Parameterized Model Order Reduction for Nonlinear Dynamical Systems

by

Bradley N. Bond

B.S., Engineering Science and Mechanics (2004)
Pennsylvania State University

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Science in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2006

Author .....
Department of Electrical Engineering and Computer Science
May 24, 2006

Certified by.....
Luca Daniel
Assistant Professor of Electrical Engineering and Computer Science
Thesis Supervisor

Accepted by .....
Arthur C. Smith
Chairman, Department Committee on Graduate Students

Parameterized Model Order Reduction for Nonlinear Dynamical Systems

by

Bradley N. Bond

Submitted to the Department of Electrical Engineering and Computer Science
on May 25, 2006, in partial fulfillment of the
requirements for the degree of
Master of Science in Electrical Engineering and Computer Science

Abstract

The presence of several nonlinear analog circuits and Micro-Electro-Mechanical (MEM) components in modern mixed signal System-on-Chips (SoC) makes the fully automatic synthesis and optimization of such systems an extremely challenging task. The research presented in this thesis concerns the development of techniques for generating Parameterized Reduced Order Models (PROMs) of nonlinear dynamical systems. Such reduced order models could serve as a first step towards the automatic and accurate characterization of geometrically complex components and subcircuits, eventually enabling their synthesis and optimization.

This work combines elements from a non-parameterized trajectory piecewise linear method for nonlinear systems with a moment matching parameterized technique for linear systems. Exploiting these two methods one can create four different algorithms for generating PROMs of nonlinear systems. The algorithms were tested on three different systems: a MEM switch and two nonlinear analog circuits. All three examples contain distributed strong nonlinearities and possess dependence on several geometric parameters.

Using the proposed algorithms, the local and global parameter-space accuracy of the reduced order models can be adjusted as desired. Models can be created which are extremely accurate over a narrow range of parameter values, as well as models which are less accurate locally but still provide adequate accuracy over a much wider range of parameter values.

Thesis Supervisor: Luca Daniel
Title: Assistant Professor of Electrical Engineering and Computer Science

Acknowledgments

First and foremost I would like to thank Luca for taking me in as his student and teaching me everything I know about model order reduction. It has been a pleasure to work with him, and I look forward to several more years under his tutelage. I would also like to thank my groupmates for creating the enjoyable atmosphere in which I have worked. They provided insightful discussions and constructive criticism on many occasions.

Others who played an instrumental role in my surviving two years in this moderately-horrible city are Anne, who took me to an even more horrible city every few weeks but kept me company every evening; Dan and Kevin, who watched Law & Order while looking out over the dumpster with me; Arthur, who sold me all of his possessions for $100; the Tang Hall Softball team, which helped me relive my Little League glory days; Andy, who cooked me delicious meals most nights of the week; my buddies from State College, who visited me on multiple occasions; NoseRoseToes, who helped me come up with catchy section titles for this thing; and the Charles River, who provided me with something to run in circles around when I got tired of working.

Most importantly, thank you to my family for helping me reach this point in my life. I wouldn't be anywhere near here without their support over the years. This includes Mom, Dad, Nick, Stephanie, Tug, Deedle, Oma and Opa, Gramps, and all the others.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The presence of several nonlinear analog circuits and Micro-Electro-Mechanical (MEM) components in modern mixed signal System-on-Chips (SoC) makes the fully automatic synthesis and optimization of such systems an extremely challenging task. The nonlinearities of such devices are both strong and distributed, hence they cannot simply be approximated by linear components or be removed from the system and modelled separately. The availability of techniques for generating Parameterized Reduced Order Models (PROMs) of nonlinear dynamical systems could serve as a first step toward the automatic and accurate characterization of geometrically complex components and subcircuits, eventually enabling their synthesis and optimization.

In the past few years, several techniques for Parameterized Model Order Reduction (PMOR) of linear systems and nonparameterized MOR for nonlinear systems have been introduced in literature (Section 2.1). However, none of these methods considered parameterized models from nonlinear systems. The key contribution of this work is a method for automatic extraction of parameterized reduced order models from complex nonlinear systems. The proposed approach builds upon a non-parameterized trajectory piecewise linear method for nonlinear systems (Section 2.2.3) and a moment matching parameterized technique for linear systems (Section 2.2). The proposed approach is derived in detail in Chapter 3, and followed by four possible algorithms to implement the method. The method has been tested on three physical systems which possess distributed strong nonlinearites, described in detail along with results

in Chapter 4. By expanding upon these results in a comparative analysis, we aim at discovering and clarifying how each algorithm affects the parameter-space accuracy of reduced models. In addition to accuracy analysis, computational complexity and timing results are presented in Chapter 5. To conclude, we will recap our observations and suggest further improvements to the proposed algorithms in Chapter 6.

Chapter 2

Background

2.1 Survey of Existing Methods

Several parameterized model order reduction techniques for linear systems have been introduced in the literature over the past few years. Some methods are based on statistical performance evaluation [4, 5] or optimization [6], while others use projection based approaches such as moment matching [7, 8, 9, 10, 11, 12, 13] or Truncated Balance Realizations (TBR) [14, 15].

One work which uses statistical performance evaluation [5] is interested in exploiting non-physical mathematical models to explore the design space for synthesis. Data mining is used to construct regressors based on some set of simulation data, and then a committee of regressors is used to fit a function to that data. Each regressor is accurate only in some region of the space, so a voting scheme is used to pick which regressors should be used at each simulation step. A process called boosting is used to build the committee of regressors. The algorithm is as follows. First, given a large set of simulation samples, evenly weight every data point. Second, sample the points based on their weights and fit a regressor to the sampled data. Finally, compute the average weighted training error for the data set, and proceed back to the second step updating the weights based on the weighted training error.

A group working on modeling interconnects for high-performance clock-net design [4] encountered computational limitations due to the size of their interconnect models.

As a solution, they proposed solving a reduced system instead of the large system at each iteration. Using an algorithm to weight the effects of N wire locations on delay and skew, they were able to consider only the the q most important wires as an approximation to the full system.

Another work on modeling interconnect delays [7] proposed a method based on matrix perturbation theory and dominant-pole-analysis to capture effects due to manufacturing variations. For projection based methods the goal is to find a low order subspace in which the state's evolution is well approximated. Once this space is identified, the linear system can be projected into such subspace using a projection matrix V. Considering the linear system

$$[G(w_1, w_2) + sC(w_1, w_2)]\, x = b, \tag{2.1}$$

where x is the state, b is the input vector, s is the Laplace variable, and G and C are the conductance and capacitance matrices that depend on geometrical parameters w_1 and w_2, the authors in [7] were able to bound the variation in the dominant eigenvalues and eigenvectors of G and C due to small changes in w_1 and w_2. Using this information they then computed an invariant Krylov subspace and projection matrix V used to compute the reduced parameterized system with $\hat{G} = V^T G V$ and $\hat{C} = V^T C V$. This method was then tested on three RLC networks.

Moment matching was first considered with PMOR after [13] and [9] introduced the idea of treating a single parameter as an additional variable in the transfer function, $H(s,p)$, and recursively computed moments with respect to both variables to approximate $H(s,p)$. These moments span an important subspace for the system and may be used to generate the columns of the projection matrix. For systems of the form

$$[s_1 G_1 + s_2 G_2 - A]\, x = Bu \tag{2.2}$$

recursive formulae for generating the Krylov subspace moments along with statements about the accuracy of the ROM at the expansion points are presented in [9]. Extensions of this technique are found in [10, 11], which use similar ideas to achieve

moment matching for an arbitrary number of parameters with arbitrary parameter dependence. This is achieved by a multivariable Taylor series expansion, treating each parameter as a variable. This technique is discussed in detail in Section 2.2.2.

Another PMOR projection based method uses Truncated Balance Realizations (TBR) to construct the projection matrix. Several techniques which use TBR to create reduced order models capturing the effects of manufacturing variations in interconnects have been proposed [14, 15]. In TBR, the controllability Grammiam X and observability Grammian Y are computed by solving the Lyapunov equations

$$AX + XA^T = -BB^T \tag{2.3}$$

$$A^TY + YA = -CC^T \tag{2.4}$$

where A, B, C are the system matrices and vectors in the linear system

$$\frac{dx}{dt} = Ax + Bu, \quad y = C^Tx. \tag{2.5}$$

The controllability and observability Grammians can be used to identify the most controllable and observable states in the system respectively. The dominant eigenvalues of the product XY are invariant under similarity transform, so the corresponding eigenvectors are used as the basis for the reduced space. One nice result of this method is that provable error bounds for the PROM are available. The first work [14] considers a spectrally weighted TBR procedure to produce reduced models which are more accurate around important frequencies. If A in (2.5) is perturbed by ΔA, then a Grammian transformation matrix $\Delta\gamma$ can be computed. If X and Y were computed from the linear system with $A = A_0$, then $\Delta\gamma$ can be used to bound the variation in the Grammians which arise from the perturbed system with $A = A_0 + \Delta A$, such that

$$X_{var} \leq \Delta\gamma X \Delta\gamma^T \tag{2.6}$$

$$Y_{var} \leq \Delta\gamma^T Y \Delta\gamma \tag{2.7}$$

where X_{var} and Y_{var} are the controllability and observability variational Grammians of

the perturbed system. The largest eigenvectors of the product $X_{var}Y_{var}$ are now used as a basis for the subspace which is invariant to perturbations ΔA to system (2.5).

