$

if vector function $\tilde{\mathbf{f}}$ approximates $\mathbf{f}$ then the corresponding approximation system

$$
\begin{aligned}
\dot{\tilde{\mathbf{x}}} &= \tilde{\mathbf{f}}(\tilde{\mathbf{x}}) + \mathbf{b}u(t) \\
\tilde{y} &= \mathbf{c}^{\mathbf{T}}\tilde{\mathbf{x}}
\end{aligned}
$$

will generate an output $\tilde{y}$ also approximating y, *independent to (any) input u.* Actually we have

$$
\frac{d}{dt}(\mathbf{x} - \tilde{\mathbf{x}})(t) = \mathbf{f}(\mathbf{x}) - \tilde{\mathbf{f}}(\tilde{\mathbf{x}}) = \mathbf{f}(\mathbf{x}) - \mathbf{f}(\tilde{\mathbf{x}}) + \mathbf{f}(\tilde{\mathbf{x}}) - \tilde{\mathbf{f}}(\tilde{\mathbf{x}})
$$

and as usual we assume we are working on a bounded domain and $\tilde{\mathbf{f}}$ approximates $\mathbf{f}$ with uniform error ϵ then on any time range $[0, T]$ we will have

$$
\begin{aligned}
|\frac{d}{dt}(\mathbf{x}(t) - \tilde{\mathbf{x}}(t))| &\leq |\mathbf{f}(\mathbf{x}) - \mathbf{f}(\tilde{\mathbf{x}})| + |\mathbf{f}(\tilde{\mathbf{x}}) - \tilde{\mathbf{f}}(\tilde{\mathbf{x}})| \\
&\leq L|\mathbf{x} - \tilde{\mathbf{x}}| + \epsilon
\end{aligned}
$$

where L is some positive constant (say the maximum absolute values of all the first order derivatives of $\mathbf{f}$). Therefore

$$
\begin{aligned}
\frac{d}{dt}|\mathbf{x}(t) - \tilde{\mathbf{x}}(t)|^2 &= \frac{d}{dt}(\mathbf{x}(t) - \tilde{\mathbf{x}}(t))^2 \\
&= 2(\mathbf{x}(t) - \tilde{\mathbf{x}}(t)) \cdot \frac{d}{dt}(\mathbf{x}(t) - \tilde{\mathbf{x}}(t)) \\
&\leq 2|\mathbf{x}(t) - \tilde{\mathbf{x}}(t)||\frac{d}{dt}(\mathbf{x}(t) - \tilde{\mathbf{x}}(t))| \\
&\leq 2L|\mathbf{x} - \tilde{\mathbf{x}}|^2 + 2\epsilon|\mathbf{x} - \tilde{\mathbf{x}}| \\
&\leq 2L|\mathbf{x} - \tilde{\mathbf{x}}|^2 + 2M\epsilon
\end{aligned}
$$

where M is some bound and then we have

$$
\frac{d}{dt}\{e^{-2Lt}|\mathbf{x}(t) - \tilde{\mathbf{x}}(t)|^2\} = \{\frac{d}{dt}|\mathbf{x}(t) - \tilde{\mathbf{x}}(t)|^2 - 2L|\mathbf{x}(t) - \tilde{\mathbf{x}}(t)|\}e^{-2Lt} \leq 2M\epsilon e^{-2Lt}
$$

integrating from 0 to t gives

$$|\mathbf{x}(t) - \tilde{\mathbf{x}}(t)|^2 \leq \frac{M(e^{2LT} - 1)}{L}\epsilon$$

thus $|\mathbf{x} - \tilde{\mathbf{x}}|$ can be made arbitrarily small with ϵ arbitrarily small, and independent of input u , so does $|y - \tilde{y}|$ which is bounded by $\|\mathbf{c}\||\mathbf{x} - \tilde{\mathbf{x}}|$. In other words, the IO map(input-output map, introduced in Chapter 1) of the approximation system is uniformly approximating that of the original one.

Traditionally people often simply use the linear approximation to linearize the original nonlinear system, which works well for those systems operating in so called linear regime (usually in a small departure from equilibrium), but works poor for general cases as observed in many examples such as the MEM system reported in [1, 2]. Usually the quadratic approximation which is mostly used in this thesis offers much better and satisfactory accuracy;in case of even higher accuracy requirement third order or even higher approximations may be needed. The next section will give a typical nonlinear system which is from nonlinear circuits and yet relatively easy to implement, together with its approximations of various orders. This typical example will recur many times in the thesis to illustrate our ideas and verify our methods or results.

2.3 A Nonlinear Circuit Example

Consider the following circuit RC structure in Fig. 2-1 where the resistors there are nonlinear resistors, whose currents depending nonlinearly on the potential difference across the resistor. We clearly label two ends of the resistors by “a” and “b” to specify the their orientations because nonlinear resistors can be asymmetric, i.e., when one flips the voltage the current may not get flipped in exactly value such as the case for dianodes whose I-V dependence are highly asymmetric. And for simplicity we take all the resistors to be the same and have the I-V dependence profile given by some function g:when the potential difference from a to b through the resistor is V the

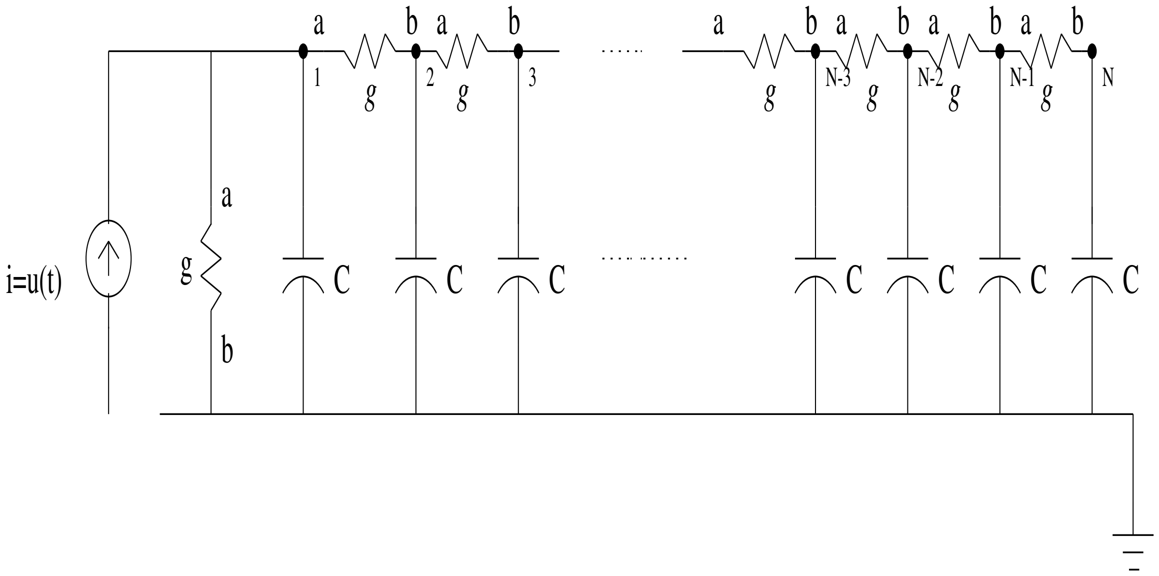

Figure 2-1: A nonlinear circuit structure

current flowing from a to b will be $I = g(V)$. If one reverses the orientation of the resistor the I-V profile will simply be $I = -g(-V)$. There is one input current source $i = u(t)$ flowing into node 1. And our output is set to be the potential at node 1 with the state variables being the potentials at nodes $1, \cdots, N$.

Looking at the currents at each node we can write down the following set of equations:

$$\begin{aligned}
u(t) &= C\frac{dv_1}{dt} + g(v_1 - v_2) + g(v_1) \\
g(v_1 - v_2) &= C\frac{dv_2}{dt} + g(v_2 - v_3) \\
&\vdots \\
g(v_{k-1} - v_k) &= C\frac{dv_k}{dt} + g(v_k - v_{k+1}) \\
&\vdots \\
g(v_{N-1} - v_N) &= C\frac{dv_N}{dt}
\end{aligned}$$

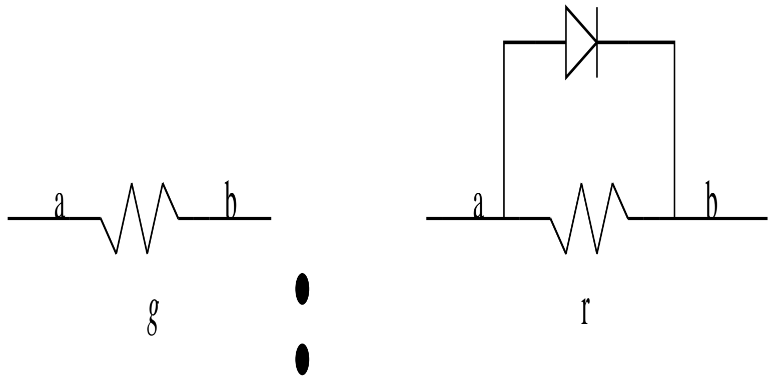

Figure 2-2: A dianode paralleled with a normal resistor

which can be arranged to our standard form as

$$C\frac{d\mathbf{v}}{dt} = \begin{bmatrix} -g(v_1) - g(v_1 - v_2) \\ g(v_1 - v_2) - g(v_2 - v_3) \\ \vdots \\ g(v_{k-1} - v_k) - g(v_k - v_{k+1}) \\ \vdots \\ g(v_{N-1} - v_N) \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} u(t) \tag{2.3}$$

And the linear and quadratic approximation of the system are simply obtained by replace the function g above by its linear and quadratic approximation in its Taylor expansion respectively, also similar for higher order approximations.

To perform real computation we choose the size of the system to be 100 ($N = 100$) and we pick a nonlinear resistor profile g to be

$$g(v) = \exp(40v) + v - 1 \tag{2.4}$$

that is, we let all the nonlinear resistors appearing in Fig. 2-1 to be consists of a dianode with IV profile $g_d(v) = \exp(40v) - 1$ paralleled with a normal (linear) resistor which has unit resistance with profile $g_r(v) = v$ (see Fig. 2-2) and the corresponding whole circuit shown in more detail in Fig. 2-3.

And for simplicity we let all the capacitors in the example have unit capacitance

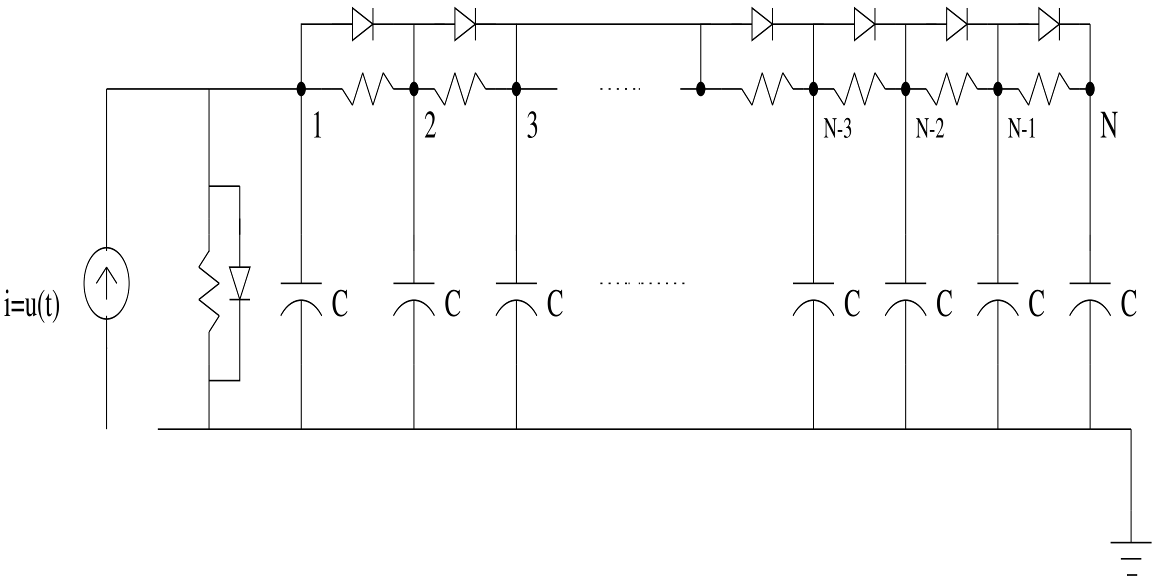

Figure 2-3: The nonlinear circuit example

$(C = 1)$. Now the original nonlinear system equations for this example is

$$\frac{d\mathbf{v}}{dt} = \begin{bmatrix} -g(v_1) - g(v_1 - v_2) \\ g(v_1 - v_2) - g(v_2 - v_3) \\ \vdots \\ g(v_{k-1} - v_k) - g(v_k - v_{k+1}) \\ \vdots \\ g(v_{N-1} - v_N) \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} u(t) \tag{2.5}$$

with $g(v) = \exp(40v) + v - 1$

For this example we will investigate its approximation up to 3rd order. The linear approximation of this $g(v)$ is

$$g_1(v) = 40v + v = 41v$$

the quadratic approximation is

$$g_2(v) = 41v + (40v)^2/2 = 41v + 800v^2$$

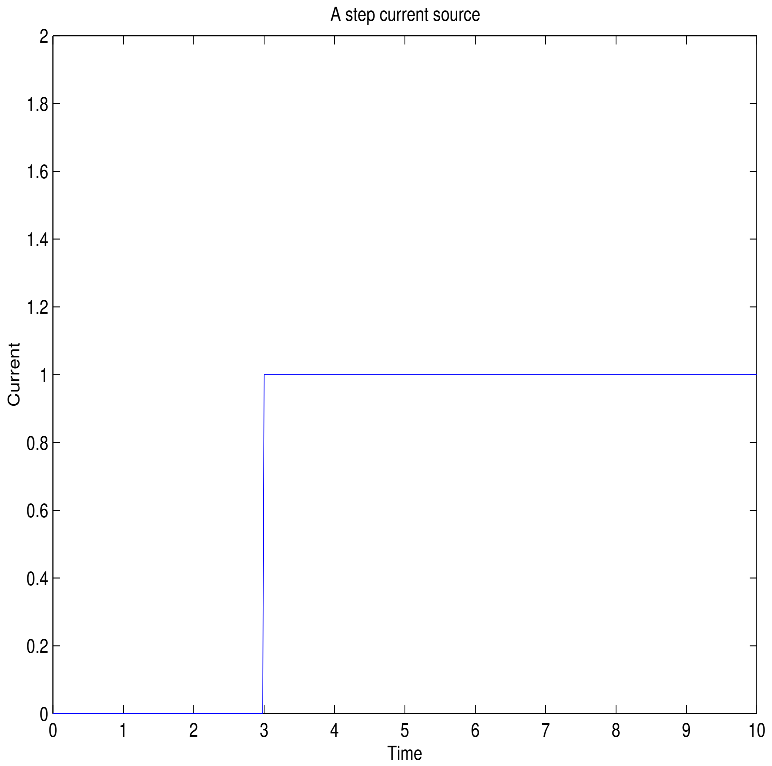

Figure 2-4: The step current source in the example

and the 3rd order approximation is

$$g_3(v) = 41v + 800v^2 + (40V)^3/(3!) = 41v + 800v^2 + (32000/3)v^3$$

Then the linear,quadratic and 3rd order approximations of system (2.5) are obtained by replacing the g there by g_1, g_2, g_3 respectively.