It was later noted elsewhere [15] that it is possible to compute the variational Grammians in the time domain rather than the frequency domain, thus eliminating the need to solve Lyapunov equations. Again letting X be the controllability Grammian, it is known that

$$X = \int_0^\infty e^{At} B B^T e^{A^T t} dt \tag{2.8}$$

$$= \int_{-\infty}^\infty (j\omega I - A)^{-1} B B^T (j\omega I - A)^{-H} d\omega \tag{2.9}$$

where Parseval's theorem is used to proceed from the first equation to the second. Considering the system

$$[sC + G]\, x = Bu, \tag{2.10}$$

the variational Grammian is then computed by exciting the system with white noise over some domain S_λ of probability density resulting from manufacturing variations in the system. This amounts to the equation

$$X_\lambda = \int_{S_\lambda} \int_{-\infty}^\infty (j\omega C + G)^{-1} B B^T (j\omega C + G)^{-H} p(\lambda) d\omega d\lambda \tag{2.11}$$

to compute the variational Grammians. Unlike moment matching schemes, the size of the ROM created with TBR does not depend on the number of parameters.

Also possible for PMOR of linear systems are methods which try to fit the transfer function of the linear system using optimization techniques [6, 16, 17, 18]. One such approach [6] attempts to fit, using measured data, a parameterized transfer function $h(t, p_1, p_2)$ with a trigonometric polynomial

$$h(t, p_1, p_2) \approx \hat{h}(t, p_1, p_2) = \frac{\sum_k b_k(p_1, p_2) \cos(kt) + j \sum_k c_k(p_1, p_2) \sin(kt)}{\sum_k a_k(p_1, p_2) \cos(kt)}. \tag{2.12}$$

A rational fit is used to solve for the parameter dependent coefficients a_k, b_k, c_k in the transfer function approximation. In this case the order of the reduced model is

determined by the order of the rational fit, and the resulting ROM is guaranteed stable for all parameter values.

Several non-parameterized Model Order Reduction (MOR) approaches are available for nonlinear systems. For instance, the reduction of weakly nonlinear systems has been shown using Volterra series and moment matching techniques [19, 20, 21, 22, 23, 24], while the reduction of strongly nonlinear systems has been shown using trajectory piecewise techniques [25, 26, 1, 2, 27, 28].

One of the first works in model order reduction for nonlinear systems [19] combined a Volterra series expansion with an Asymptotic Waveform Evaluation (AWE) approach. Using Volterra responses, the transient response of a nonlinear system can be easily approximated by repetitively computing the transient response of a linear equivalent circuit with different sources. Once the linear system and sources are found, a q-pole approximation to the linear system can be used to speed up transient simulation of the linear system. The examples considered in this work contained large numbers of linear lumped elements and mildly nonlinear terminations.

The simplest approximation to a nonlinear system is a single linear system. The authors in [20] use a moment matching scheme to create ROMs for a nonlinear MEM switch (considered in Section 4.3) by approximating the nonlinearity with a linear function created by expanding about an equilibrium point. For the projection matrix, moments are generated using the Arnoldi process on the linear system. Results showed that for small inputs the linear model provides a decent approximation. A second order approximation to the nonlinear system is later considered in an attempt to approximate the system response to large inputs, but for this case even the second order approximation fails to accurately capture the nonlinear effects.

A similar but more general approach to approximating weakly nonlinear systems would be to use a low-order polynomial expansion of the nonlinear function [21, 22]

$$\frac{dx}{dt} = f(x) + Bu \approx \sum_{k=1}^{m} \phi_k(x^{(1)}, \ldots, x^{(m)}) \tag{2.13}$$

where $x^{(m)}$ is the m^{th} Kronecker product of x with itself (e.g. $x^{(3)} = x \otimes x \otimes x$). Then

the expansion of $f(x)$ can be written as

$$f(x) \approx A_1 x^{(1)} + \cdots + A_m x^{(m)}, \tag{2.14}$$

which leads to the state space systems

$$\frac{dx_1}{dt} = A_1 x_1 + Bu \tag{2.15}$$

$$\frac{dx_2}{dt} = A_1 x_2 + A_2(x_1 \otimes x_1) \tag{2.16}$$

$$\frac{dx_3}{dt} = A_1 x_3 + A_2 \left[x_1 \otimes x_2 + x_2 \otimes x_1\right] + A_3 \left[x_1 \otimes x_1 \otimes x_1\right] \tag{2.17}$$

for $m = 3$. This new set of systems is linear in the variables x_1, x_2, x_3, because the latter cross terms are treated as additional inputs. Krylov vectors can now be computed for each of these linear systems for each input, and the resulting moments form the basis of an important subspace for this system. The linear systems can now be projected into the subspace to obtain a set of low-order linear models.

A drawback of the above method is that the order of the reduced system becomes q^m for an m^{th} order expansion where x_1 is size q. One possible approach to continue increasing accuracy without increasing the complexity of the reduced system is to identify a more optimum subspace for projection. The authors in [24] use the above polynomial approximation to the nonlinear function, but consider matching the moments of higher order transfer functions for the nonlinear system which arise from a Volterra series expansion of the system. This is in contrast to the previous approach which merely used moments from a the first order transfer function of the three linear systems in (2.15). The authors present a recursive algorithm for computing an arbitrary number of moments from an arbitrary order transfer function for a nonlinear system.

For highly nonlinear systems it is not possible to accurately approximate the nonlinearity with a single linear system, so a collection of linear systems must be used. This idea is used in the TPWL technique [26]. The nonlinear system is approximated

by a weighted sum linear systems

$$\frac{\partial \hat{x}}{\partial t} = \sum_i \hat{w}_i(x) \left[\hat{K}_i + \hat{A}_i x \right] + \hat{b}u \tag{2.18}$$

where the weights $w_i(x)$ are functions of the current state. The linear systems are chosen by linearizing the nonlinear system at important points in the state space. The important state space points are found by simulating the nonlinear system and choosing points along the trajectory where the model behaves very nonlinearly. The linear systems can now be projected into a low-order subspace. The projection matrix which defines this subspace can be constructed with a moment matching scheme [26] or with TBR [2]. One extension to this method is a PieceWise Polynomial (PWP) approach which uses a collection of second order approximations to the nonlinear system [27, 28], yielding an ROM of the form

$$\frac{\partial \hat{x}}{\partial t} = \sum_i \hat{w}_i(x) \left[\hat{K}_i + \hat{A}_i x + \hat{H}_i x \otimes x \right] + \hat{b}u. \tag{2.19}$$

One major drawback of the trajectory piecewise methods is that the accuracy of the reduced model can be highly dependent on the "richness" of the inputs chosen for training.

2.2 Relevant Model Order Reduction

2.2.1 Moment Matching for Linear Systems

Consider a linear system

$$\frac{dx}{dt} = Ax(t) + bu(t), \quad y(t) = c^T x(t), \tag{2.20}$$

where the state x has a very large order N. In the frequency domain, the system becomes

$$sX(s) = AX(s) + bu. \tag{2.21}$$

One approach to reduce the order of the system is with an orthonormal projection matrix V such that $x \approx V\hat{x}$, and $\hat{x}$ is order $q << N$. The columns of V define the basis of the reduced space and are carefully chosen to preserve the input/output relationship (i.e. transfer function) of the system.

To find a good basis for V, begin by rearranging (2.21) and solving for $X(s)$, to obtain

$$X = (sE - A)^{-1}bu. \tag{2.22}$$

By defining $\tilde{A} = A^{-1}E$ and $\tilde{b} = -A^{-1}b$, we obtain

$$X = (I - s\tilde{A})^{-1}\tilde{b}. \tag{2.23}$$

It is now easy to take a Taylor series expansion of (2.23) about $s = 0$, resulting in

$$X = \sum_{k=0}^{\infty} s^k \tilde{A}^k \tilde{b} = \tilde{b} + s\tilde{A}\tilde{b} + s^2\tilde{A}^2\tilde{b} + \dots \tag{2.24}$$

where the vectors $[\tilde{b}, \tilde{A}\tilde{b}, \tilde{A}^2\tilde{b}...]$ are referred to as the moments of the expansion. Close to the expansion point $s = 0$ the first q terms of (2.24) are a good approximation to $X(s)$. This approximates $X(s)$ as a linear combination of the first q moments in the expansion, thus those moments form a basis for a reduced space in which $X(s)$ can be well approximated. To obtain this important subspace, the projection matrix should be constructed such that

$$\text{colspan}(V) \subset \{\tilde{b}, \tilde{A}\tilde{b}, \tilde{A}^2\tilde{b}..., \tilde{A}^{q-1}\tilde{b}\}. \tag{2.25}$$

The columns of the projection matrix should be orthonormalized as they are added to V to avoid problems which arise from computing the moments using finite precision arithmetic. As q increases the terms $\tilde{A}^q\tilde{b}$ will converge to the largest eigenvector of $\tilde{A}$. The result of this is that for $q > 15$, the columns of V begin to become linearly dependent, and thus adding additional columns does not increase the order of the reduced space. This orthonormalization can be done with the Arnoldi algorithm,

which ensures $V^T V = I$, where I is the identity matrix.

Replacing X in (2.21) with $V\hat{X}$ results in

$$sEV\hat{X} = AV\hat{X} + bu. \tag{2.26}$$

This new system has q unknowns but is overdetermined with N equations. One method to reduce the number of equations is to left multiply the system by V^T. This results in

$$s\hat{E}\hat{X} = \hat{A}\hat{X} + \hat{b}u. \tag{2.27}$$

where $\hat{A} = V^T AV$, $\hat{b} = V^T b$, and $\hat{c} = cV$. Using this projection matrix to approximate $X \approx V\hat{X}$ guarantees that the transfer function of the reduced system, $\hat{H}(s)$, will match exactly q moments of the transfer function of the large system, $H(s)$, around the expansion point $s = 0$ [29]. Using V^T to reduce the number of equations preserves stability and passivity of the model, because $V^T AV$ is a congruence transform, and thus preserves definiteness of the matrix A.

The above considered the case where $X(s)$ was approximated by an expansion about $s = 0$, however $X(s)$ can easily be expanded about any point. For example, first define a new variable $\tilde{s}$ such that $s = \tilde{s} + s_0$. Expanding about $\tilde{s} = 0$ is now equivalent to expanding about $s = s_0$. The model becomes

$$\begin{aligned} (\tilde{s} - s_0)EX &= Ax + bu & (2.28)\\ [\tilde{s}E + s_0E - A]\,X &= bu & (2.29)\\ -\left[I - s(s_0E - A)^{-1}E\right]X &= bu. & (2.30) \end{aligned}$$

Define the new matrix $\tilde{A} = (s_0E - A)^{-1}E$ and the vector $\tilde{b} = (s_0E - A)^{-1}b$, and proceed as above following (2.23).

2.2.2 Parameterized Model Order Reduction

Consider a linear system whose dynamical descriptor matrices in Laplace domain are functions of the Laplace frequency variable s, and of some other geometrical parameters, $p_1, ..., p_\mu$,

$$sEX = A(p_1, ..., p_\mu)X + bu. \tag{2.31}$$

In order to preserve parameter dependence in the ROM after projection, the system must be of the form

$$sEX = \sum_i [g_i(p)A_i]X + bu \tag{2.32}$$

where $g_i(p)$ are scalar functions of the parameters, and A_i have no parameter dependence. This form can be attained, for example, by linearization or with a polynomial fitting scheme. Introducing additional parameters $\tilde{s}$, as shown in [10, 11], the approximation becomes

$$sEX = [A_0 + \tilde{s}_1 A_1 + ... + \tilde{s}_\mu A_\mu]X + bu. \tag{2.33}$$

As before, a good basis for the reduced system can be found by approximating X with a Taylor series expansion. This time treating s and $\tilde{s}_1, \ldots, \tilde{s}_\mu$ as variables and using a multivariable expansion yields

$$X = [I - \tilde{s}_1 M_1 - ... - \tilde{s}_p M_p]^{-1} b_M u \tag{2.34}$$

$$= \sum_n (\tilde{s}_1 M_1 + ... + \tilde{s}_\mu M_\mu)^n b_M u \tag{2.35}$$

where $M_0 = A_0^{-1}E$, $M_i = A_0^{-1}A_i$, and $b_M = A_0^{-1}b$. Keeping the first q terms in the expansion guarantees the transfer function of the reduced system $H(s, \tilde{s}_1, ..., \tilde{s}_\mu)$ will match a total of q moments in both the s variable and the parameter variables at expansion point ($s = \tilde{s}_1 = \ldots = \tilde{s}_\mu = 0$) of the large system transfer function if V is constructed such that

$$\text{colspan}(V) \subset \{F_0 b_M,\ F_1 b_M, ..., F_{q-1} b_M\}, \tag{2.36}$$

where detailed recursive formulae for the calculation of F_i can be found in [10]. In the same article it is noted that for a large number of parameters $\tilde{s}$ and a modest number of moments m matched for each parameter, this method may generate systems of substantial order $O(\mu^m)$.

The expansion in (2.34) may also be performed about different frequencies $s \neq 0$ and parameters $p \neq 0$. This can be achieved in the same manner as in the previous section, by defining $s = \bar{s} + s_0$ and $\tilde{s}_i = \bar{\tilde{s}}_i + \tilde{s}_{0i}$ where s_0 and $\tilde{s}_{0i}$ are the desired expansion points, and then expanding about the new variables at zero, $\bar{s} = 0$ and $\bar{\tilde{s}} = 0$. This changes the exact formulae for M_i and F_i, but the procedure remains the same.

2.2.3 Nonlinear Model Order Reduction

Consider a nonlinear system in the form

$$\frac{dx}{dt} = F(x(t)) + bu(t), \qquad y = c^T x(t). \tag{2.37}$$

Applying the usual projection scheme with projection matrix V such that $x \approx V\hat{x}$ results in

$$\frac{d\hat{x}}{dt} = V^T F(V\hat{x}(t)) + \hat{b}u(t). \tag{2.38}$$

Since $F(x)$ is a nonlinear function, the term $F(V\hat{x}) \approx F(x)$ is still a function evaluation on a vector of length N. Evaluating this term at every timestep in a reduced order model would be too expensive. It appears projection is cheap only when the system is linear in the state variables, so it is possible to linearize (2.37) about some state x_0 to obtain the linear model

$$\frac{dx}{dt} = A_0 x + K_0 + bu(t) \tag{2.39}$$

where A_0 is the Jacobian of $F(x)$ at $x = x_0$, and $K_0 = F(x_0) - A_0 x_0$. This linear system can now be reduced using standard projection techniques. However, due to the linearization, this model is only accurate in the state-space near x_0, and there

is no reason to believe the state will remain in that region during simulation. To account for this, it will be necessary to create multiple linearized models from (2.37) at different state-space points $x_1, ...x_k$. The different linear models will be accurate in different regions of the state space. The nonlinear system can now be approximated by a weighted combinations of the linear models,

$$\frac{dx}{dt} = \sum_{i=1}^{k} w_i(x, X) \left[A_i x + K_i\right] + bu(t) \tag{2.40}$$

where $w_i(x, X)$ are some weighting functions which depend on the proximity of the state x with respect to the linearization points $X = [x_1, x_2, ..., x_k]$. To ensure the model is accurate in all regions of the space, linearizations would need to be created uniformly across the space. Assuming a grid of 10 linearizations per dimension in a space of dimension $N = 10,000$, a total of 10^{10000} models would be required. This is computationally infeasible.

A better approach is to create models only in regions of the space where the state is likely to travel during simulation. The trajectory piecewise linear (TPWL) method [1] uses typical 'training' inputs to drive the state through the state-space creating training trajectories. These training trajectories are then compared to trajectories arising from the simulation of linearizations of the nonlinear system, and in regions along the trajectory where the model behaves very nonlinear, a linearization is made. Once a collection of linear models is amassed, each linear system can be projected into a subspace with the standard projection technique. The projection matrix is constructed using moments (2.25) from each linear model created during training. The final reduced system is then

$$\frac{d\hat{x}}{dt} = \sum_{i=1}^{k} \hat{w}_i(\hat{x}, \hat{X}) \left[\hat{A}_i \hat{x}(t) + \hat{K}_i\right] + \hat{b}u(t) \tag{2.41}$$

where $\hat{A}_i = V^T A_i V$, $\hat{K}_i = V^T K_i$, $\hat{b} = V^T b$, $\hat{x} = V^T x$, $\hat{X} = [V^T x_1, ..., V^T x_k]$, and k is the number of linear models. The resulting model (2.41) is still a nonlinear system because of the state-dependence of the weighting functions, but simulation is cheap

because the order of the model is $q << N$. The relative weights, $\hat{w}_i(\hat{x}, \hat{X})$, of each linear model vary dynamically as the state evolves. One example of possible weighting functions is

$$\hat{w}_i(\hat{x}, \hat{X}) = \frac{exp[\frac{-\beta d_i}{m}]}{\sum_i exp[\frac{-\beta d_i}{m}]}. \tag{2.42}$$

where β is some constant (typically around 25), $d_i = ||\hat{x} - \hat{x}_i||$, and $m = min_i(d)$. This is the weighting scheme used in [1]. Several other weighting schemes have been introduced recently [30, 31] in an attempt to ensure smoothness of the ROM output. As was shown in [30], only the closest linearization points have nonzero weights at any given time, so rather than computing several hundred or thousand weights which are zero, it is more efficient to compute only the five closest weights and set the others to zero. This saves computation time and results in little or no loss of accuracy.

Chapter 3

Parameterized Model Order Reduction for Nonlinear Systems

3.1 Proposed Method

3.1.1 PROM Description Derivation

A system possessing nonlinear dependence on both the state $x(t)$ and some parameters p_i may be of the form

$$\frac{dx}{dt} = F(x(t), p_1, p_2, ..., p_\mu, u). \tag{3.1}$$

If the vector-valued function in (3.1) can be written as

$$F(x(t), p_1, p_2, ..., p_\mu, u) = \sum_{j=1}^{\mu} g_j(p) f_j(x, u) \tag{3.2}$$

where $g_j(s)$ are scalar functions, then the parameters can be redefined as $\tilde{s}_j = g_j(s)$ to obtain

$$\frac{dx}{dt} = \sum_j \tilde{s}_j f_j(x, u). \tag{3.3}$$

To account for terms with no parameter dependence, define $\tilde{s}_0 = 1$. If the system cannot immediately be written as (3.2), then some polynomial fitting scheme or Taylor series approximation must be used to capture any nonlinear parameter dependence,

resulting in

$$\frac{dx}{dt} = \sum_j f(p, x, u) \approx \sum_j \tilde{s}_j \tilde{f}_j(x, u). \tag{3.4}$$

Due to such approximation, this model is valid only in some parameter space region near the linearization or fitting points.

With the system in the forms form of (3.4), the nonlinear functions $f_j(x, u)$ can be approximated as linear functions without affecting the parameter dependence. Consider, for example,

$$\frac{dx}{dt} = f_1(x) + \tilde{s} f_2(x) \tag{3.5}$$

A linear approximation of $f_1(x)$ and $f_2(x)$ about some point x_0 gives

$$\frac{dx}{dt} \approx [f_1(x_0) + J_1(x - x_0)] + \tilde{s}\,[f_2(x_0) + J_2(x - x_0)] \tag{3.6}$$

where the system still has the same structure with respect to the parameters. This allows, as in [1], approximation of the nonlinear functions $f(x, u)$ as a collection of local linearizations around different points x_i in the state space:

$$\frac{dx}{dt} = \sum_{i=1}^{k} \sum_{j=1}^{p} w_i(x, X) \tilde{s}_j [\tilde{A}_{ij} x(t) + \tilde{K}_{ij} + B_j u(t)] \tag{3.7}$$

where $w_i(x, X)$ are weighting functions which vary dynamically with the state.