Now we pick a specific current source $i = u(t)$ to be a step source from 0 to 1 which is turned on at time $t = 3$ (see Fig. 2-4).

Step sources are one of the most commonly used source in this type of simulations to study the behavior and performance of the original and approximation systems. It is easy to represent and it will also reveal some of the most important feature of the system such as the steady state and how the system approach to it. With step source we can judge the accuracy of an approximation or reduced system to match the original system through comparing its steady state and transition behavior, which

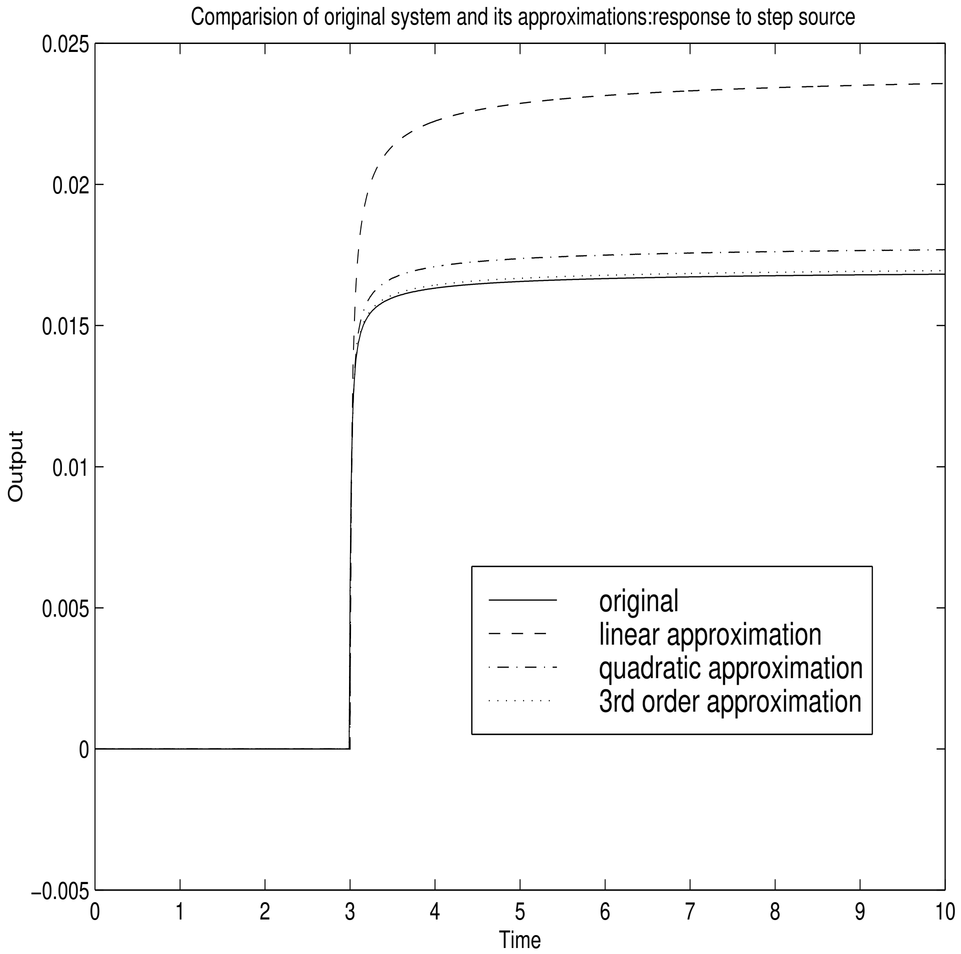

Figure 2-5: Comparison of the original system in the example and its linear,quadratic and 3rd order approximations

are our too important criteria, with original ones and see how well they capture or match them.

We proceed to integrate[1] the ODEs in the (2.5) with the g substituted by (2.4) and its various approximations g_1, g_2, g_3 and compute the corresponding outputs for the original and approximations systems with the results given in Fig. 2-5.

We see for this nonlinear system the linear approximation departures seriously from the original system, therefore even the traditional linear model order reduction can generate a reduced order linear system which matches this ($N = 20$)linear

[1]Backward Euler method are often used when integrating ODES which have steep transition because forward Euler method can break down in this case.

approximation exactly, it would still be quite far from the original nonlinear system. We also see that the quadratic approximation(which now retains some nonlinearality) gives a much better (about 6 times closer) approximation to the original system so a good reduced order quadratic system generated out of it (which is our goal in Chapter chap:quad) would also be a good approximation for the original system. And we further see that the 3rd order approximation gives really high accuracy as expected, though it may be more time consuming to deal with in computation.

Chapter 3

Linear System Reduction

3.1 Model Order Reduction for Linear Systems

We see in Chapter 2 that the linearization method will not generally give a good approximation for nonlinear system, but reducing a linear system is a subject in its own right and particularly the methods in reducing linear systems also give us helpful hints in developing methods for reducing nonlinear systems; therefore in this section we will give a self-contained review of model order reduction for linear systems as a preparation and also useful tool for later nonlinear model order reductions.

There are actually several methods to do model order reduction for linear systems(see for example [3]) such as slowest modes, explicit moment matching, Arnoldi method, Lanczos method (these two also belong to "implicit moment matching") and truncated linear system etc. Among them the Arnoldi method is one of the most popular and successful method for its effectiveness and ease to use and it is this method that will be reviewed here and also be useful in later part of the thesis.

For a linear system[1] given as in (1.1,1.2)

$$\dot{\mathbf{x}} \quad = \quad \mathbf{A}\mathbf{x} + \mathbf{b}u(t)$$

[1]We will only deal with *nondegenerate* systems, i.e, we always assume A to be nonsingular and for nonlinear state equation (1.5) we assume the Jacobian of $\mathbf{f}$ to be nonsingular too. These requirements are almost always satisfied in realistic situations

$$y = \mathbf{c^T x}$$

we can perform Laplace transform on the equations to get

$$sX(s) = \mathbf{A}X(s) + \mathbf{b}U(s)$$

and

$$Y(s) = \mathbf{c}^T X(s)$$

where we make use of the fact that $\mathbf{x}(0) = 0$ and we now have the Laplace transform of output linked with that the input simply as

$$Y(s) = G(s)U(s) \tag{3.1}$$

where

$$G(s) = \mathbf{c}^T(s\mathbf{I} - \mathbf{A})^{-1}\mathbf{b} = -\mathbf{c}^T\mathbf{A}^{-1}(\mathbf{I} - s\mathbf{A}^{-1})^{-1}\mathbf{b} \tag{3.2}$$

is the all important *transfer function* of the linear system (1.1)-(1.2) and we can expand $G(s)$ as

$$G(s) = -\mathbf{c}^T\mathbf{A}^{-1}(\mathbf{I} + s\mathbf{A}^{-1} + s^2\mathbf{A}^{-2} + ...)\mathbf{b} = -\sum_{i=0}^{\infty} m_i s^i \tag{3.3}$$

where

$$m_i = \mathbf{c}^T\mathbf{A}^{-(i+1)}\mathbf{b} = \mathbf{c}^T\mathbf{A}^{-i}(\mathbf{A}^{-1}\mathbf{b}) \tag{3.4}$$

is called the ith moment($i = 0, 1, \cdots$). And Arnoldi method (as well as many other linear model order reduction methods) is based on the idea to try to come up with a reduced system (1.3)(1.4)

$$\begin{aligned} \dot{\mathbf{z}} &= \mathbf{A}_r\mathbf{z} + \mathbf{b}_r u(t) \\ y_r &= \mathbf{c}_r^T\mathbf{z} \end{aligned}$$

whose transfer function $G_r(s) = \mathbf{c}_r^T(s\mathbf{I} - \mathbf{A}_r)^{-1}\mathbf{b}_r$ will have the same moments with

$G(s)$ up to some degree:

$$\mathbf{c}_r^T \mathbf{A}_r^{-(i+1)} \mathbf{b}_r = \mathbf{c}^T \mathbf{A}^{-(i+1)} \mathbf{b}, i = 0, 1, \cdots, k \tag{3.5}$$

where k usually depends on the size of the reduced system (it is of the order of the size of the reduced system in the Arnoldi case). Thus the reduced linear system approximates the original linear system (independent to the input) because its transfer function (independent of u) approximates the transfer function of the original linear system.

The Arnoldi method is based on the following "Arnoldi algorithm" ([4]) which has 3 inputs[2]:

$\mathbf{A}$, an N by N matrix; $\mathbf{b} \in R^N$; and an integer q, which will be the size chosen for the reduced system and usually $q \ll N$

and will generate 3 outputs:

$\mathbf{V}$, an N by q matrix; $\mathbf{H}$, an q by q matrix; and $\mathbf{v}_{q+1} \in R^q$ which will satisfy the following 3 properties:

Property 1.$\mathbf{V}$ is an column-orthonormal matrix: its q columns form a set of orthonormal vectors in R^N;

Property 2.

$$\mathbf{AV} = \mathbf{VH} + h\mathbf{v}_{q+1}\mathbf{e}_q^T \tag{3.6}$$

where h is a scalar and the $\mathbf{e}_q$ is the qth standard unit vector in R^q (unless stated otherwise all standard units vectors $\mathbf{e}_i$ appearing below will be in R^q by default)

Property 3.

$$\mathbf{A}^k\mathbf{b} = ||\mathbf{b}||\mathbf{VH}^k\mathbf{e}_1 \tag{3.7}$$

where $k = 0, \cdots, q-1$ and here $|| \cdot ||$ is the 2-norm (the default norm used in the thesis).

The Arnoldi algorithm is actually a *modified* Gram-Schmidt process for the *Krylov subspace* $K_q(\mathbf{A}, \mathbf{b}) = span\{\mathbf{b}, \mathbf{Ab}, \mathbf{A}^2\mathbf{b}, \cdots, \mathbf{A}^{q-1}\mathbf{b}\}$ with the orthogonalized q vectors

[2]the A and b here do not refer to the A and b above

$\mathbf{v}_1, \cdots, \mathbf{v}_q$ constitute of the q columns of the matrix $\mathbf{V}$ as described below:
First form unit vector $\mathbf{v}_1 = \mathbf{b}/\|\mathbf{b}\|$
Then we know $\mathbf{A}\mathbf{v}_1 - <\mathbf{A}\mathbf{v}_1, \mathbf{v}_1> \mathbf{v}_1$ is orthogonal to $\mathbf{v}_1$ (here $<\mathbf{a}, \mathbf{b}>$ is the R^N inner-product $\mathbf{a}^T\mathbf{b}$) then let

$$\mathbf{v}_2 = \frac{\mathbf{A}\mathbf{v}_1 - <\mathbf{A}\mathbf{v}_1, \mathbf{v}_1> \mathbf{v}_1}{\|\mathbf{A}\mathbf{v}_1 - <\mathbf{A}\mathbf{v}_1, \mathbf{v}_1> \mathbf{v}_1\|}$$

which will be orthonormal with $\mathbf{v}_1$ and if we denote $h_{1,1} = <\mathbf{A}\mathbf{v}_1, \mathbf{v}_1>$ and $h_{2,1} = \|\mathbf{A}\mathbf{v}_1 - <\mathbf{A}\mathbf{v}_1, \mathbf{v}_1> \mathbf{v}_1\|$ we will have from above equation:

$$\mathbf{A}\mathbf{v}_1 = h_{1,1}\mathbf{v}_1 + h_{2,1}\mathbf{v}_2$$

Similarly we can proceed to form

$$\mathbf{v}_3 = \frac{\mathbf{A}\mathbf{v}_2 - <\mathbf{A}\mathbf{v}_2, \mathbf{v}_1> \mathbf{v}_1 - <\mathbf{A}\mathbf{v}_2, \mathbf{v}_2> \mathbf{v}_2}{\|\mathbf{A}\mathbf{v}_2 - <\mathbf{A}\mathbf{v}_2, \mathbf{v}_1> \mathbf{v}_1 - <\mathbf{A}\mathbf{v}_2, \mathbf{v}_2> \mathbf{v}_2\|}$$

which is orthonormal to both $\mathbf{v}_1$ and $\mathbf{v}_2$ and again denote $h_{1,2} = <\mathbf{A}\mathbf{v}_2, \mathbf{v}_1>$, $h_{2,2} = <\mathbf{A}\mathbf{v}_2, \mathbf{v}_2>$ and $h_{3,2} = \|\mathbf{A}\mathbf{v}_2 - <\mathbf{A}\mathbf{v}_2, \mathbf{v}_1> \mathbf{v}_1 - <\mathbf{A}\mathbf{v}_2, \mathbf{v}_2> \mathbf{v}_2\|$ we have

$$\mathbf{A}\mathbf{v}_2 = h_{1,2}\mathbf{v}_1 + h_{2,2}\mathbf{v}_2 + h_{3,2}\mathbf{v}_3$$

continue in this manner through to form the $\mathbf{v}_q$ from orthonormalizing $\mathbf{A}\mathbf{v}_{q-1}$ with $\mathbf{v}_1, \cdots, \mathbf{v}_{q-1}$ and after $\mathbf{v}_q$ we perform this one more step to form[3] a

$$\mathbf{v}_{q+1} = \frac{\mathbf{A}\mathbf{v}_q - <\mathbf{A}\mathbf{v}_q, \mathbf{v}_1> \mathbf{v}_1 - <\mathbf{A}\mathbf{v}_q, \mathbf{v}_2> \mathbf{v}_2 - \cdots - <\mathbf{A}\mathbf{v}_q, \mathbf{v}_q> \mathbf{v}_q}{\|\mathbf{A}\mathbf{v}_q - <\mathbf{A}\mathbf{v}_q, \mathbf{v}_1> \mathbf{v}_1 - <\mathbf{A}\mathbf{v}_q, \mathbf{v}_2> \mathbf{v}_2 - \cdots - <\mathbf{A}\mathbf{v}_q, \mathbf{v}_q> \mathbf{v}_q\|}$$

which is orthonormal to all[4] $\mathbf{v}_1, \cdots, \mathbf{v}_q$ with

$$\mathbf{A}\mathbf{v}_q = h_{1,q}\mathbf{v}_1 + h_{2,q}\mathbf{v}_2 + \cdots + h_{q,q}\mathbf{v}_q + h_{q+1,q}\mathbf{v}_{q+1}$$