Taking advantage of the linear parameter dependence, standard projection techniques can be applied to each of the linear systems in (3.7). For instance, using a projection matrix V, the system becomes

$$\frac{d\hat{x}}{dt} = \sum_i \sum_j \hat{w}_i(\hat{x}, \hat{X}) \tilde{s}_j [\hat{A}_{ij} \hat{x}(t) + \hat{K}_{ij} + \hat{B}_j u(t)] \tag{3.8}$$

$$y = \hat{c}^T \hat{x}(t) \tag{3.9}$$

where $\hat{A}_{ij} = V^T \tilde{A}_{ij} V$, $\hat{K}_{ij} = V^T \tilde{K}_{ij}$, $\hat{b_j} = V^T b_j$, $\hat{c} = cV$, $x(t) = V\hat{x}(t)$, and $\hat{X} = [V^T x_1, ..., V^T x_k]$. This reduced system is order q but still possesses the original linear dependence on the parameters.

In order to complete the procedure, two algorithms remain to be specified: how to choose linearization points x_i, and how to construct projection matrix V. These two methods will be discussed in detail in the following sections, and then combined to create the proposed Nonlinear Parameterized Model Order Reduction (NLPMOR) algorithms.

3.1.2 Selecting Linearization Points

In standard TPWL [25, 26, 1, 27, 28] linearization points are chosen along state trajectories generated by typical training inputs. Using a similar idea, additional trajectories can be created by training at a set of points in the parameter space. This enriches the collection of linearization points so that linear models are available near state-space regions where variations in the parameter are likely to take the model. Training at different points in the parameter space to collect linearization points will be referred to in this thesis as "training at multiple parameter-space points". The TPWL scheme training only at the nominal parameter value will be referred to in this thesis as "training at a single parameter-space point".

It is important to note here that the training trajectories must be generated by a system which is already linear in all of the parameters (3.2). The purpose of training at different parameter values is not to create different linearizations in the parameter effectively sampling the parameter dependence, but rather to drive the state to different regions of the state space. Suppose two training trajectories are created by training with two different nonlinear models. If both systems are linearized at the same state x_0 on the trajectories, then it is possible that the resulting two linear models will be different. This is discussed in more detail in Section 3.1.4.

Additional training trajectories add linear models but do not affect the order of the reduced system. Since the weighting functions in (3.8) are typically nonzero for just a few models at any particular time, only the closest models are considered for weighting and a larger set of models does not significantly affect simulation time.

3.1.3 Constructing the Projection Matrix

As in PMOR for linear systems [9, 10, 11] the columns of V can be constructed from a multivariable Taylor series expansion about each parameter $\tilde{s}_j$. The moments may be matched at a set of points in the parameter space which lie within some typical range of parameter values.

This is similar to the scheme used in section 2.2.2, but in this case the model is nonlinear. Therefore, the projection vectors are taken from the Taylor expansion of the linearized models created during training. Such vectors however are not necessarily the best basis for the reduced space. It is difficult to say a priori which linearized models will provide the best basis for the reduced space. Hence, it may be beneficial to create Krylov vectors at every single point along the training trajectories. This does not significantly increase the computational cost because solving the nonlinear system with an implicit time integration scheme (e.g. Newton's method with Backward Euler) requires linearizations at every time step, so the additional cost is only a few system solves per step.

One other small difference arising in the nonlinear case is the constant vector 'K' in (3.8)- an artifact of the state linearizations. This term can be treated as a second input term, i.e. 'B_2', with constant input, i.e. $u_2 = 1$. To account for this term, several Krylov vectors should also be generated from the set $[K, AK, A^2K, ...]$ at each linear model.

Matching moments about multiple expansion points for every linear model quickly increases the size of the projection matrix V. If V becomes too large, simulation of the reduced order model will become costly. To keep the order of the reduced system small, an SVD may be performed on the projection matrix. The SVD is relatively inexpensive because the projection matrix is very tall, but also very "skinny". After SVD, only vectors corresponding to the largest q singular values remain, resulting in a reduced system of order q.

3.1.4 Linearizing in the Parameters

It was stated in Section 3.1.2 that if (3.1) is nonlinear in p, it must be linearized in $\tilde{s} = g_i(p)$ to preserve the parameter dependence in the ROM. It was also claimed that all of the training must be done with system 3.4. This is necessary because linearizations in the parameter may alter the state dependence of the system. This makes it possible to obtain two different models from linearizations at the same state x. Consider, for example, linearizing (3.1) at two different parameter space points p_1 and $p_2 = 0$ resulting in the two systems

$$F(x,p) \approx F(x,p_1) + \frac{dF(x,p_1)}{dp}(p - p_1) \tag{3.10}$$

$$F(x,p) \approx F(x,0) + \frac{dF(x,0)}{dp}(p). \tag{3.11}$$

These models would then be used for training, and linearized again in the state x, resulting in equations with a constant term, a term linear in p, a term linear in x, and a term linear in the product xp. If the two models from the two linearization points are the same they must be equal at all values of p and x, which means the constant terms must be equal. This requires $F(x_0,p_1) - p_1\frac{dF(x,p_1)}{dp} = F(x_0,0)$, but this is clearly not true for any arbitrary function $F(x,p)$ (e.g $F(x,p) = e^{xp}$).

Suppose two different models are generated at the same state x_i. The PROM now contains two different linear models for the same state; one which is accurate when the simulation parameter value is near p_1, and another which is accurate when the parameter value is near p_2. One way to ensure the PROM uses the correct model is to associate the linear models with the parameter linearizations from which they came. This would require an additional sum in the final system over parameter linearizations

$$\frac{dx}{dt} = \sum_k \sum_i \sum_j w_{i,k}(x,X)\tilde{s_j}[A_{k,ij}x(t) + K_{k,ij}] + bu(t). \tag{3.12}$$

This increases the cost of simulating the PROM, and the general assumption is that only small perturbations in the parameter will be made, so a single parameter space linearization in sufficient in these examples.

	Training at Single Point in Parameter Space	Training at Multiple Points in Parameter Space
MOR Moment Matching in V	TPWL	Algorithm 2
PMOR Moment Matching in V	Algorithm 1	Algorithm 3

Table 3.1: The 4 options for creating a ROM using our two techniques on a nonlinear system

Algorithm 1 Trained at Single Parameter Point with PMOR Moment Matching

1: **for** $m = 1$: Number of Inputs **do**
2: Linearize at initial state x_0
3: **while** $t < t_{final}$ **do**
4: **if** $\| x_t - x_i \| < \delta$ **then**
5: Simulate current linear model to next time t
6: **else**
7: Linearize at current state and set $x_0 = x_t$
8: Use equations (2.34) and (2.36) to create vectors for V
9: **end if**
10: **end while**
11: **end for**
12: Perform SVD on projection matrix V
13: Project systems using equation (2.41)

3.2 Proposed NLPMOR Algorithms

Combining the parameterization options in sections 3.1.2 and 3.1.3 gives rise to four different algorithms for parameterized model order reduction of a nonlinear system. The four options are presented in Table 3.1. The first option, TPWL, is essentially the TPWL algorithm if it were applied to a system which was already linear in some set of parameters. For this reason this algorithm will not be discussed in detail, but will be used for comparison purposes.

The first extension of the TPWL algorithm in Table 3.1, shown in Algorithm 1, combines the TPWL training scheme with parameterized moment matching for the projection matrix. The second option, described in Algorithm 2, does not match moments for the parameters, but trains the system at multiple points in the parameter space. The final system is now a collection of linear models created by training

Algorithm 2 Trained at Multiple Parameter Values with MOR Moment Matching

```
for m = 1: Number of Inputs do
  for p = 1: Number of Parameter Training Values do
    Linearize at initial state x_0
    while t < t_final do
      if || x_t − x_i ||< δ then
        Simulate current linear model to next time t
      else
        Linearize at current state and set x_0 = x_t
        Use equations (2.24) and (2.25) to create vectors for V
      end if
    end while
  end for
end for
Perform SVD on projection matrix V
Project systems using equation (2.41)
```

at different sets of parameter values. The final method, described in Algorithm 3, combines the two previous methods. The system is trained at multiple points in the parameter space, and the models projection matrix is built with moments generated from a multivariable Taylor series expansion to account for parameter changes.

Since training trajectories are expensive, the cost may be reduced by using "approximate training trajectories". For an exact training trajectory, let x_t be the state of the original nonlinear system at time t and x_i be the state at time t of the linear system created by linearizing at x_0. In this case, linear models are created when x_t strays from the exact state at that time step x_i, that is, when $||x_t - x_i|| > \delta$. For the approximate trajectory, let x_t be the state at time t of the linear system created by linearizing at x_0, and let x_i be the most recent linearization point x_0. Linear models are then created when the state x_t strays from the region of space where linearization x_i is valid, that is, $||x_t - x_i|| > \delta$. When used with approximate trajectories the cost of training is reduced to $O(Tr^p)$, however models created at points where the state approximately evolves may be less useful. All of the algorithms can be modified to incorporate this training method, but only exact training trajectories are considered in the rest of this work.