[3] If we can, see the footnote on page 32
[4] So $\mathbf{V}\mathbf{v}_{q+1} = 0$

where $h_{1,q} =<\mathbf{A}\mathbf{v}_q, \mathbf{v}_1>, h_{2,q} =<\mathbf{A}\mathbf{v}_q, \mathbf{v}_2>, \cdots, h_{q,q} =<\mathbf{A}\mathbf{v}_q, \mathbf{v}_q>$ and

$$h_{q+1,q} = \|\mathbf{A}\mathbf{v}_q - <\mathbf{A}\mathbf{v}_q, \mathbf{v}_1> \mathbf{v}_1 - <\mathbf{A}\mathbf{v}_q, \mathbf{v}_2> \mathbf{v}_2 - \cdots - <\mathbf{A}\mathbf{v}_q, \mathbf{v}_q> \mathbf{v}_q\|$$

Now let

$$\mathbf{V} = (\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_q) \tag{3.8}$$

$$\mathbf{H} = \begin{bmatrix} h_{11} & h_{12} & \cdots & \cdots & h_{1q} \\ h_{21} & h_{22} & \cdots & \cdots & h_{2q} \\ & h_{32} & \ddots & & \vdots \\ & & \ddots & \ddots & \vdots \\ & & & h_{q-1,q} & h_{qq} \end{bmatrix} \tag{3.9}$$

where those blank parts in $\mathbf{H}$ are all zeros and $\mathbf{H}$ is some times called an "upper Heisenberg matrix". From the above description of the modified Gram-Schmidt process we obviously have *Property 1* and *Property 2* (where we set $h = h_{q+1,q}$). For *Property 3* we note that since the part of $\mathbf{H}$ below (not including) its lower-2nd diagonal[5] are all zeros it is easy to prove by induction that the part of $\mathbf{H}^k$ below its lower-$(k+1)$th diagonals are all zeros , where $k = 0, \cdots, q-2$ and therefore their $(q, 1)$ entries are all zero:

$$\mathbf{e}_q^T \mathbf{H}^k \mathbf{e}_1 = 0 \tag{3.10}$$

where $k = 0, \cdots, q-2$ and $\mathbf{e}_1, \mathbf{e}_q \in R^q$. We also note that the first column of matrix $\mathbf{V}$ is actually $\mathbf{v}_1 = \mathbf{b}/\|\mathbf{b}\|$ so

$$\mathbf{b} = \|\mathbf{b}\|\mathbf{V}\mathbf{e}_1$$

and from (3.6) we have

$$\|\mathbf{b}\|\mathbf{A}\mathbf{V}\mathbf{e}_1 = \|\mathbf{b}\|\mathbf{V}\mathbf{H}\mathbf{e}_1 + h\|\mathbf{b}\|\mathbf{v}_{q+1}\mathbf{e}_q^T\mathbf{e}_1 = \|\mathbf{b}\|\mathbf{V}\mathbf{H}\mathbf{e}_1$$

[5]Here we refer to the diagonal consists of $h_{2,1}, h_{3,2}, \cdots, h_{q,q-1}$. The main diagonal both the first lower and upper diagonal in this sense

i.e.

$$\mathbf{Ab} = \|\mathbf{b}\|\mathbf{VHe}_1$$

then with both equations (3.6) and (3.10),we get easily by induction that

$$\mathbf{A}^k\mathbf{b} = \|\mathbf{b}\|\mathbf{VH}^k\mathbf{e}_1$$

where $k = 0, 1, \cdots, q-1$ thus *Property 3* holds.

The Arnoldi algorithm described above can be summarized in more concise form as following:

$[\mathbf{V}, \mathbf{v}_{q+1}, \mathbf{H}] = arnoldi(\mathbf{A}, \mathbf{b}, q)$

{

$\mathbf{v}_1 = \mathbf{b}/\|\mathbf{b}\|$

for $j = 1 : q$

$\{\mathbf{w} = \mathbf{Av}_j$

for $i = 1 : j$

$\{h_{i,j} = \mathbf{w}^T\mathbf{v}_i$

$\mathbf{w} = \mathbf{w} - h_{i,j}\mathbf{v}_i\}$

$h_{j+1,j} = \|\mathbf{w}\|$

if[6] $(h_{j+1,j} \neq 0)\mathbf{v}_{j+1} = \mathbf{w}/h_{j+1,j}$ }

}

To reduce the original linear system (1.1) and (1.2) we just set $\mathbf{A}_1 = \mathbf{A}^{-1}$ and $\mathbf{b}_1 = \mathbf{A}^{-1}\mathbf{b}$ then apply the Arnoldi algorithm above on $\mathbf{A}_1, \mathbf{b}_1$ and q to get the $\mathbf{V}$ and $\mathbf{H}$ then set

$$\mathbf{A}_r = \mathbf{H}^{-1}, \mathbf{b}_r = \|\mathbf{b}_1\|\mathbf{H}^{-1}\mathbf{e}_1, \mathbf{c}_r = \mathbf{V}^T\mathbf{c} \tag{3.11}$$

[6]In case of some $h_{j+1,j}$ becomes 0, which is rare—such as the $\mathbf{b}$ happen to be an eigenvector of $\mathbf{A}$ or $q = n$—this means the q chosen for the size of the reduced system is unnecessarily overlarge.

then use *Property 3* we will have

$$\mathbf{c}_r^T\mathbf{A}_r^{-(k+1)}\mathbf{b}_r = ||\mathbf{b}_1||\mathbf{c}^T\mathbf{V}^T\mathbf{H}^{k+1}\mathbf{H}^{-1}\mathbf{e}_1 = ||\mathbf{b}_1||\mathbf{c}^T\mathbf{V}^T\mathbf{H}^k\mathbf{e}_1 = \mathbf{c}^T\mathbf{A}_1^k\mathbf{b}_1 = \mathbf{c}^T\mathbf{A}^{-(k+1)}\mathbf{b} \tag{3.12}$$

which is exactly equation 3.5 for $k = 0, 1, \cdots, q-1$

Therefore if we use $\mathbf{A}_r, \mathbf{b}_r, \mathbf{c}_r$ to form

$$\begin{aligned} \dot{\mathbf{z}} &= \mathbf{A}_r\mathbf{z} + \mathbf{b}_r u(t) \\ y_r &= \mathbf{c}_r^T\mathbf{z} \end{aligned}$$

This will be a reduced linear system for (1.1-1.2) of size q and its transfer function will match up to $(q-1)$th moments of the original system.

Remarks:

1. If the original linear system is given in the form of

$$\mathbf{A}\dot{\mathbf{x}} = \mathbf{x} + \mathbf{b}u(t) \tag{3.13}$$

$$y = \mathbf{c^T}\mathbf{x} \tag{3.14}$$

with transfer function

$$G(s) = -\mathbf{c}^T(\mathbf{I} - s\mathbf{A})^{-1}\mathbf{b} = -\sum_{i=0}^{\infty}(\mathbf{c}^T\mathbf{A}^i\mathbf{b})s^i$$

then the reduced system of size q by Arnoldi method is even simpler given by

$$\mathbf{A}_r\dot{\mathbf{z}} = \mathbf{z} + \mathbf{b}_r u(t) \tag{3.15}$$

$$y_r = \mathbf{c}_r^T\mathbf{z} \tag{3.16}$$

for which

$$\mathbf{A}_r = \mathbf{H}, \mathbf{c}_r = \mathbf{V}^T\mathbf{c}, \mathbf{b}_r = ||\mathbf{b}||\mathbf{e}_1$$

where $\mathbf{V}$ and $\mathbf{H}$ are gotten by applying Arnoldi algorithm on $\mathbf{A}$, $\mathbf{b}$ and q directly and it will match the moments[7] of the original (3.13-3.14) for up to $q-1$ th order.

2. For the system above in Remark 1, there is a very important point of view to easily interpret the Arnoldi method of model order reduction by thinking the matrix $\mathbf{V}$ form the Arnoldi algorithm as a *projection operator* from R^N to R^q, that is , make the following change of variable

$$\mathbf{x} = \mathbf{V}\mathbf{z} \tag{3.17}$$

in the original system (3.13-3.14) to get

$$\begin{aligned} \mathbf{A}\mathbf{V}\dot{\mathbf{z}} &= \mathbf{V}\mathbf{z} + \mathbf{b}u(t) && (3.18) \\ y &= \mathbf{c}^{\mathbf{T}}\mathbf{V}\mathbf{z} && (3.19) \end{aligned}$$

Then multiply both side of equation 3.18 by $\mathbf{V}^T$ and note from the properties of Arnoldi algorithm that $\mathbf{V}^T\mathbf{V} = \mathbf{I}_q$ and $\mathbf{V}^T\mathbf{A}\mathbf{V} = \mathbf{V}^T\mathbf{V}\mathbf{H} + h\mathbf{V}\mathbf{v}_{q+1}\mathbf{e}_q^T = \mathbf{H}$ we will get

$$\begin{aligned} \mathbf{H}\dot{\mathbf{z}} &= \mathbf{z} + \mathbf{V}^T\mathbf{b}u(t) && (3.20) \\ y &= \mathbf{c}^{\mathbf{T}}\mathbf{V}\mathbf{z} && (3.21) \end{aligned}$$

which is actually (3.15-3.16).

The change of variable (3.17) is a projection of the state variables from R^N to R^q by the matrix $\mathbf{V}$(which has those important moment-matching property given by Arnoldi process) and this state variables projection is very important and will be used again later in nonlinear system model order reduction.

3. The Arnoldi model order reduction method for linear system of form (1.1-1.2) as we described earlier in this section is actually derived[8] by first transforming the

[7]the kth moment in this case is $\mathbf{c}^T\mathbf{A}^k\mathbf{b}$

[8]It is worth while to point out that it will not work conveniently (having the desired moment matching property) to make a similar variable-projection as in Remark 2. directly for system (1.1-1.2) due to the fact that in Arnoldi process generally $(\mathbf{V}^T\mathbf{A}\mathbf{V})^{-1} \neq \mathbf{V}^T\mathbf{A}^{-1}\mathbf{V}$

system (1.1-1.2) to

$$\begin{aligned}
\mathbf{A}^{-1}\dot{\mathbf{x}} &= \mathbf{x} + \mathbf{A}^{-1}\mathbf{b}u(t) & (3.22)\\
y &= \mathbf{c}^{\mathbf{T}}\mathbf{x} & (3.23)
\end{aligned}$$

then reduce it by Arnoldi as described in Remark 1,and finally transform the reduced system back to be with the same form as (1.1-1.2).

4. The Arnoldi method usually works very well for reducing linear system. Just for example, the following linear system , of size $N = 1000$ is from a discretized heat conduction equation on a 1-D bar with a step heat scourge at one end as the input and the temperature at this end as output.

$$\begin{aligned}
\dot{\mathbf{x}} &= \frac{1}{\Delta x^2}\begin{bmatrix} -2 & 1 & & & \\ 1 & -2 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & -2 & 1 \\ & & & 1 & -1 \end{bmatrix}_{1000\times 1000} \begin{bmatrix} x_1 \\ \vdots \\ x_{1000} \end{bmatrix} + \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix} u(t) & (3.24)\\
y &= x_{1000} = (0, \cdots, 0, 1) \cdot \mathbf{x} & (3.25)
\end{aligned}$$

And we use Arnoldi method to reduce it to a $q = 5$ (as versus 1000!) linear system with the comparison shown in Fig 3-1.

We see although the reduced system is 200 times smaller than the original it still approximates the original *linear* system very accurately. And it is very worthwhile to point out that reduced order linear systems generated by any model order reduction methods for *linear* system based on moment matching will always match the steady state exactly, because the steady state(obtained by set $\dot{\mathbf{x}} = 0$ on the lefthand side of the linear state equation) is $-\mathbf{c}^T\mathbf{A}^{-1}\mathbf{b}$ — (negative) first moment!

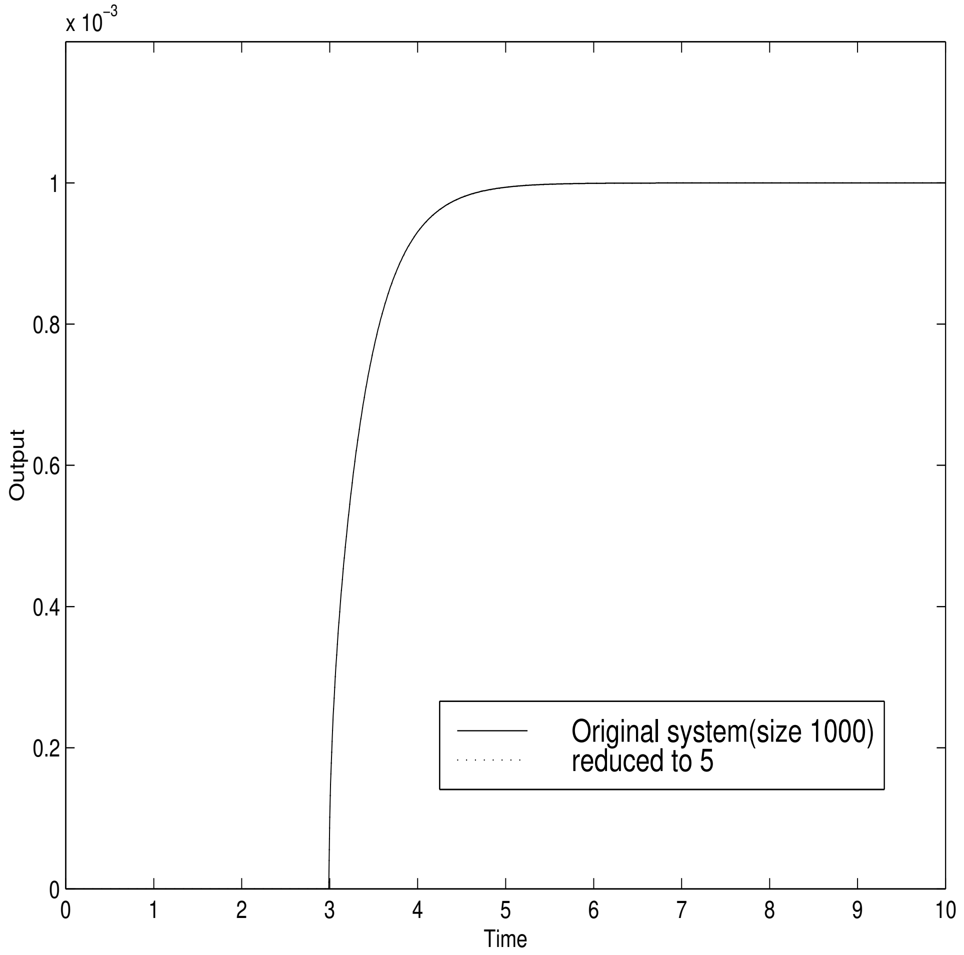

Figure 3-1: Comparison of the behavior of the original size-1000 linear system with the size-5 reduced linear system,the reduced system matches the steady state exactly

3.2 Linearization Model Order Reduction for Nonlinear Systems

As we have implied in Chapter 2 that the linearization model order reduction method for nonlinear systems will generally give no better approximation for the original nonlinear system than its linearization shown there. The linearization and linear model order reduction method will only work satisfactorily for nonlinear systems which are actually close to linear systems or operating within or near its linear regime (see for example[1, 2]). For most nonlinear systems, we will have to use methods other than it.

Chapter 4

Quadratic Reduction for Nonlinear Systems

This chapter will present the "quadratic reduction" method to perform model order reduction[1] on nonlinear systems, which will reduce the original system of any nonlinearality to a reduced system of quadratic nonlinearality. As already implied in Chapter 2 this method is based on the strategy of first approximate the original nonlinear system by its quadratic approximation (which generally gives a much better approximation to the original system than its linear approximation (linearized system) does); and then reduce this quadratic approximation system (which has the same size with the original nonlinear system) to a much smaller quadratic system which matches that quadratic approximation system[2](to high precision) therefore this reduced quadratic system will be a good reduced nonlinear system to the original nonlinear system. It will represent both a reduction in size and in nonlinearality yet will be seen to still match the original system as well as the quadratic approximation does. The basis of the method will be how to reduce a quadratic system, as to be

[1] "Model" is treated simply as a synonym of "system" here.

[2] Three different level of systems will be involved in this Chapter. To avoid confusion of which is which , we will call the quadratic approximation to the original nonlinear system the "quadratic approximation system" (which has the same size with the original nonlinear system) , and call the reduced system from this quadratic approximation system "reduced quadratic system" (which is still quadratic and is supposed to have a much smaller size than the quadratic approximation system). Similar name conventions will also be conformed with in following chapters

presented in Section 4.1; although in Section. 4.2.2 we will present methods to get a reduced quadratic system without explicitly computing the quadratic approximation system.

4.1 Model Order Reduction for Quadratic Systems

4.1.1 Tensorial Notation for Quadratic Systems

Any quadratic system with size N that we are considering can be written in the following concise form:

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{x}^T\mathbf{W}\mathbf{x} + \mathbf{b}u(t) \tag{4.1}$$

$$y = \mathbf{c}^T\mathbf{x} \tag{4.2}$$

where $\mathbf{W}$ is a 3-D array or an $N \times N \times N$ tensor.

Equation 4.1 is a vector equality and every the quadratic part lies on $\mathbf{x}^T\mathbf{W}\mathbf{x}$ with each element of which expressed as a quadratic form of the N state variables $x_1, \cdots, x_N$. As a 3-D array $\mathbf{W}$ can be thought of as N of pages of matrices where expressed in convenient Matlab convention the kth page is $W(:,:,k)$ which is a $N \times N$ matrix,denoted as $\mathbf{W}_k$, and the vector $\mathbf{x}^T\mathbf{W}\mathbf{x}$, denoted here as $\mathbf{p}$ is simply determined by

$$\mathbf{p}(k) \equiv p_k = \mathbf{x}^T\mathbf{W_k}\mathbf{x} \tag{4.3}$$

with $k = 1, \cdots, N$

As a tensor[3] $\mathbf{W}$ with 3 indices we usually write it as $\mathbf{W} = (w_{ij}^k)$ where we stress that in tensor notations *the superscripts are indices instead of powers*[4], as easily

[3]More detailed and complete treatise about tensor operations and notations can be found in almost any book on tensor analysis or differential geometry , see for example, the classic [5] or [6]

[4]The distinction of superscripts and subscripts actually have more subtle meaning involving coordinate transformation properties but this aspect will not appear in our usage of tensors

distinguishable from the context. . We will call $\mathbf{W}$ the *quadratic tensor* of the system. The benefits of using tensor notations is that it can make the indices and multi summations we use later clearer and easier to keep track with(tensors are still stored and manipulated as multi-dimensional arrays in our computations). Here for $\mathbf{x}^T\mathbf{R}\mathbf{x}$ we will have

$$(\mathbf{x}^T\mathbf{W}\mathbf{x})^k = w^k_{ij}x^ix^j \tag{4.4}$$

where again all superscripts are indices and we also make use of the popular *"Einstein summation convention"* that a pair of up-stairs and down-stairs index (i.e., a superscript and subscript of the same letter) means summation over this index so equation 4.4 says the same thing with equation 4.3.

Tensors with m superscripts and n subscripts are called a (m, n) type tensor and in this framework a vector $\mathbf{x} = (x^1, ..., x^N)^T$ is a (1,0) tensor and a matrix[5] $\mathbf{A} = (a^i_j)$ is a (1,1) tensor and our quadratic tensor $\mathbf{W}$ is a tensor of *type* (1,2) and *size* $N \times N \times N$.

The two main tensor operations that we will use are the tensor product and the contraction:

Tensor product with the notation $\otimes$ is a binary operation defined for any two tensors[6]. The tensor product of an (m,n) tensor with a (p,q) tensor is simply an (m+p,n+q) tensor defined element-wise. For example the tensor product of (1,2) tensor $\mathbf{W}$ and matrix $\mathbf{A}$ will be a (2,3) tensor, if denoted as $\mathbf{R}$, determined by

$$\mathbf{R} \equiv (r^{pq}_{ijk}) \equiv S \otimes A = (w^p_{ij}a^q_k)$$

The contraction is an operation on one tensor by summation over a pair of upper and lower case indices. For example, the contraction *operator* T^1_2 is defined on any (2,2) tensor and it acting on $\mathbf{Q} = (q^{ij}_{kl})$ will *contract* the 1st superscript with the 2nd subscript to generate a (1,1) tensor(i.e. a matrix) $\mathbf{P} = (p^i_j)$ where

$$\mathbf{P} = T^1_2(\mathbf{Q})$$

[5]when a matrix is written is this form the convention will be that the superscript i is the row index while the subscript j is the column index

[6]They do not even to have the same dimension

and

$$p^i_j = q^{ki}_{jk} \equiv \sum_k q^{ki}_{jk}$$

Contractions can also be carried out in parallel, for example, T^{12}_{12} will mean to contract the first superscript with the first subscript and also the second superscript with the second subscript, for example on (2,2) tensors this would be $T^{12}_{12}[(\alpha^{ij}_{kl})] = (\alpha^{uv}_{uv})$ which results in a scalar.

It is the contraction composed with tensor product that will give a variety of ways to generate new tensors out of the old. For example with this notation the product of two matrices $\mathbf{A} = (a^i_j)$ and $\mathbf{B} = (b^u_v)$ (suppose they are dimensionally matched to multipliable) is simply another matrix (an (1,1) tensor)

$$\mathbf{C} = T^2_1(\mathbf{A} \otimes \mathbf{B})$$

And the goal of model order reduction for quadratic system (4.1-4.2) is to find a reduced quadratic system with size q

$$\dot{\mathbf{z}} = \hat{\mathbf{A}}\mathbf{z} + \mathbf{z}^T\hat{\mathbf{W}}\mathbf{z} + \hat{\mathbf{b}}u(t) \tag{4.5}$$

$$\hat{y} = \hat{\mathbf{c}}^T\mathbf{z} \tag{4.6}$$

where $z \in R^q$ and $\hat{\mathbf{W}}$ a $q \times q \times q$ (quadratic) tensor.

4.1.2 The Abstract Concept of Model Order Reduction

It is helpful to make some contemplation about the abstracted concept of model order reduction. We can say a model order reduction is an operator Mor which acts on some class of systems , for example on the quadratic systems of form (4.1-4.2)—we denote this class of system as QUA—or on the class nonlinear systems (NOL) of the form (1.5-1.6) or on the class of linear systems (LIN). And when it operated on a "original" system S belonging to this (domain) class and with state variables $\mathbf{x} \in R^N$

it will generate a "reduced" system S_r in some class and with state variables[7] $\mathbf{x}$ for the state variables$\mathbf{z} \in R^q$ (and q is supposed to be much smaller than N) For example, the Arnoldi model order reduction for linear systems described in Chapter 3) is a model order reduction operator from LIN to LIN, and the quadratic reduction method(s) for reducing nonlinear systems to be presented in this chapter will be some model order reduction operator(s) from NOL to QUA and the model order reduction for quadratic systems in this section will be a reduction operator from QUA to QUA.[8]

With specific sizes, we can denote a model order reduction operator Mor which reduces size-N systems to size q systems as Mor_q^N and it is easy to see the following requirement will be a fundermental criterion for such a model order reduction to be a satisfiable one:

Requirement(Recovery Property): When $q = N$, the reduced system should be *equivalent* to the original system(although they may differ in *form*) . That is, given original system S, $S_r = Mor_q^N[S]$ should generate exactly the *same* (not just very close) output $y(t)$ with that of S for same input $u(t)$

A model oder reduction must satisfy the above recovery property to be expected to give a general acceptable reduced system S_r with $q < N$ and when we increase q we can expect S_r to be closer and closer to the original S until become equivalent to S when q reaches N (no reduction).

An important and useful way to get a model order reduction operator is by way of a state variables projection, like what we do in the *Remark 2* of Chapter 3. There we are working on the class of linear systems $LLIN$("left LIN") with form (3.13-3.14) use the matrix $\mathbf{V}$ to make a projection form the high dimension (N) state space to low dimension (q) state space (see also the footnote on page 34). Because $\mathbf{V}$ is orthonormal matrix there it will be invertible when $q = N$ so the recovery property is easily seen to be satisfied. If we denote this reduction operator as $LArn$, acting from

[7]We use different letter for state variables simply to avoid confusions

[8]In this spirit and language , if we also consider the quadratic approximation procedure for nonlinear systems to be an "approximation" operator QA from NOL to QUA then the (explicit) quadratic reduction methods for nonlinear systems in this chapter is simply the model order reduction for quadratic systems composed with QA.

$LLIN$ to $LLIN$, and also denote the transformation of a linear system of the form

$$\begin{aligned}\dot{\mathbf{x}} &= \mathbf{A}\mathbf{x} + \mathbf{b}u(t) \\ y &= \mathbf{c}^{\mathbf{T}}\mathbf{x}\end{aligned}$$

which belongs to LIN to

$$\begin{aligned}\mathbf{A}^{-1}\dot{\mathbf{x}} &= \mathbf{x} + \mathbf{A}^{-1}\mathbf{b}u(t) \\ y &= \mathbf{c}^{\mathbf{T}}\mathbf{x}\end{aligned}$$

which belongs to $LLIN$ as an operator L (for "left multiplication") , acting from LIN to $LLIN$ then the Arnoldi model order reduction for LIN in Chapter 3 is just formed as $Arn = L^{-1} \circ LArn \circ L^{-1}$ where $LArn$ reduces the size and L keeps the size when operating.

This suggests to us that the similar *linear*[9] state variables projection from original state space to low dimensional (reduced) state space and which is *invertible* when $q = N$ could also be applicable for nonlinear systems especially quadratic systems with appropriate forms to make a promising model order reduction.

4.1.3 Reducing a Quadratic System

With all the preparation in the preceding sections, we now proceed to reduce the quadratic system

$$\begin{aligned}\dot{\mathbf{x}} &= \mathbf{A}\mathbf{x} + \mathbf{x}^T\mathbf{W}\mathbf{x} + \mathbf{b}u(t) \\ y &= \mathbf{c}^T\mathbf{x}\end{aligned}$$

which is of size N.