Algorithm 3 Trained at Multiple Parameter Values with PMOR Moment Matching

```
for m = 1: Number of Inputs do
  for p = 1: Number of Parameter Training Values do
    Linearize at initial state x_0
    while t < t_final do
      if || x_t − x_i || < δ then
        Simulate current linear model to next time t
      else
        Linearize at current state and set x_0 = x_t
        Use equations (2.34) and (2.36) to create vectors for V
      end if
    end while
  end for
end for
Perform SVD on projection matrix V
Project systems using equation (2.41)
```

Chapter 4

Examples

4.1 Overview

Three examples were chosen to test the NLPMOR algorithms. All three contain distributed strong nonlinearities and possess dependence on some geometrical parameters. For each example a derivation of the system model is presented, followed by some results from the algorithms for different parameters.

4.2 Diode Transmission Line

4.2.1 System Description

The first example considered is a nonlinear transmission line with diodes. This example was chosen because it was used in the original TPWL papers [1, 2]. This provides some relative accuracy comparisons for the new method. The transmission line, shown in Figure 4-1, is a nonlinear analog circuit containing a chain of strongly nonlinear diodes, resistors and capacitors. Choosing the system state to be the nodal voltages, the system equations are easily derived using Kirchoff's current law and nodal analysis. An equation for interior node j then has the form

$$C\frac{dx_j}{dt} = \frac{x_{j-1} - 2x_j + x_{j+1}}{R} + I_d[e^{\alpha(x_{j-1}-x_j)} - e^{\alpha(x_j - x_{j+1})}] \tag{4.1}$$

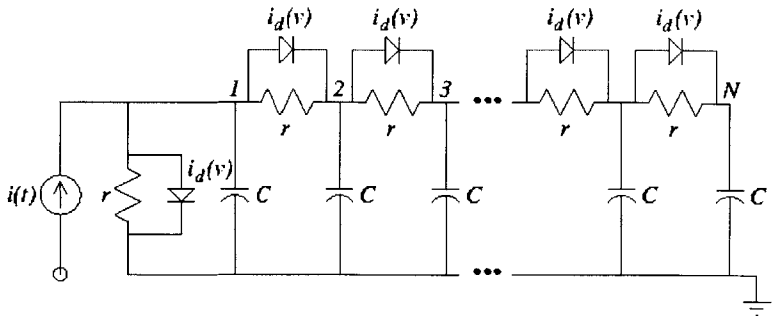

Figure 4-1: A nonlinear transmission line circuit containing diodes [1, 2].

leading to a state space system of the form

$$E\frac{dx}{dt} = Gx + D(x) + bu(t). \tag{4.2}$$

Here G is the conductance matrix, E is the capacitance matrix, $D(x)$ is a vector valued function containing the constitutive relations for the diodes, and b is the input vector, and the input is an ideal current source $u(t) = i(t)$. All resistors have value 1Ω and all capacitors are $10pF$. The constitutive relation for the diodes is $I(V) = I_d(e^{\alpha V} - 1)$, where α is defined as $\frac{1}{v_t}$, and v_T is the threshold voltage. Values of $I_d = 0.1nA$ and $\alpha = 40$, which corresponds to $v_t = 25mV$, were used.

4.2.2 Results

Parameters of interest in this system are the capacitor values, the resistor values, and some diode parameters such as "turn-on" voltage v_T and saturation current I_d. These parameters can affect the characteristics of travelling waves on this transmission line. All models in this section were trained with a sinusoidal input, $u(t) = (cos(\omega t) + 1)/2$ with $\omega = 2\pi GHz$. For PMOR moment matching, the projection matrix was built with moments from a Taylor series expansion (equation (2.34)) about the frequency $s_0 = j2\pi GHz$ and any parameter values used for training. The frequency expansion point $s_0 = j2\pi GHz$ was also used for MOR moment matching.

The first parameter considered is the diode threshold voltage v_T. To simplify the

equations, the parameter is defined as $\alpha = \frac{1}{v_T}$. The original system (4.1) is nonlinear in α, so (4.2) is linearized about some nominal value α_0 to obtain the state space model

$$\frac{dx}{dt} = Gx + D(x, \alpha_0) + \frac{\partial}{\partial \alpha}\left[D(x, \alpha)\right](\alpha - \alpha_0) + bu(t). \tag{4.3}$$

Note that the system is still nonlinear in the state. A model was created using sinusoids as training inputs and a nominal parameter value of $\alpha = 40$ for linearization and training. As mentioned before, linearizing at α_0 introduces some initial error as the parameter changes from α_0. Figure 4-2 compares the simulation output of the

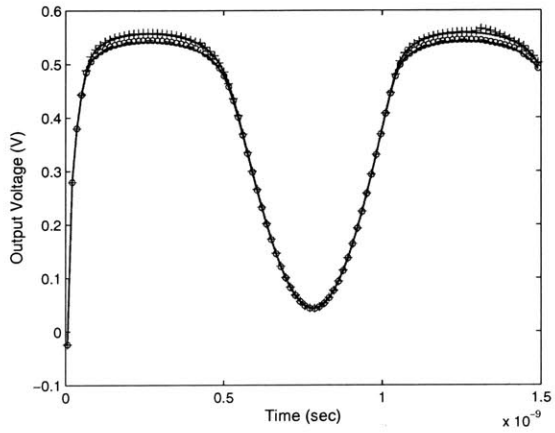

Figure 4-2: A model created using TPWL for the diode transmission line parameterized in $p = \frac{1}{v_t}$ using $p_0 = \frac{1}{40}$ is simulated at $p = 0.98p_0$ (circles) and at $p = 1.02p_0$ (plusses), and then compared to output of the full nonlinear system (solid lines). The model was reduced from large order $N = 200$ to reduced order $q = 20$.

full nonlinear systems with that of a PROM at two different parameter values.

The second set of parameters considered are the resistor values and the diode saturation current I_d. To keep linearity in parameters, define $p_1 = \frac{1}{R}$ as the parameter value instead of R, and let $p_2 = I_d$. In this case no linearizations in the parameter are required, resulting in the model

$$\frac{\partial x}{\partial t} = p_1\tilde{G} + p_2\tilde{D}(x) + bu(t), \tag{4.4}$$

where $\tilde{G}$ and $\tilde{D(x)}$ are defined similarly to the terms in (4.2), except the parameters

are removed. A PROM was created by training with sinusoidal inputs at parameter points near $I_d = 0.1nA$ and $R = 1\Omega$. Figure 4-3 examines the error of the PROM as

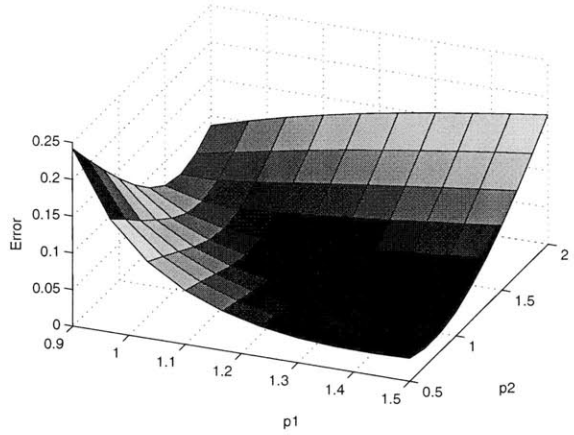

Figure 4-3: The error resulting from simulations of a PROM created for the diode transmission line parameterized in $p_1 = \frac{1}{R}$ and $p_2 = I_d$. The mode was reduced from large order $N = 100$ to reduced order $q = 20$.

the two parameters are varied. The error plotted is the norm of $e(t)$, where

$$e(t) = |y(t) - \hat{y}(t)| \tag{4.5}$$

with $y(t)$ the full nonlinear system output and $\hat{y}(t)$ the PROM output.

4.3 Micromachined Switch

4.3.1 System Description

The second example is a micromachined switch [1, 2]. The switch consists of a polysilicon fixed-fixed beam suspended over a polysilicon pad on a silicon substrate as shown in Fig. 4-4. When a voltage is applied between the beam and the substrate, the electrostatic force generated pulls the beam down towards the pad. If the force is large enough, the beam will come into contact with the pad closing the circuit. In addition to being a switch, this device can also be used as a pressure sensor due to its extreme sensitivity to surrounding atmospheric conditions.

The unknowns of interest in this system are the deflection of the beam, $u(x,t)$, and the air pressure between the beam and substrate, $p(x,y,t)$. The system of equations

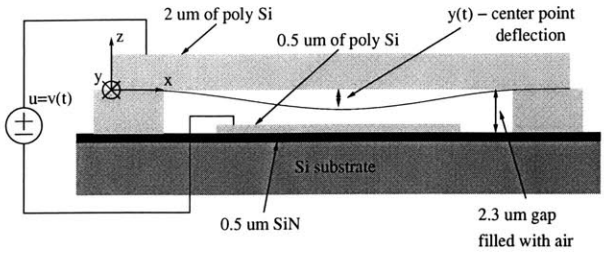

Figure 4-4: The MEM switch is a polysilicon beam fixed at both ends and suspended over a semiconducting pad and substrate [1, 2].

is assembled by discretizing the coupled 1D Euler's Beam Equation (4.6) and the 2D Reynold's squeeze film damping equation (4.7), taken from [1]. A finite difference scheme was used for the discretization, and since the length of the beam is much greater than the width, the vertical deflection is assumed to be uniform across the width and only pressure was discretized in the width.

$$\hat{E}I_0h^3w\frac{\partial^4u}{\partial x^4} - S_0hw\frac{\partial^2u}{\partial x^2} = F_{elec} + \int_0^w (p-p_a)dy - \rho_0hw\frac{\partial^2u}{\partial t^2} \tag{4.6}$$

$$\nabla\cdot((1+6K)u^3p\nabla p) = 12\mu\frac{\partial(pu)}{\partial t} \tag{4.7}$$