With the consideration in the footnote on page 34 and to get a convenient reduced

[9]In principle we could also imagine a nonlinear change of variables $\mathbf{x} = \mathbf{g}(\mathbf{z})$ for nonlinear systems but unless for special forms it will usually be not easy to rearrange the resulted system to an easy-to-dealt-with form while still keeping the size of the *reduced* system down

system we first transform the system to

$$\begin{aligned}
\mathbf{A}^{-1}\dot{\mathbf{x}} &= \mathbf{x} + \mathbf{A}^{-1}\mathbf{x}^T\mathbf{W}\mathbf{x} + \mathbf{A}^{-1}\mathbf{b}u(t) \\
y &= \mathbf{c}^{\mathbf{T}}\mathbf{x}
\end{aligned}$$

which differs from (3.13-3.14) only with the addition of the quadratic term. Inspired by the success of the Arnoldi linear projection of state variables in reducing (3.13-3.14) and the corresponding discussions in Section 4.1.2 we set $\mathbf{A}_1 = \mathbf{A}^{-1}, \mathbf{b}_1 = \mathbf{A}^{-1}\mathbf{b}$ then apply the Arnoldi algorithm described on page 3.1 on $\mathbf{A}_1, \mathbf{b}_1$ and a q chosen for the reduced system size to get the matrices $\mathbf{V}$ and $\mathbf{H}$ where $\mathbf{V}$ is an $N \times q$ matrix ,$\mathbf{V}^T\mathbf{V} = \mathbf{I}_q$ and $\mathbf{H} = \mathbf{V}^T\mathbf{A}\mathbf{V}$

Then we perform the similar change of variable which is the key state space projection[10] :

$$\mathbf{x} = \mathbf{V}\mathbf{z}$$

where $\mathbf{z} \in R^q$ to get[11]

$$\begin{aligned}
\mathbf{A}_1\mathbf{V}\dot{\mathbf{z}} &= \mathbf{V}\mathbf{z} + \mathbf{A}_1\mathbf{z}^T\mathbf{V}^T\mathbf{W}\mathbf{V}\mathbf{z} + \mathbf{b}_1u(t) \\
y &= \mathbf{c}^{\mathbf{T}}\mathbf{V}\mathbf{z}
\end{aligned}$$

then

$$\begin{aligned}
\mathbf{H}\dot{\mathbf{z}} &= \mathbf{V}\mathbf{A}_1\mathbf{V}\dot{\mathbf{z}} = \mathbf{z} + \mathbf{V}^T\mathbf{A}_1\mathbf{z}^T\mathbf{V}^T\mathbf{W}\mathbf{V}\mathbf{z} + \mathbf{V}^T\mathbf{b}_1u(t) \\
y &= \mathbf{c}^{\mathbf{T}}\mathbf{V}\mathbf{z}
\end{aligned}$$

[10]Since $\mathbf{V}$ is still invertible when $q = N$ so the recovery property is still satisfied

[11]Here $\mathbf{V}^T\mathbf{W}\mathbf{V}$ still represents page-wise matrix multiplication as a convenient shorthand in this case with result of the multiplication being N pages of $q \times q$ matrices which is also a $N \times q \times q$ type(1,2) tensor and can be further page-wise multiplied with $\mathbf{z}^T$ and $\mathbf{z}$

and

$$
\begin{aligned}
\dot{\mathbf{z}} &= \mathbf{H}^{-1}\mathbf{z} + \mathbf{H}^{-1}\mathbf{V}^T\mathbf{A}_1\mathbf{z}^T\mathbf{V}^T\mathbf{W}\mathbf{V}\mathbf{z} + \mathbf{H}^{-1}\mathbf{V}^T\mathbf{b}_1 u(t) \\
y &= \mathbf{c}^{\mathbf{T}}\mathbf{V}\mathbf{z}
\end{aligned}
$$

Now we set

$$
\begin{aligned}
\hat{\mathbf{A}} &= \mathbf{H}^{-1} &(4.7)\\
\hat{\mathbf{b}} &= \mathbf{H}^{-1}\mathbf{V}^T\mathbf{b}_1 = \|\mathbf{b}_1\|\mathbf{H}^{-1}\mathbf{e}_1 &(4.8)\\
\hat{\mathbf{c}} &= \mathbf{V}^T\mathbf{c} &(4.9)
\end{aligned}
$$

which is actually the same as equation 3.11 because these are still the reduced data of the linear part.

For the complicated quadratic term $\hat{\mathbf{g}}(\mathbf{z}) = \mathbf{H}^{-1}\mathbf{V}^T\mathbf{A}_1\mathbf{z}^T\mathbf{V}^T\mathbf{W}\mathbf{V}\mathbf{z}$ now (temporarily) appearing above, we need to rewrite it as a standard form to have a reduced quadratic system in real sense. For this we define the $q \times N$ matrix $\mathbf{D} = \hat{\mathbf{A}}\mathbf{V}^T\mathbf{A}_1 \equiv \mathbf{H}^{-1}\mathbf{V}^T\mathbf{A}_1 \equiv \mathbf{H}^{-1}\mathbf{V}^T\mathbf{A}^{-1}$ and the $\mathbf{B} = \mathbf{V}^T\mathbf{W}\mathbf{V}$ which is actually a $N \times q \times q$ type(1,2) tensor. Then

$$\mathbf{g}(\mathbf{z}) = \mathbf{D}(\mathbf{z}^T\mathbf{B}\mathbf{z})$$

and if $\mathbf{D} = (d^i_j)$,$\mathbf{B} = (\beta^k_{st})$ we denote $[\mathbf{v}]_i$ for the ith element of any vector $\mathbf{v}$ we have, in terms of elements,

$$[\hat{\mathbf{g}}(\mathbf{z})]_i = d^i_k(z^s\beta^k_{st}z^t) = z^s(d^i_k\beta^k_{st})z^t = z^s\hat{w}^i_{st}z^t$$

or

$$\mathbf{g}(\mathbf{z}) = \mathbf{z}^T\hat{\mathbf{W}}\mathbf{z} \tag{4.10}$$

where $\hat{\mathbf{W}} = (\hat{w}^i_{st})$ with $\hat{w}^i_{st} = d^i_k\beta^k_{st}$ or in tensor notation

$$\hat{\mathbf{W}} = T^2_1(\mathbf{D} \otimes \mathbf{B}) \tag{4.11}$$

Therefor we finally reduced the original quadratic system to

$$\begin{aligned}\dot{\mathbf{z}} &= \hat{\mathbf{A}}\mathbf{z} + \mathbf{z}^T\hat{\mathbf{W}}\mathbf{z} + \hat{\mathbf{b}}u(t) \\ \hat{y} &= \hat{\mathbf{c}}^T\mathbf{z}\end{aligned}$$

which is exactly of the form (4.5-4.6).

We can now summarize the above method for reducing a size N quadratic system

$$\begin{aligned}\dot{\mathbf{x}} &= \mathbf{A}\mathbf{x} + \mathbf{x}^T\mathbf{W}\mathbf{x} + \mathbf{b}u(t) \qquad (4.12)\\ y &= \mathbf{c}^T\mathbf{x} \qquad (4.13)\end{aligned}$$

to size q as follows:

Step 1. Set $\mathbf{A}_1 = \mathbf{A}^{-1}, \mathbf{b}_1 = \mathbf{A}^{-1}\mathbf{b}$

Step 2. Apply Arnoldi algorithm(page 32) on $\mathbf{A}_1, \mathbf{b}_1, q$ to get $\mathbf{V}$ and $\mathbf{H}$

Step 3. Set $\hat{\mathbf{A}} = \mathbf{H}^{-1}, \hat{\mathbf{b}} = ||\mathbf{b}_1||\mathbf{H}^{-1}\mathbf{e}_1$ and $\hat{\mathbf{c}} = \mathbf{V}^T\mathbf{c}$

Step 4. Let $\mathbf{D} = \hat{\mathbf{A}}\mathbf{V}^T\mathbf{A}_1$ and $\mathbf{B} = \mathbf{V}^T\mathbf{W}\mathbf{V}$ to get $\hat{\mathbf{W}} = T_1^2(\mathbf{D} \otimes \mathbf{B})$

Then

$$\begin{aligned}\dot{\mathbf{z}} &= \hat{\mathbf{A}}\mathbf{z} + \mathbf{z}^T\hat{\mathbf{W}}\mathbf{z} + \hat{\mathbf{b}}u(t) \\ \hat{y} &= \hat{\mathbf{c}}^T\mathbf{z}\end{aligned}$$

will be a reduced quadratic system of size q

To verify the effectiveness of this method we apply it on the quadratic approximation of the nonlinear circuit system in Section 2.3 and shown in Fig. 2-3 with $N = 100$ capacitors and the same resistor profile $g(v)$ function chose there.

This quadratic system is of the form (4.12-4.13) with

$$\mathbf{A} = \begin{bmatrix} -82 & 41 & & & \\ 41 & -82 & 41 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & -2 & 1 \\ & & & 1 & -1 \end{bmatrix}_{100\times 100}$$

which is tridiagonal and the quadratic tensor $\mathbf{W} = (w^i_{jk}) = (\mathbf{W}^i)$ where $N\times N$ matrix (the ith page of this (1,2) tensor) $\mathbf{W}^i = (w^i_{::})$ is given by the following rules:

$\mathbf{W}^1$ has its [1:2,1:2] block being $\begin{bmatrix} -1600 & 800 \\ 800 & -800 \end{bmatrix}$ and all other parts zeros;

$\mathbf{W}^i$ where $i = 2, \cdots, N{-}1$ has its $[i{-}1{:}i{+}1, i{-}1{:}i{+}1]$ block being $\begin{bmatrix} 800 & -800 & 0 \\ -800 & 0 & 800 \\ 0 & 800 & -800 \end{bmatrix}$ and all other parts zeros;

$\mathbf{W}^i$ has its lower-right $[N-1{:}N, N-1{:}N]$ block being $\begin{bmatrix} 800 & -800 \\ -800 & 800 \end{bmatrix}$ and all other parts zeros;

We now reduce this size-100 quadratic system to $q = 5, q = 10$ and $q = 20$ respectively with the results (still using the step source as in Section 2.3) shown in Fig. 4-1. We see the $q = 5$ reduced system indeed departs from the original quadratic system in transit behavior while capturing the steady state still well, and both the $q = 10$ (10 times smaller than the original) and $q = 20$ reduced system match the original system almost accurately so that we cannot even well distinguish them in the plot.

And we also try various other form of sources including sinusoidal, exponential and polynomial sources with their results shown in Fig. 4-2–4-5. In all of them we see the $q = 10$ reduced quadratic system (let alone the $q = 20$ one) gives a very good accuracy in matching the original quadratic system.

We also record the computation time for integrating the $N = 100$ and its sys-

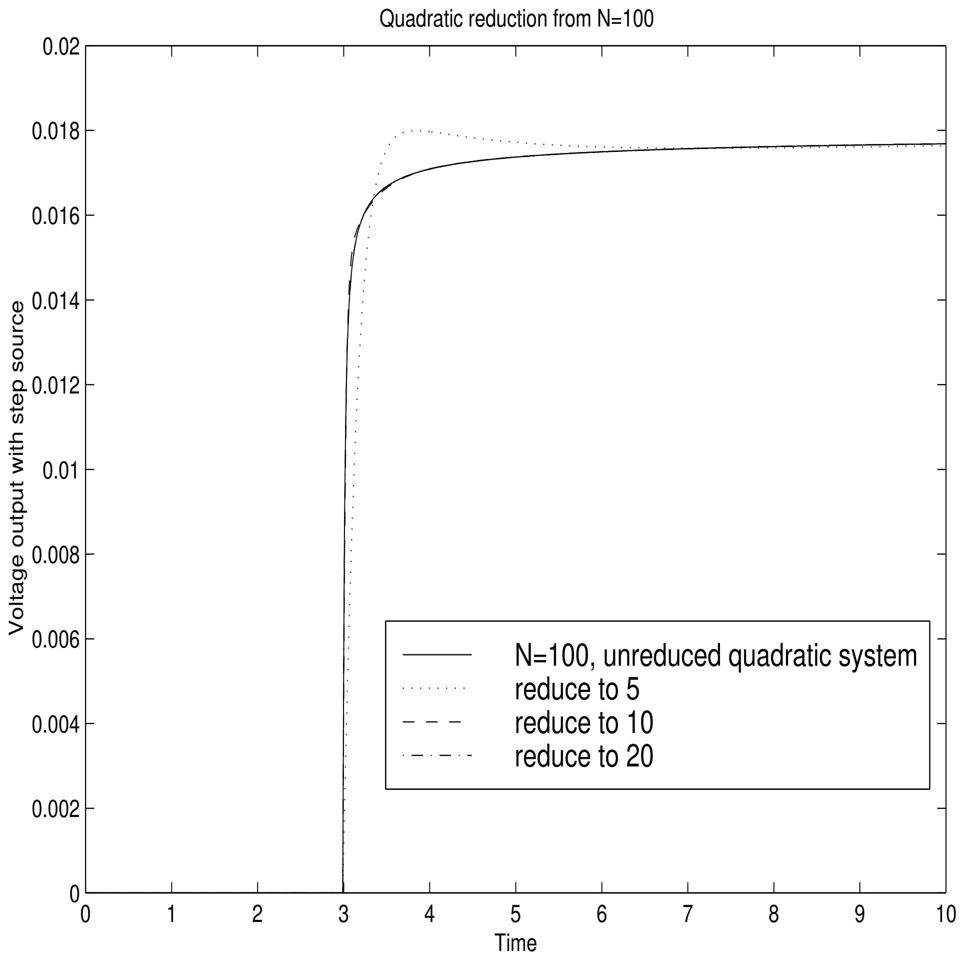

Figure 4-1: Comparison of the original size-100 quadratic system to its quadratic reduction to size 5 ,10 and 20

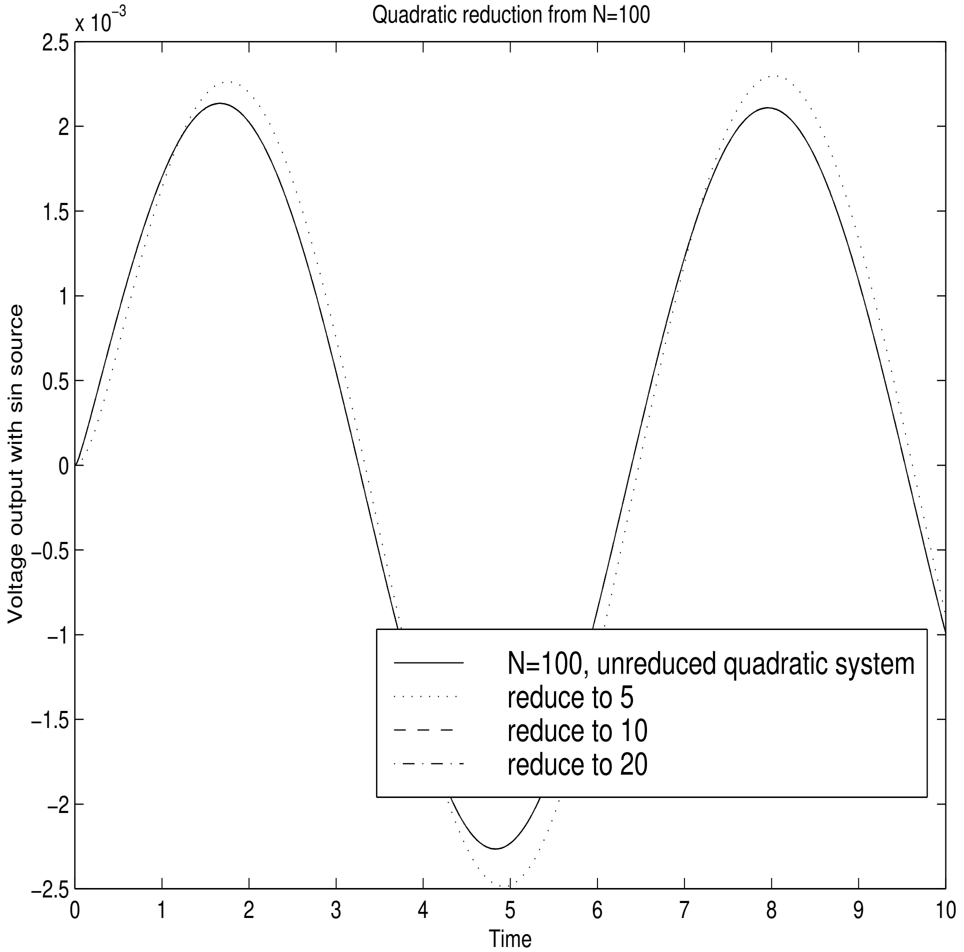

Figure 4-2: Comparison of the original size-100 quadratic system to its quadratic reduction to size 5,10 and 20:response to a sinusoidal source $u(t) = \sin(t)/10$

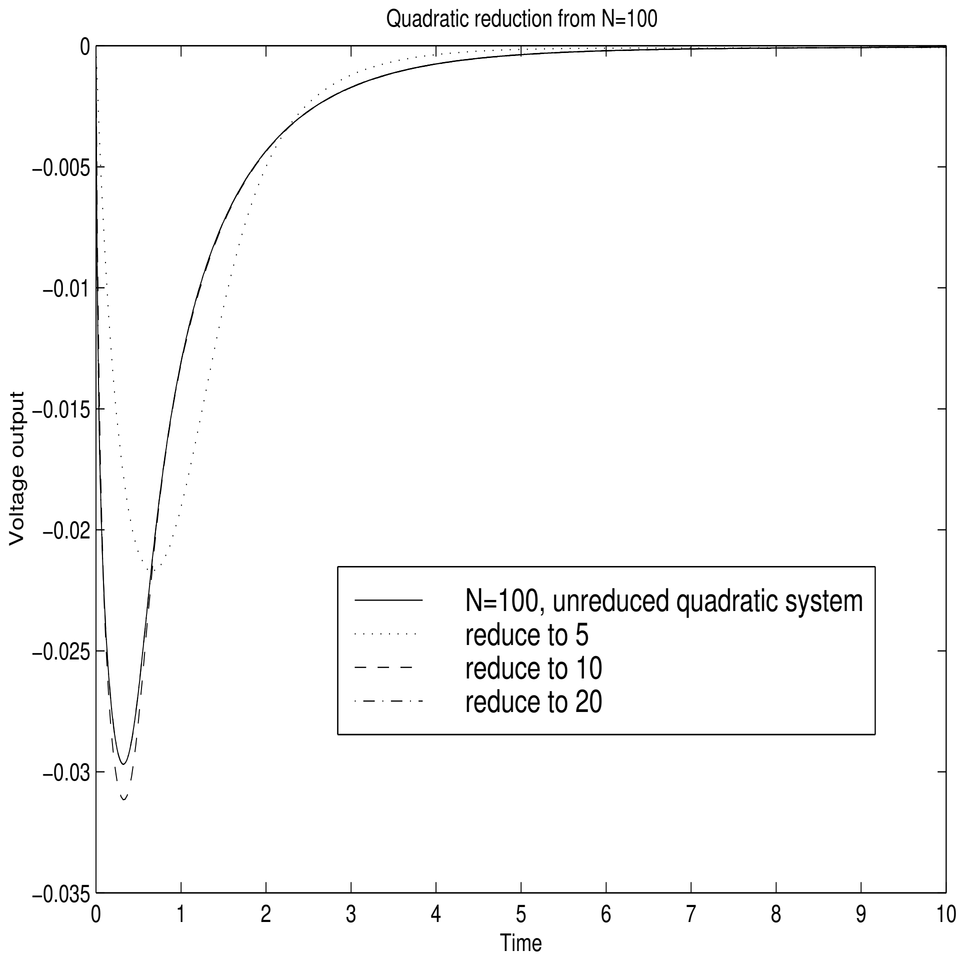

Figure 4-3: Comparison of the original size-100 quadratic system to its quadratic reduction to size 5,10 and 20:response to a source $u = -e^{-t}$

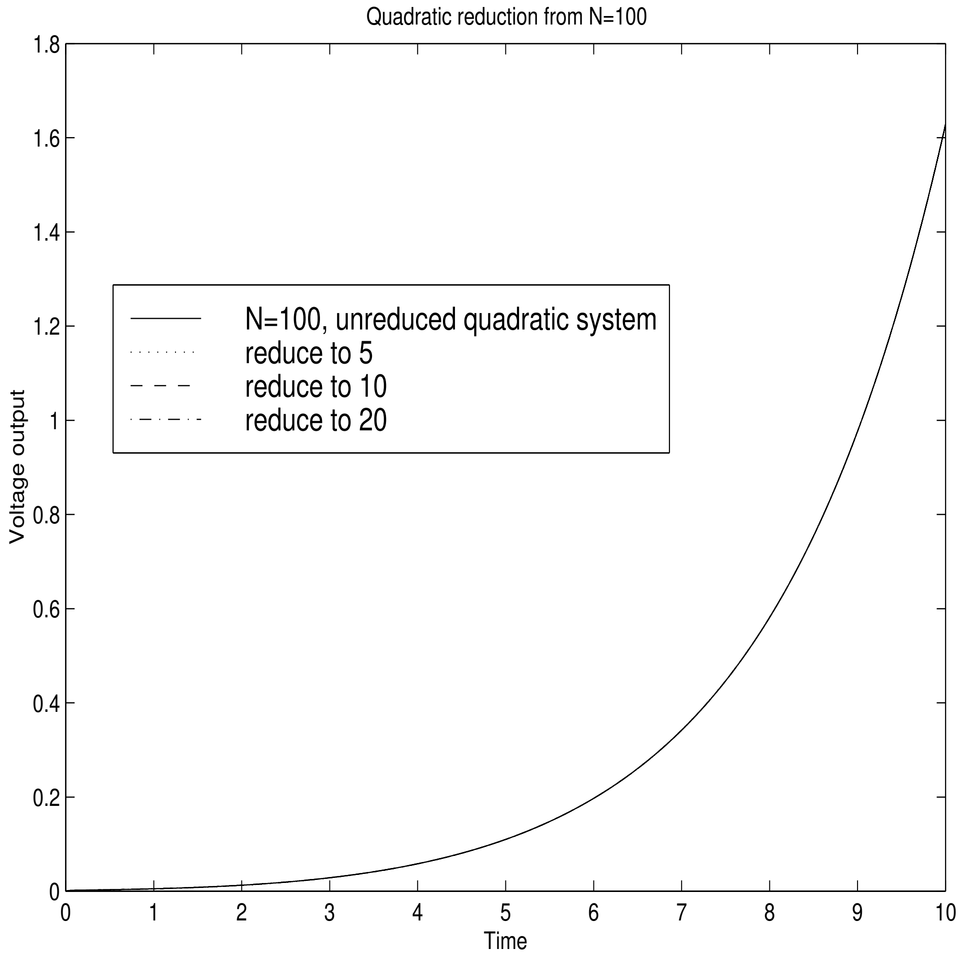

Figure 4-4: Comparison of the original size-100 quadratic system to its quadratic reduction to size 5,10 and 20:response to a source $u = e^t/10$; in this case even the q=5 match the original one accurately

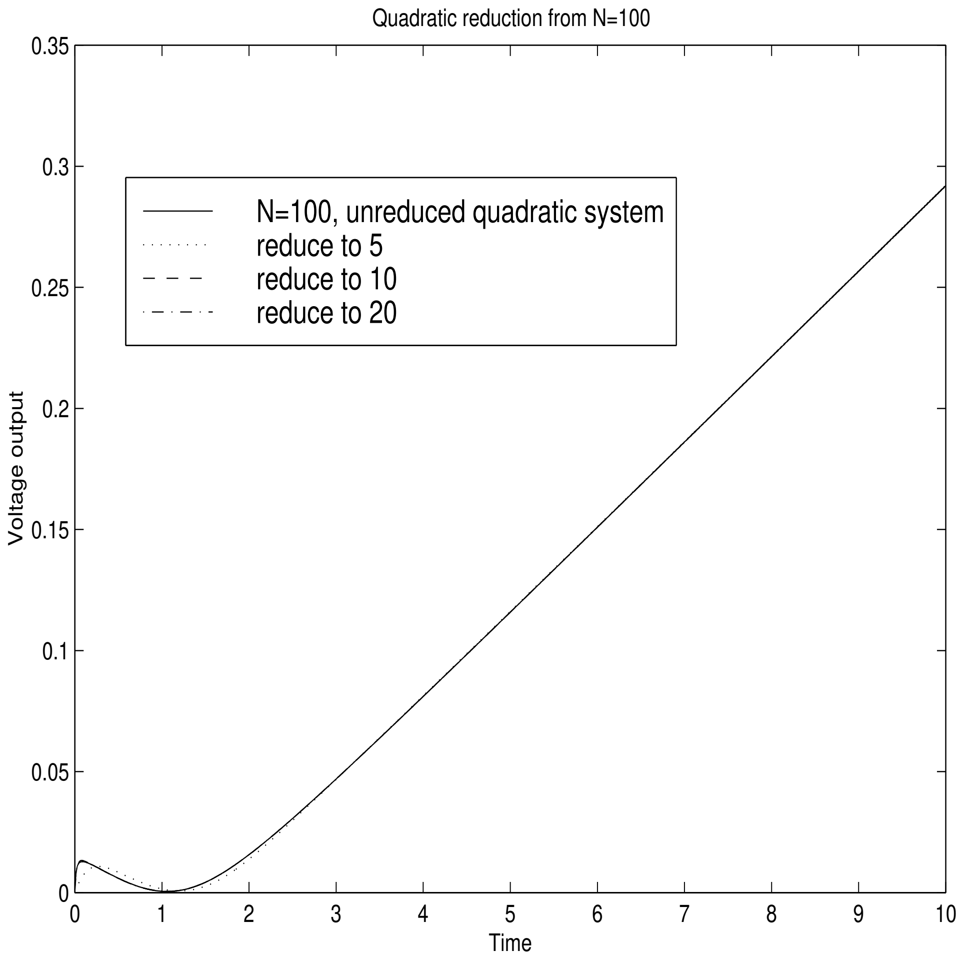

Figure 4-5: Comparison of the original size-100 quadratic system to its quadratic reduction to size 5,10 and 20:response to a source $u = (t-1)^2$

Computation time (in sec)	unreduced quadratic system(N=100)	reduce to 5	reduce to 10	reduce to 20
with step source	902.3	1.77	8.95	43.8
with sin source	1120	1.60	7.66	45.7
with source $u = -e^{-t}$	1177	1.25	7.59	45.3
with source $u = e^t/10$	5329	8.14	71.4	267.4
with source $u = (t-1)^2$	822.8	0.84	7.20	26.6

Table 4.1: Comparison of computation time for integrating unreduced and reduced quadratic systems

tems in the above computations in Table 4.1 from which we see the reduced systems (particularly the $q = 10$ one which gives very satisfactory match) indeed make a tremendous reduction in computation time. And since the reduced quadratic tensor $\mathbf{W}_r$ of size $q \times q \times q$ is usually dense (even when the original one is sparse) the computational complexity for integrating the reduced quadratic system will increase as $O(q^3)$ with the size of the reduced system, which comes from doing the vector-tensor multiplication $\mathbf{z}^T\mathbf{W}_r\mathbf{z}$.

4.2 Quadratic Reduction for Nonlinear Systems

In the previous section we see that the quadratic reduced system can give a very satisfactory accuracy in matching the unreduced quadratic system. And in Chapter 2 we already see the quadratic approximation of a nonlinear system usually approximate the original nonlinear system to a much better accuracy than the linearization system, therefore the quadratic reduced system generated from the quadratic approximation of a nonlinear system can offer a quite satisfactory reduced system for the original nonlinear system which both saves much of the computation time and also match the original nonlinear system to a good accuracy as well as the original quadratic approximation does.