Here, $F_{elec} = -\frac{\epsilon_0 wv^2}{wu^2}$ is the electrostatic force across the plates resulting from the applied voltage v, while v^2 is the input to the system. The beam is $2.2\mu m$ above the substrate ($z_0 = 2.2\mu m$), $610\mu m$ in length, and has a width of $40\mu m$. The other constants are permittivity of free space $\epsilon_0 = 8.854 * 10^{-6}\frac{F}{m}$, permeability $\mu = 1.82 * 10^{-5}\frac{kg}{ms}$, moment of inertia $I_0 = 1/12$, Young's modulus $\hat{E} = 149GPa$, Knudsen number $K = \frac{\lambda}{z_0}$, $\lambda = 0.064$, stress coefficient $S_0 = -3.7$, and density $\rho_0 = 2300\frac{kg}{m^3}$. By setting the state-space variables to $x_1 = u$, $x_2 = \frac{\partial u^3}{\partial t}$, and $x_3 = p$, the following

dynamical system results.

$$\begin{aligned}
\frac{\partial x_1}{\partial t} &= \frac{x_2}{3x_1^2} \\
\frac{\partial x_2}{\partial t} &= \frac{2x_2^2}{3x_1^3} + \frac{3x_1^2}{\rho_0 hw}[\int_0^w (x_3 - p_a)dy + S_0 hw\frac{\partial^2 x_1}{\partial x^2} \\
&\quad -EIh^3 w\frac{\partial^4 x_1}{\partial x^4}] - \frac{3\epsilon_0}{2\rho_0 h}v^2 \\
\frac{\partial x_3}{\partial t} &= -\frac{x_2 x_3}{3x_1^3} + \frac{1}{12\mu x_1}\nabla((1+6\frac{\lambda}{x_1})x_1^3 x_3 \nabla x_3)
\end{aligned}$$

The beam is fixed at both ends and initially in equilibrium, so the applied boundary conditions are

$$u(x,0) = u_0, p(x,y,0) = p_a, u(0,t) = u(l,t) = u_0. \tag{4.8}$$

Other constraints enforced are

$$\frac{\partial p(0,y,t)}{\partial x} = \frac{\partial p(l,y,t)}{\partial x} = 0, p(x,0,t) = p(x,w,t) = p_a, \tag{4.9}$$

where the initial height and pressure are $u_0 = 2.3\mu m$ and $p_a = 1.103 * 10^5 Pa$.

4.3.2 Results

For this example a designer may be interested in changing some geometrical properties of the device, such as the height and width, or possibly some material properties, such as Young's modulus E and stress coefficient S. For this system, PROMs were created by training using either a sinusoidal input with $u(t) = (v\cos(\omega t))^2$, and $\omega = \frac{10\pi}{30} MHz$, or a step input $u(t) = v^2$ for $t > 0$ with $v = 7$. The system output is the height of the beam center point. For both MOR and PMOR moment matching the projection matrix V was built with moments from a Taylor series expansion about DC $s_0 = 0$. PMOR moment matching used parameter expansion points equal to the parameter values used in training.

The first parameter considered is E, Young's modulus of the beam. This yields a state space system of the form

$$\frac{\partial x}{\partial t} = A_0(x) + EA_1(x) + bu(t). \tag{4.10}$$

A model was created by training at parameter values near $E_0 = 149,000$ with step inputs. The model was then simulated over a range of parameter values and compared to the output of the full nonlinear system. Shown in Figure 4-5 is the maximum percent error over time for each parameter value, i.e.

$$\frac{\max}{t}\left(\frac{|y(t) - \hat{y}(t)|}{|y(t)|}\right) * 100 \tag{4.11}$$

where $y(t)$ is the output of the full model and $\hat{y}(t)$ is the output of the PROM.

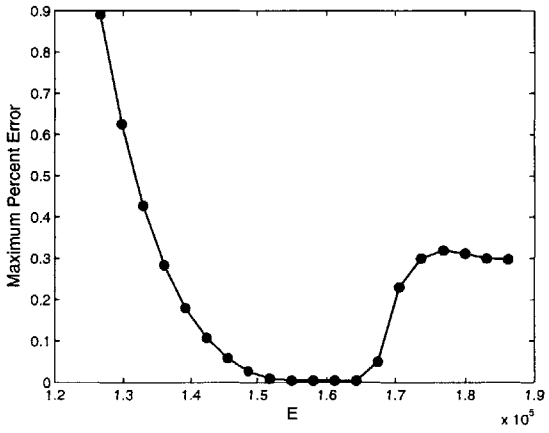

Figure 4-5: A PROM created with TPWL algorithm for the micromachined switch parameterized in E and simulated over a range of parameter values. For each value, the maximum percent error (4.11)is plotted for output resulting from a step input. This model was reduced from original order $N = 150$ to $q = 40$.

The second set of parameters considered are E and S. For this example a model was created by training with sine inputs at E_0 and $S_0 = -3.7$, and then it was simulated at three different sets of parameter values. The outputs of this simulation along with the output of the full nonlinear system are shown in Figure 4-6.

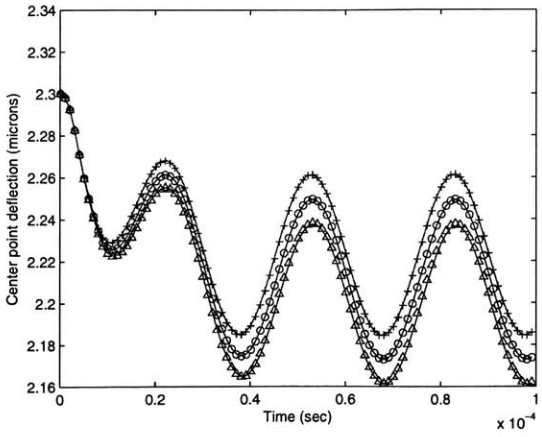

Figure 4-6: Output of a micromachined switch model parameterized in E and S simulated at three different sets of parameter values. The circles correspond to the nominal parameter values $[E, S] = [E_0, S_0] = [149,000, -3.7]$, the plusses correspond to $[E, S] = [1.05E_0, 0.98S_0]$, the triangles represent $[E, S] = [0.98E_0, 1.05S_0]$, and the solid linear are from the full nonlinear system. The model was reduced from large order $N = 70$ to reduced order $q = 20$.

4.4 Pulse Narrowing Transmission Line

4.4.1 System Description

The final example considered is a nonlinear transmission line used for signal shaping. One example of such a line, shown in Figure 4-7, contains distributed nonlinear capacitors. The resulting wave equation for this transmission line contains a nonlinear term which sharpens the peaks in a wave travelling down the line. Hence these devices may be useful in pulse narrowing applications. A thorough analysis of this line can be found in [3]. The nonlinearity arises from the voltage dependence of the capacitors, $C_n = C(V_n) \approx C_0(1 - b_c V_n)$. Setting the system state to the node voltages and branch currents, system equations can be derived using Kirchoff's current law and nodal analysis. The input is an ideal voltage source $u(t) = V_s(t)$, and the output is the voltage at some node m along the line, $y(t) = V_m(t)$. Using this formulation,

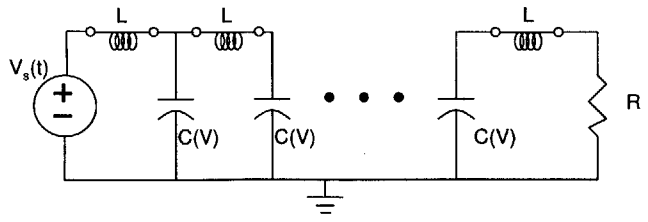

Figure 4-7: A pulse narrowing transmission line circuit containing nonlinear capacitors [3].

the system equations for an interior node n would be of the form

$$C_n(V_n)\frac{dV_n}{dt} = I_{n-1} - I_n \tag{4.12}$$

$$L_n\frac{dI_n}{dt} = V_n - V_{n+1} \tag{4.13}$$

leading to the state space model

$$\begin{bmatrix} \dot{V} \\ \dot{I} \end{bmatrix} = \begin{bmatrix} f_V(V,I) \\ f_I(V,I) \end{bmatrix} + \begin{bmatrix} 0 \\ B \end{bmatrix} u(t) \tag{4.14}$$

where the n^{th} equations of f_V and f_I are

$$f_{Vn} = \frac{I_{n-1} - I_n}{C_0 - b_c C_0 V_n} \tag{4.15}$$

$$f_{In} = \frac{V_n - V_{n+1}}{L}. \tag{4.16}$$

Here B_2 is the vector of voltage source inputs and typical capacitor and inductor values are 100 picoFarads and 100 picoHenries respectively.

4.4.2 Results

Quantities of interest for the pulse narrowing transmission line are the inductor values, the capacitor values, and b_c, a parameter which adjusts the nonlinearity of the

line. These three parameters all help shape the wave as it travels down the line, so all may need to be adjusted to design the optimum pulse narrowing line for some application. For this example PROMs were created by training with a sinusoidal input with $u(t) = v\sin(\omega t)$ with $\omega = 10\pi GHz$. The system output is the voltage at a node somewhere along the line, typically the 25^{th} node was used. For both MOR and PMOR moment matching the projection matrix V was built with moments from a Taylor series expansion about DC, and $s_0 = 10GHz$. PMOR moment matching generated moments about parameter expansion points equal to the parameter values used in training.

For this line, the first parameter considered is the inductor values. To keep things linear, the parameter is actually defined as $p = \frac{1}{L}$, resulting in the model

$$\begin{bmatrix} \dot{V} \\ \dot{I} \end{bmatrix} = p\left(\begin{bmatrix} f_V(V,I) \\ f_I(V,I) \end{bmatrix} + \begin{bmatrix} 0 \\ B_1 \end{bmatrix} u(t)\right) + \begin{bmatrix} g_V(V,I) \\ g_I(V,I) \end{bmatrix}. \tag{4.17}$$

This example also tests parameterization of the input vector B.