For clearer demonstration and reference we plot the behavior of the original nonlinear system for the nonlinear circuit example in Section 2.3 with size 100 (for which we already reduce its quadratic approximation to $q = 10$ in last section) together with those of the reduced quadratic system (to $q = 10$) and its reduced linear system (to

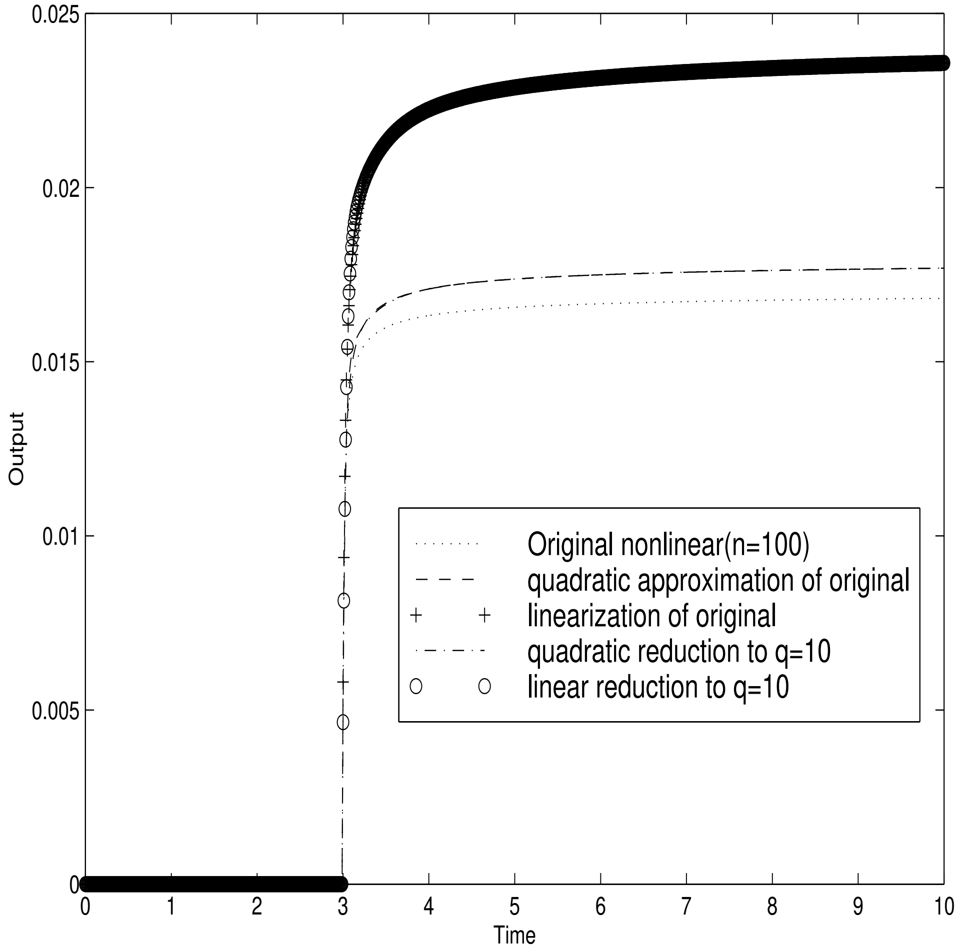

Figure 4-6: Comparison of the original nonlinear system(size 100) with the reduced systems generated by quadratic reduction and by linearization to size 10.The response output is for the step source and here we also plot the original quadratic approximation and linearization systems for reference

$q = 10$) by linearization in Fig. 4-6 and Fig 4-7 with step source and an exponential source respectively. And we also compare the integration[12] time for computing them in Table4.2. From these we clearly see the reduced quadratic system (with 10 times smaller in size) gives a quite good approximation to the original nonlinear system which is much closer than the linear reduced one but still costing only a fraction of the time used to integrate the original nonlinear.

[12]Implemented on a faster machine than the one used for computations in Table 4.1

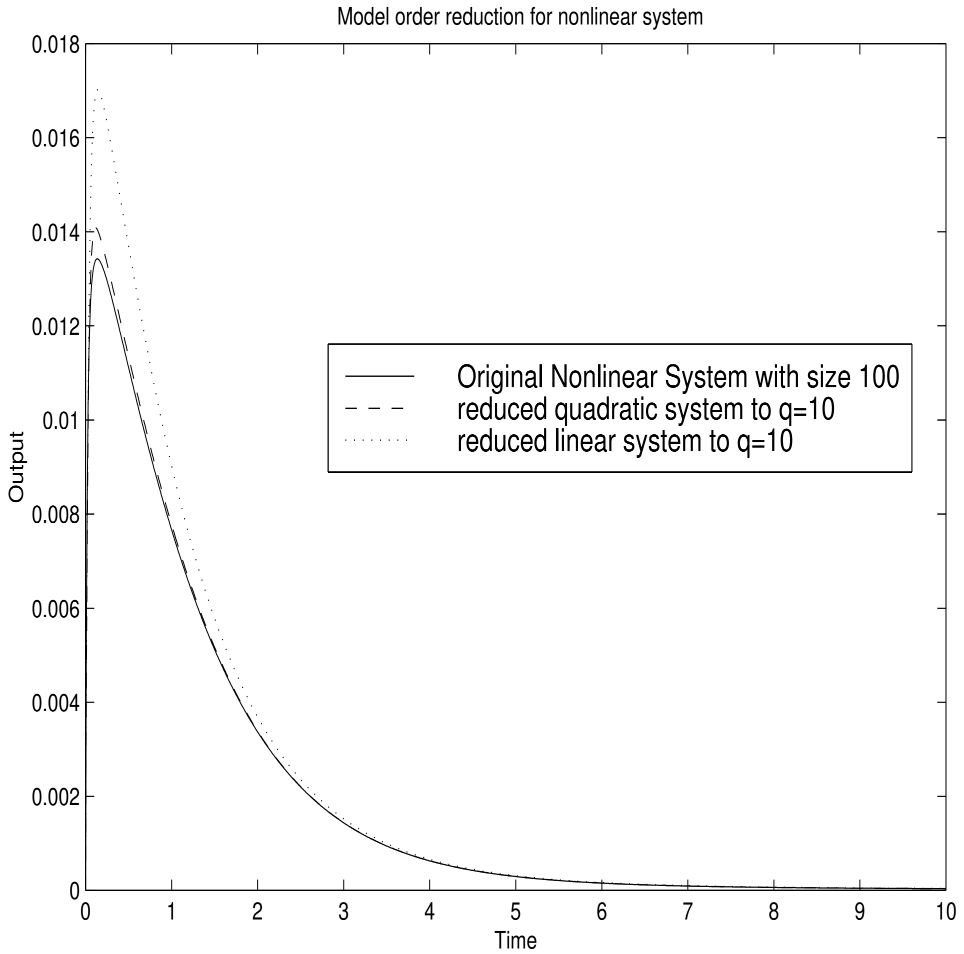

Figure 4-7: Comparison between the same original nonlinear system with its quadratic and linear reductions but for an input source $u = e^{-t}$

Computation time (in sec)	Original nonlinear (size 100)	quadratic reduced system(size 10)	linear reduced system(size 10)
with step source	98.3	6.74	1.60
with source $u = e^{-t}$	115.3	6.77	1.60

Table 4.2: Comparison of computation time to integrating original nonlinear system and its quadratic and linear reduced systems

4.2.1 Explicit Quadratic Reduction Method

Suppose we have the following nonlinear system

$$\begin{aligned}\dot{\mathbf{x}} &= \mathbf{f}(\mathbf{x}) + \mathbf{b}u(t)\\ y &= \mathbf{c}^T\mathbf{x}\end{aligned}$$

and $\mathbf{f}(\mathbf{x})$ has the following 2nd order Taylor expansion (we assume $\mathbf{f}(\mathbf{0}) = 0$) as usual)

$$\mathbf{f}(\mathbf{x}) \approx \mathbf{D_f}(0)\mathbf{x} + \frac{1}{2}\mathbf{x}^T\mathbf{H_f}(0)\mathbf{x} \tag{4.14}$$

where $\mathbf{f} = (f_1, f_2, \cdots, f_N)^T, \mathbf{D_f}(0) = (\frac{\partial f^i}{\partial x_j}|_0)$ is the Jacobian at 0 and $\mathbf{H_f}(0) = (\frac{\partial^2 f^i}{\partial x_j \partial x_k}|_0)$ is the Hesse tensor of $\mathbf{f}$ at 0 with type (1,2).

Therefore the above original nonlinear system has the quadratic approximation

$$\begin{aligned}\dot{\mathbf{x}} &= \mathbf{A}\mathbf{x} + \mathbf{x}^T\mathbf{W}\mathbf{x} + \mathbf{b}u(t)\\ y &= \mathbf{c}^T\mathbf{x}\end{aligned}$$

with $\mathbf{A} = \mathbf{D_f}(0)$ and $\mathbf{W} = \frac{1}{2}\mathbf{H_f}(0)$ here.

And the explicit quadratic reduction method to reduce the above nonlinear system is just first compute the Jacobian and Hesse tensor of $\mathbf{f}$ to get the quadratic approximation system then use the reduction method for quadratic systems described in the previous section to get the reduced quadratic system. This will be convenient if the original nonlinear system is given in simple analytic form and it is easy to get the entire quadratic approximation system.

4.2.2 Implicit Quadratic Reduction Methods

The implicit quadratic reduction is to generate the reduced quadratic system for the original nonlinear systems without computing all the second derivatives in the Hesse tensor explicitly[13] which(although only once during the reduced model generation

[13]We will still need to compute the Jacobian in these methods, however

process) can be time-consuming (having N^3 elements) or not worthwhile to compute especially when the original system is very large or not given in a simple analytic form.

To do this, we first compute the Jacobian of $\mathbf{f}$ at 0 to get the matrix $\mathbf{A}$ and in the same way as described in last section to get matrices $\mathbf{H}$ and $\mathbf{V} = (v^i_j)$ by applying Arnoldi on $\mathbf{A}^{-1}, \mathbf{A}^{-1}\mathbf{b}, q$ then get the $\hat{\mathbf{A}}, \hat{\mathbf{b}}, \hat{\mathbf{c}}$ and also compute the auxiliary matrix $\mathbf{D} = (d^i_j) = \mathbf{H}^{-1}\mathbf{V}^T\mathbf{A}^{-1}$

Recall the reduced quadratic tensor $\hat{\mathbf{W}}$ is formed by

$$\hat{W}^k_{ij} = d^k_l \beta^l_{ij}$$

where

$$(\beta^l_{ij}) = \mathbf{B} = \mathbf{V}^T\mathbf{W}\mathbf{V}$$

so $\beta^l_{ij} = v^r_i w^l_{rs} v^s_j = \frac{1}{2} v^r_i \frac{\partial^2 f^l}{\partial x_r \partial x_s}|_0 v^s_j$

therefore

$$\begin{aligned} \hat{w}^k_{ij} &= \frac{1}{2} d^k_l \frac{\partial^2 f^l}{\partial x_r \partial x_s}|_0 v^r_i v^s_j \\ &= v^s_j \frac{\partial}{\partial x_s}(\frac{1}{2} d^k_l \frac{\partial f^l}{\partial x_r} v^r_i)|_0 \end{aligned}$$

which we recognize as a directional derivative.

If we denote vector $\mathbf{v}_j = (v^\cdot_j)$ and define functions

$$g^k_i(\mathbf{x}) = \frac{1}{2} d^k_l \frac{\partial f^l}{\partial x_r}(\mathbf{x}) v^r_i$$

or in matrix form a matrix function

$$\mathbf{G}(\mathbf{x}) = (g^k_i(\mathbf{x})) = \frac{1}{2}\mathbf{D}\cdot\mathbf{D}_{\mathbf{f}}(\mathbf{x})\cdot\mathbf{V} \tag{4.15}$$

then we have

$$\hat{w}^k_{ij} = D_{\mathbf{v}_j} g^k_i(0) \tag{4.16}$$

or in more compact notation

$$\hat{w}^{:}_{:j} = D_{\mathbf{v}_j}\mathbf{G}(0)$$

Using difference we can also give an approximation formula to be used in computation for equation 4.16 as

$$\hat{w^k_{ij}} \doteq \frac{(g^k_i)(\epsilon\mathbf{v}_j) - (g^k_i)(\mathbf{0})}{\epsilon} = \frac{(d^k_l\frac{\partial f^l}{\partial x_r}v^r_i)(\epsilon\mathbf{v}_j) - (d^k_l\frac{\partial f^l}{\partial x_r}v^r_i)(\mathbf{0})}{2\epsilon} \tag{4.17}$$

where ϵ is a very small positive number.

We can get another formula for $\hat{w}^k_{ij}$ even avoiding using functions g^k_i which have first derivatives in them ,which will give us a second order type *difference* formula. This is by observing further that

$$g^k_i(\mathbf{x}) = \frac{1}{2}d^k_l\frac{\partial f^l}{\partial x_r}(\mathbf{x})v^r_i = v^r_i\frac{\partial}{\partial x_r}(\frac{1}{2}d^k_l f^l)(\mathbf{x}) = D_{\mathbf{v}_i}h_k(\mathbf{x})$$

where we define functions

$$h^k(\mathbf{x}) = \frac{1}{2}d^k_l f^l \tag{4.18}$$

or

$$\mathbf{h}(\mathbf{x}) = (h^k) = \frac{1}{2}\mathbf{D}\cdot\mathbf{f}(\mathbf{x})$$

therefore

$$\begin{aligned}
\hat{w^k_{ij}} &\doteq \frac{(g^k_i)(\epsilon\mathbf{v}_j) - (g^k_i)(\mathbf{0})}{\epsilon} \\
&= \frac{D_{\mathbf{v}_i}h_k(\epsilon\mathbf{v}_j) - D_{\mathbf{v}_i}h^k(0)}{\epsilon} \\
&\doteq \frac{\frac{h^k(\epsilon\mathbf{v}_i+\epsilon\mathbf{v}_j)-h^k(\epsilon\mathbf{v}_j)}{\epsilon} - \frac{h^k(\epsilon\mathbf{v}_i)-h^k(0)}{\epsilon}}{\epsilon} \\
&= \frac{h^k(\epsilon(\mathbf{v}_j+\mathbf{v}_i)) - h^k(\mathbf{v}_j) - h^k(\epsilon\mathbf{v}_i) + h^k(0)}{\epsilon^2}
\end{aligned} \tag{4.19}$$

One can also have a centralized-like difference scheme,written in vector form it is

$$\hat{w}^{:}_{ij} \doteq \frac{\mathbf{h}(\frac{\epsilon(\mathbf{v}_j+\mathbf{v}_i)}{2}) - \mathbf{h}(\frac{\epsilon(\mathbf{v}_j-\mathbf{v}_i)}{2}) - \mathbf{h}(\frac{\epsilon(\mathbf{v}_i-\mathbf{v}_j)}{2}) + \mathbf{h}(-\frac{\epsilon(\mathbf{v}_j+\mathbf{v}_i)}{2})}{\epsilon^2} \tag{4.20}$$

In the first scheme (4.17) we need to evaluate the Jacobian q times (at the vectors $\epsilon\mathbf{v}_1, \cdots, \epsilon\mathbf{v}_q$) and there are two summation loops from 1 to N (indexed by l and r in (4.17) which will need $O(N^2)$ computations when the Jacobian is dense and $O(N)$ when it is sparse like in out circuit example) when computing the functions g_i^k and there are q^2 such functions to compute. So the total complexity is $O(q^3N^2)$ when the Jacobian is dense and $O(q^3N)$ when it is sparse. In the second scheme(see (4.19) or (4.20)) we have q^2 function evaluations to compute h_k (q such functions) and there is only one summation loop of size N when computing h_k so this scheme has a total complexity of $O(q^3N)$ and could be better than the first scheme when the Jacobians are dense.