The model was created using sinusoidal training inputs and parameter values around $L = 10pH$. Three parameter values were then chosen for simulation, and the output of the PROM and full nonlinear system at those parameter values are compared in Figure 4-8.

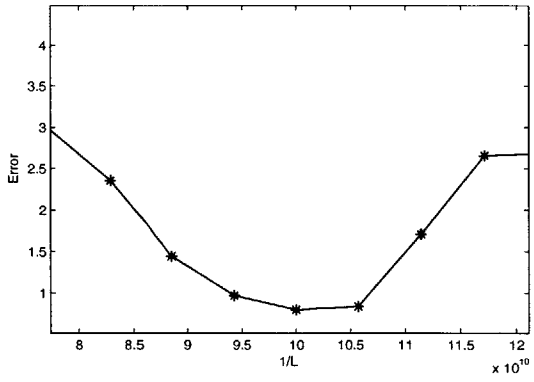

Figure 4-8: Error from a model of the pulse narrowing transmission line parameterized in $p = \frac{1}{L}$ compared to the full nonlinear system . The model was simulated at a range of parameter values around $p_0 = 10^{11}$. The model was built using TPWL trained at p_0 and was reduced from large order $N = 200$ to order $q = 50$.

The second set of parameters for this example are the capacitor values together with the inductor values. To preserve parameter linearity, the parameters are defined as $p_1 = \frac{1}{C}$ and $p_2 = \frac{1}{L}$.

$$\begin{bmatrix} \dot{V} \\ \dot{I} \end{bmatrix} = p_1 \left(\begin{bmatrix} f_V(V,I) \\ f_I(V,I) \end{bmatrix} + \begin{bmatrix} 0 \\ B_1 \end{bmatrix} u(t) \right) + p_2 \begin{bmatrix} g_V(V,I) \\ g_I(V,I) \end{bmatrix}. \tag{4.18}$$

The model was created from training with sinusoidal inputs at parameter values near $C = 10pF$ and $L = 10pH$. Figure 4-9 compares two outputs from this two-parameter

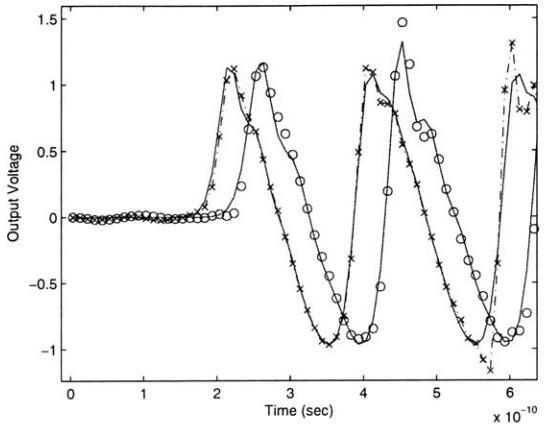

Figure 4-9: Output from a model of the pulse narrowing transmission line simulated at two different sets of parameter values. The model was trained at $[1/L, 1/C] = [p_{10}, p_{20}] = [10^1 1, 10^1 1]$ and was simulated at $[p_1, p_2] = [0.9p_{10}, 0.9p_{20}]$ (circles), and $[p_1, p_2] = [1.1p_{10}, 1.1p_{20}]$ (crosses). The solid lines represents the full nonlinear system of order $N = 100$, while the reduced model has order $q = 50$.

model to the outputs of the full nonlinear system at those same parameter values.

Chapter 5

Algorithm Analysis

5.1 Overview

In this chapter we examine and compare in detail the accuracy of the models generated by the four algorithms in Table 3.1. First we will take a careful look at the effects of approximating the nonlinear system with a collection of linear systems. This includes the limitations on the parameter-space accuracy imposed by linearizing the original system in the parameters. Second, we examine the choice of projection basis for the reduced system by comparing the methods for generating projection matrix columns. The two moment generating techniques considered are a single variable Taylor series expansion and a multivariable Taylor series expansion. Finally, we wish to determine whether or not the set of linear models from which the projection matrix vectors are generated has a significant affect on accuracy of the PROM.

Following the accuracy analysis, computational costs are considered for the different model construction methods. The order of complexity expressed as a function of the number of system solves is presented for each of the four algorithms in Table 3.1. Lastly, timing results are presented for transient simulations of the PROMs and are compared to simulations of the full nonlinear system. For these PROMs, the effects on simulation time produced by varying the reduced model order q and the number of linear models k are examined.

5.2 Parameter Space Accuracy

5.2.1 Benefits of Training in Parameter Space

To examine the benefit of training at different points in the parameter space (as in section 3.1.2), models created by TPWL and Algorithm 1 (shown in Table 3.1) are compared with models created by Algorithms 2 and 3. The benefit of training at multiple points in the parameter space is a vaster collection of linear models to choose from. Training with different parameter values will drive the state into different regions of the space and create different trajectories. This will explore regions of the space where a changing parameter takes the state during simulation. The PROM may need these models to accurately represent parameter changes. This method does not add any additional parameterization to the models. The models are still linearized at the same parameter value.

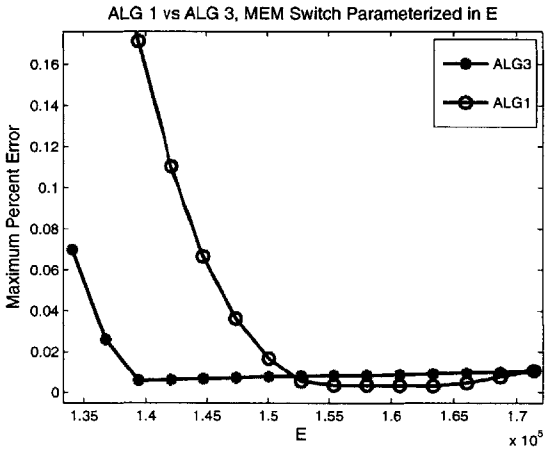

Figure 5-1: Models created for micromachined switch parameterized in E using Algorithms 1 and 3 were created and simulated over a range of parameter values. Each model was reduced from large order $N = 150$ to reduced order $q = 30$.

The first comparison examines reduced order models of the Micromachined Switch created using Algorithms 1 and 3. Treating E as the parameter, two models were created by training at $E = E_0 = 149,000$ and $E = 0.95E_0, 1.05E_0$ for Algorithms 1 and 3 respectively. The projection matrix for both models were created by matching parameter moments at E_0 and frequency moments at the input frequency $f = 1GHz$.

The models were simulated at a set of parameter values in $[0.9E_0, 1.1E_0]$ and their outputs compared to the output of the full nonlinear system. Figure 5-1 compares the maximum percent error for each model (defined in (4.11)).

A similar comparison is made in Figure 5-2 using models of the Diode transmission line created with TPWL and Algorithm 2. In this figure the error considered is the norm of the error vector, $error = ||y(t) - \hat{y}(t)||$. These models were constructed by training at at $I_{d0} = 10^{-10}$ and $I_{d0} = 0.5I_{d0}, 1.3I_{d0}$ for Algorithm 2.

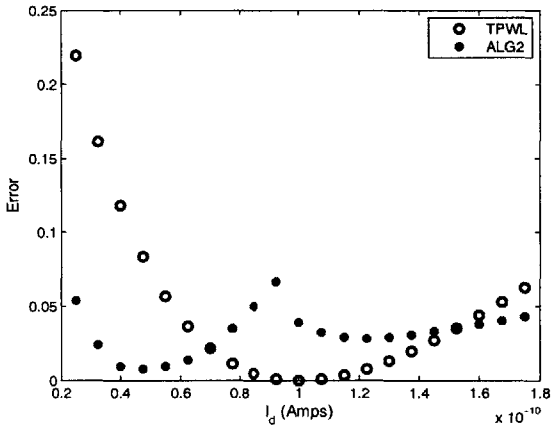

Figure 5-2: Norm of the error (defined in (4.5)) over time for models created with TPWL trained at $I_{d0} = 10^{-10}$ and with Algorithm 2 trained at $1.3I_{d0}, 0.5I_{d0}$. The system was reduced from original order $N = 100$ to reduced order $q = 50$.

Both Figures 5-2 and 5-1 show the best accuracy near the training parameter value for the model created by training at a single point. However both figures also show that the model created by training at multiple parameter space points has much better accuracy in the larger region around the training values. This is intuitive because TPWL and Algorithm 1 have all of their linear models created by training at the nominal parameter value, whereas Algorithms 1 and 3 have some linear models created from training at different parameter values. The meaning of this is that TPWL and Algorithm 2 do an excellent jo approximating (3.1) around the nominal parameter value, while Algorithms 1 and 3 do a good job approximating (3.1) over a range of parameter values.

To examine this trend one step further and see some real outputs, Figures 5-

3 and 5-4 consider pulse sharpening transmission line models created with TPWL trained at $\frac{1}{L} = p_0 = 10^{11}$ and Algorithm 2 trained at $p = p_0, 2p_0$. Figure 5-3 shows the error ($|y(t) - \hat{y}(t)|$) in the output of the two models simulated far from the nominal parameter value, $L = 1.8p_0$, while Figure 5-4 compares the error over time for the models simulated at at a parameter value near the nominal value, $p = 1.05p_0$. Both figures reinforce with the previous conjecture that TPWL is more accurate near the nominal value while Algorithm 2 is more accurate far from the nominal value.