Chapter 5

Further and Other Methods for Reducing Nonlinear Systems

5.1 Third and Higher Order Methods to Reduce Nonlinear Systems

We see in the previous chapter that the quadratic reduction for nonlinear system based on its quadratic approximations can generate reduced systems approximating the original to a quite good accuracy and with low computation cost. We can use 3rd or higher order approximations when even higher precision is required for the reduced system to match the original nonlinear system whose higher order approximations matches the original more accurately, but with some increased computation time and memory requirement compared with the quadratic reductions.

The 3rd order method is just a natural extension of the quadratic reduction method in last Chapter. Here for a nonlinear system in form (1.5-1.6) we approximate the nonlinear term[1] $\mathbf{f}(\mathbf{x}) = (f^1(\mathbf{x}), f^2(\mathbf{x}), \cdots, f^N(\mathbf{x}))^T$ by its 3rd order Taylor expansion:

$$\mathbf{f}(\mathbf{x}) \doteq \mathbf{A}\mathbf{x} + \mathbf{x}^T\mathbf{W}\mathbf{x} + P(\mathbf{x}, \mathbf{R}) \tag{5.1}$$

[1]Again we assume $\mathbf{f}(0) = 0$

where again $\mathbf{A}$and $\mathbf{W}$ are the linear and quadratic part as given in last Chapter and $\mathbf{R}$ is for the 3rd degree part and it is a (1,3) type $N \times N \times N \times N$ tensor given as

$$\mathbf{R} = (R^l_{ijk}) = \frac{1}{3!}(\frac{\partial^3 f^l}{\partial x_i \partial x_j \partial x_k}|_0) \tag{5.2}$$

and the 3rd order term $P(\mathbf{x}, \mathbf{R})$ in equation 5.1 is a vector-(1,3) tensor multiplication which results in a vector given as (indexed by l)

$$P(\mathbf{x}, \mathbf{R}) = (R^l_{ijk} x^i x^j x^k) = T^{234}_{123}(\mathbf{R} \otimes \mathbf{x} \otimes \mathbf{x} \otimes \mathbf{x}) \tag{5.3}$$

and we know the resultant 3rd degree system

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{A}\mathbf{x} + \mathbf{x}^T \mathbf{W} \mathbf{x} + P(\mathbf{x}, \mathbf{R}) + \mathbf{b} u(t) && (5.4) \\ y &= \mathbf{c}^T \mathbf{x} && (5.5) \end{aligned}$$

can give a even higher precision approximation to the original nonlinear system as observed in Fig. 2-5

And the process to generate a reduced 3rd degree system from the 3rd order approximation is also directly similar to the quadratic reduction in last Chapter by still using the Arnoldi type state variable projection to transform all the linear, quadratic and 3rd order term to low dimensional form. We briefly summarize the procedure as follows:

Step 1. Set $\mathbf{A}_1 = \mathbf{A}^{-1}, \mathbf{b}_1 = \mathbf{A}^{-1}\mathbf{b}$

Step 2. Apply Arnoldi algorithm(page 32) on $\mathbf{A}_1, \mathbf{b}_1, q$ to get $\mathbf{V}$ and $\mathbf{H}$

Step 3. Set $\hat{\mathbf{A}} = \mathbf{H}^{-1}, \hat{\mathbf{b}} = \|\mathbf{b}_1\| \mathbf{H}^{-1}\mathbf{e}_1$ and $\hat{\mathbf{c}} = \mathbf{V}^T\mathbf{c}$. This reduces the linear components.

Step 4. Let $\mathbf{D} = \hat{\mathbf{A}}\mathbf{V}^T\mathbf{A}_1$ and $\mathbf{B} = \mathbf{V}^T\mathbf{W}\mathbf{V}$ to get $\hat{\mathbf{W}} = T^2_1(\mathbf{D} \otimes \mathbf{B})$ This reduces the quadratic part.

Step 5. Let[2] $\hat{\mathbf{R}} = T^{2345}_{1234}(\mathbf{D} \otimes \mathbf{R} \otimes \mathbf{V} \otimes \mathbf{V} \otimes \mathbf{V})$. This reduces the 3rd degree part.

[2]Written in elements this is $\hat{R}^l_{ijk} = d^l_s R^s_{pqr} v^p_i v^q_j v^r_k$

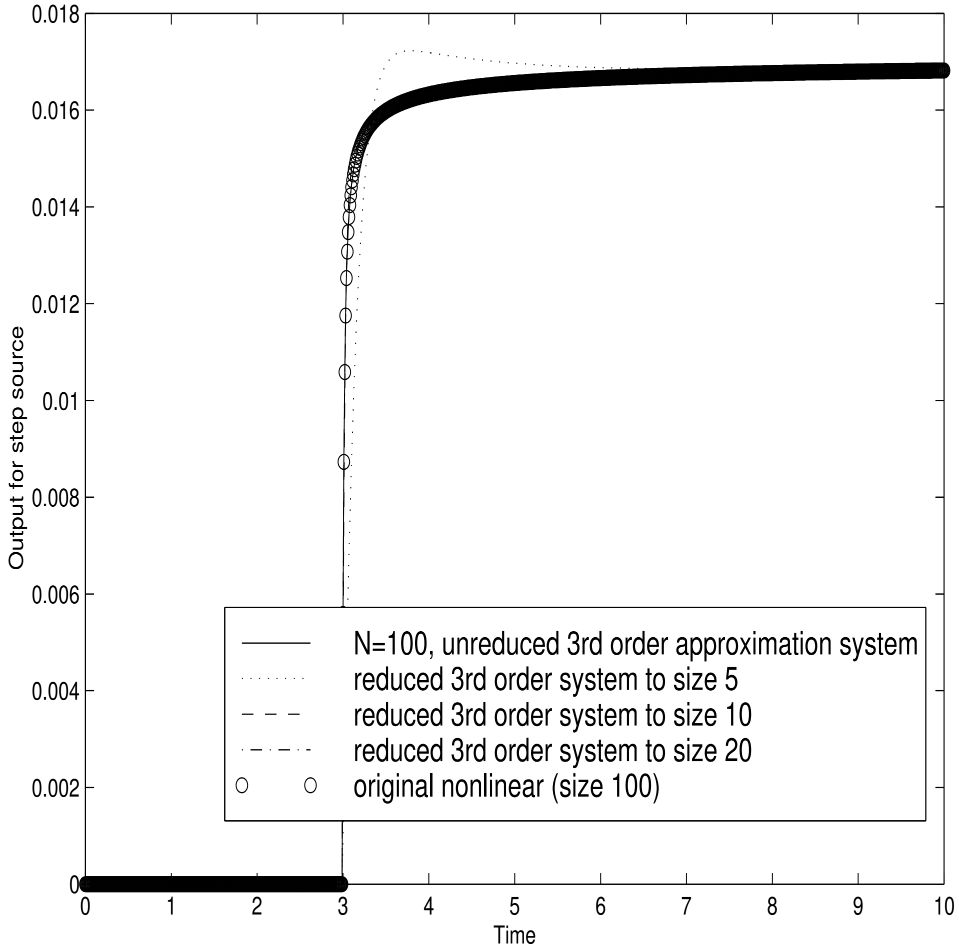

Figure 5-1: Comparison of the original size-100 nonlinear system to its 3rd order approximation and reduction to sizes 5 ,10 and 20

Then

$$\begin{aligned}\dot{\mathbf{z}} &= \hat{\mathbf{A}}\mathbf{z} + \mathbf{z}^T\hat{\mathbf{W}}\mathbf{z} + +P(\mathbf{z}, \hat{\mathbf{R}}) + \hat{\mathbf{b}}u(t) \\ \hat{y} &= \hat{\mathbf{c}}^T\mathbf{z}\end{aligned}$$

will be a reduced 3rd degree system of size q

We can apply the above 3rd order reduction method[3] for the nonlinear circuit example system to get the result given in Fig. 5-1

We see the 3rd order reduced system is almost an exact match to the original

[3]using backward Euler integration with approximate Newton method

nonlinear system, although it uses as much as about 1/3 of the integration time[4] used to integrate the original nonlinear system which is not as significant a time reduction as compared with quadratic reduced system.

The even higher degree reduction methods can be posed just similarly as the 3rd order reduction method. But those will significantly increase the time and memory cost and therefore are rarely used.

5.2 Dealing with General Forms of Nonlinear Systems

All the nonlinear systems that we have been considering so far are of or near the standard form which has the feature that the input term is separated from the state variables. But our model order reduction approach can be extended to deal with even more general forms of systems.

Suppose we are given a nonlinear system of the following form

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{b}(\mathbf{x})u(t) \tag{5.6}$$

$$y = \mathbf{g}(\mathbf{x}) \tag{5.7}$$

in which $\mathbf{b}(\mathbf{x})$ is a nonlinear vector function and the state variables are not separated from the input[5]. We can still expand all the nonlinear functions $\mathbf{f}, \mathbf{b}, \mathbf{g}$ to quadratic forms and get the following generalized quadratic system as approximation to the original:

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{x}^T\mathbf{W}\mathbf{x} + (\mathbf{b} + \mathbf{B}\mathbf{x} + \mathbf{x}^T\mathbf{S}\mathbf{x})u(t) \tag{5.8}$$

[4] On a SunUltraSparc 30 machine(300MHz) it costs 92.5 seconds to integrate the original size-100 nonlinear system, 4.8 seconds to integrate the quadratic reduced size-10 system and 32 seconds to integrate the third order reduced size-10 system

[5] In some cases we can try to find a coordinate transformation $\bar{\mathbf{x}} = \bar{\mathbf{x}}(\mathbf{x})$ with Jacobian $\mathbf{J}(\mathbf{x}) = \frac{\partial(\bar{x}_1,\cdots,\bar{x}_N)}{\partial(x_1,\cdots,x_N)}$ such that $\mathbf{J}(\mathbf{x}) \cdot \mathbf{b}(\mathbf{x}) = \bar{\mathbf{b}}$ is a constant vector,then the transformation $\bar{\mathbf{x}} = \bar{\mathbf{x}}(\mathbf{x})$ can separate the input from the (new) state variables

$$y = \mathbf{c} + \mathbf{E}\mathbf{x} + \mathbf{x}^T\mathbf{U}\mathbf{x} \tag{5.9}$$

and our approach of using state-variable projection together with the tensor contraction can still apply to this system to generate a reduced quadratic system of the similar form.

For the even more general form of nonlinear system

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, u) \tag{5.10}$$

$$y = \mathbf{g}(\mathbf{x}) \tag{5.11}$$

where $\mathbf{f}$ is a general nonlinear function of both state variables and input we may still expand $\mathbf{f}$ to get the quadratic approximation of the above form which may have multi-inputs. And the higher order approaches can also be extended similarly.

5.3 Some Other Methods for Nonlinear Model Order Reduction

There are some other methods existed for doing model order reductions on nonlinear systems. [7, 8] try to use balancing technique to reduce nonlinear system satisfying certain conditions; [9] discusses using decomposition method to try to extract a small set of basis functions for modeling in the context of fluid integration systems. [10] studies generating low order macromodels for a type of MEMS system using a generalized-coordinate approach. Most of these methods are either only posed for some specific types of nonlinear systems or have not been practically applicable for general nonlinear model order reductions. A more detailed review for many of these methods is given in [11]. We also feel that our methods in this thesis may have limitations in some situations, for example when the nonlinear term of the nonlinear state equation is very bounded even for large state variables (therefore the quadratic approximation of it is even farther away from the original nonlinear term than the linear approximation globally) or the output of the system is tending to be unbounded for large simulation time (such that the error between the quadratic approximation and the original grows to be very large); for these cases further model order reduction methods beyond the simple PP-type (using *Polynomial* approximation together use one *Projection* of state variables) reduction methods in this thesis should be developed.

Chapter 6

Conclusions

This thesis develops some practical and efficient methods for doing model order reduction on nonlinear systems.

We obtain a highly successful method for reducing quadratic systems and demonstrate its accuracy and efficiency through nonlinear circuit example. This method coupled with the quadratic approximation for nonlinear systems generate quadratic reduced order systems for original nonlinear systems which offer a much better accuracy than linearization methods. We also extend this method to third or even higher order which offer even higher accuracy for reducing the nonlinear systems.

This approach is applicable for a large category of nonlinear systems whose quadratic or higher order polynomial approximations give a closer approximation for the original nonlinear systems than the linearization do.

Bibliography

[1] F. Wang and J. White, Automatic Model Order Reduction of a Microdevice using the Arnoldi Approach, *International Mechanical Engineering Congress and Exposition*, Anahiem, November 1998, pp. 527-530

[2] Qiutao (Frank) Wang, Macro-modeling of Micro-electrical-mechanical System Devices, MIT MS Thesis, June 1998.

[3] MIT 6.336 Lecture Notes *Introduction to Numerial Algorithms*, Instructor:Jacob White, 1998

[4] L.M.Silveira,M.Kamon and J.White, Efficient Reduced-Order Modeling of Frequency-Dependent Coupling Inductances Associated with 3-D Interconnect Structures, *IEEE Trans. on Components,Packaging and Manufacturing Technology* Part B, 19(2):283-288, 1996

[5] L.P.Eisenhart,Riemannian Geometry, Princeton, Princeton University Press, 1949

[6] J.R.Munkres,Analysis on manifolds,Redwood City, Calif. Addison-Wesley Pub. Co., Advanced Book Program, 1990, c1991

[7] M.A.Scherpen,Balancing for Nonlinear Systems, *System and Control Letters* 21:143-153,1993

[8] X.Ma and J.A.De Abreu-Garcia, On the Computation of Reduced Order Models of Nonlinear Systems Using Balancing Technigues, In *Proceedings of the 27th Conference on Decison and Control*, December 1998

[9] S.R.Sipcic and A.Pecore, Karhunen-Loéve Decomposition in Dynamical Modeling:Symbolic Computation Perspectives, *Mathematical Modeling and Scientific Computing* 6,1996

[10] L.D.Gabbay, Computer Aided Macromodeling for MEMS, MIT EE Department Ph.D Thesis, June 1998

[11] J.R.Phillips, Some Aspects of Model Reduction for Nonlinear Systems, MIT EE Department Area Exam Report, Febuaray,1997