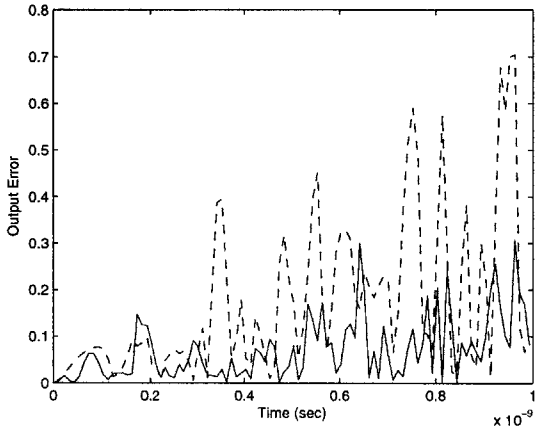

Figure 5-3: The error in the output of two models created with TPWL (dashed line) and Algorithm 2 (solid line) for the pulse narrowing transmission line parameterized in $p = \frac{1}{L}$ and simulated at $p = 1.8p_0$. The models were reduced from order $N = 100$ to $q = 50$.

5.2.2 Benefits of Parameterizing the Projection Matrix

The benefits of parameterizing the projection matrix via PMOR moment matching, as in section 3.1.3, can be examined by comparing models created with TPWL and Algorithm 2 with models created by Algorithm 1 and Algorithm 3.

Using the multivariable Taylor series expansion for creating V decouples the parameters. It creates vectors needed for the system response to capture changes in the parameters. This is especially important for systems with multiple parameters which may all be changing independently of one another. Figure 5-5 compares the

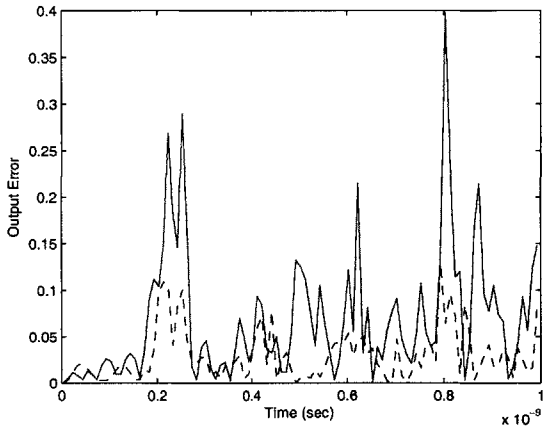

Figure 5-4: The error in the output of two models created with TPWL (dashed line) and Algorithm 2 (solid line) for the pulse narrowing transmission line parameterized in $p = \frac{1}{L}$ and simulated at $p = 1.05p_0$. The models was reduced from order $N = 100$ to $q = 50$.

total simulation error at different parameter values for models created with TPWL and Algorithm 1 for the diode transmission line parameterized in I_d.

For the circuit example parameterized in I_d, Figure 5-6 compares the error of models with differing reduced orders created by TPWL and Algorithm 1. All reduced order models were trained at $I_d = I_0$, and Algorithm 1 was expanded about $I_d = I_0$ for V. The figure shows that as the model order is increased, the error from models created with Algorithm 1 decreases much faster than the error from the models created with TPWL.

Again, the results match up with what is intuitively expected. Algorithms 1 and 3 match moments in the parameter space, creating a subspace which has the ability to capture changes in the parameter. This is essentially equivalent to taking a higher order approximation to $X(s,p)$ around p. On the other hand TPWL and Algorithm 2 do not match any moments around p, but instead match more moments in the expansion of $X(s,p)$ around s. This causes TPWL and Algorithm 2 to be very accurate at the nominal value p_0, but have very little ability to capture changes in the parameter.

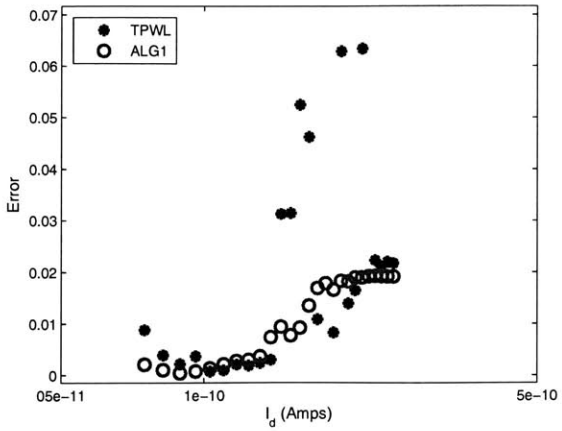

Figure 5-5: Two reduced order models of the diode transmission line parameterized in I_d and simulated at a range of parameter values using sinusoidal inputs. The models were created using TPWL and Algorithm 1 and were reduced from large order $N = 100$ to reduced order $q = 40$.

5.2.3 Benefits of using extra models for Krylov vectors

For a linear system, the first few Krylov vectors form a good basis for accurately approximating the system response. Similarly for the nonlinear system, the first few Krylov vectors from some set of linearized models span a low-dimensional subspace in which the state evolution can be well approximated. However, it is not known beforehand which linear models will provide vectors for the most important subspace. Thus, a possibly better approach would be to use Krylov vectors from the linear models at every time step along training trajectories. Recall from Section 3.1.3 that the trajectories are computed by solving a nonlinear system using an explicit time integration scheme, thus a linearized model is created at every timestep. Therefore, the additional cost of using more Krylov vectors is only the cost of performing system solves to obtain the vectors - there are no additional linearizations which need to be performed. The resulting projection matrix may be quite large, so an SVD must be taken to retain only the q most important singular vectors.

To examine the benefits of this method, two models were created using the TPWL Algorithm: one uses only Krylov vectors generated at linear models, while the other uses Krylov vectors generated at every point along the trajectory. Both PROMs used

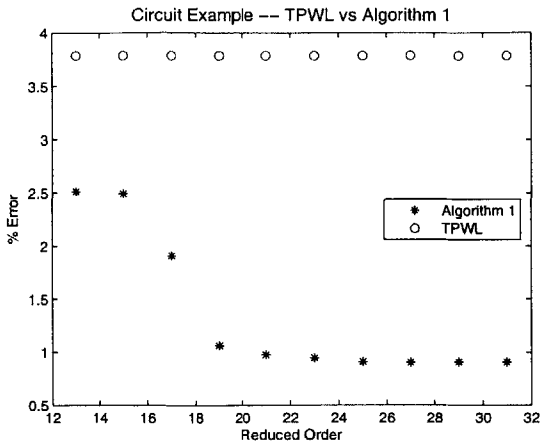

Figure 5-6: Comparison of reduced order models created with TPWL and Algorithm 1 for the diode transmission line parameterized in I_d. The percent error between the model output and real output are compared for different order models, created from a large model of order $N = 100$. Both models were trained at $I_d = I_{d0}$.

the same number of linear models and had the same reduced order q. Figure 5-7 compares the error of the output from these two models.

To see if this is always the case, another example is considered; this time, the diode transmission line parameterized in $1/R$. Figure 5-8 compares the error of the two models, and in this case there is little difference. For the most part the two models with $q = 6$ have the same output, but as p strays far from the nominal value, the model created from extra vectors beings to fair slightly better. The error from the model with extra vectors is never worse than the error from the model with fewer vectors.

It is not surprising that the model with more Krylov vectors to choose from is more accurate. Let V_1 be the projection matrix before reduction from SVD created by taking moments at every trajectory step, and let V_2 be the projection matrix before SVD created by taking moments only from the linear models. Clearly $V_1 \supset V_2$ since the linear models are created at points on the trajectory. Therefore SVD(V_1) will do no worse than SVD(V_2) since every vector in V_2 is also in V_1. In many cases SVD(V_1) will find a better subspace, resulting in a more accurate PROM, which is what happened in Figure 5-7.

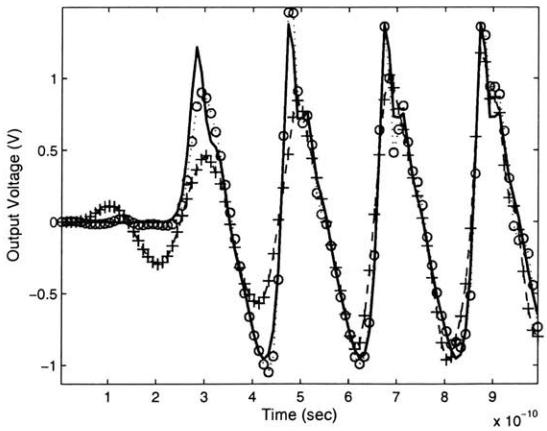

Figure 5-7: Two models of the pulse narrowing transmission line parameterized in $\frac{1}{L}$. The circles correspond to the model which was projected with V built from moments at every point of the trajectories, the crosses correspond to the model projected with V built using only moments from the $k = 181$ linear models. Both models were projected from large order N=200 down to reduced order $q = 50$.

It is important to note here that using the projection matrix created from moments taken along every step of the projection matrix can no longer be considered "moment matching". The vectors in V may not all be from the linear models used in the PROM, so it cannot be said for sure that the transfer function of those reduced models will match exactly any number of moments of the transfer function of some linearization of the original model, as was the case in Section 2.2.1.

5.2.4 Analysis of Linearizing in Parameters

The final factor considered in parameter space accuracy is the linearization of the nonlinear system in the parameters, seen in (3.4). The first parameter considered is the diode turn on voltage v_T. To simplify the equations, the parameter is defined as $\alpha = \frac{1}{v_T}$. The original system (4.1) is nonlinear in α, so (4.2) is linearized about some nominal value α_0 to obtain the state space model

$$\frac{dx}{dt} = Gx + D(x, \alpha_0) + \frac{\partial}{\partial \alpha}\left[D(x, \alpha)\right](\alpha - \alpha_0) + bu(t). \tag{5.1}$$

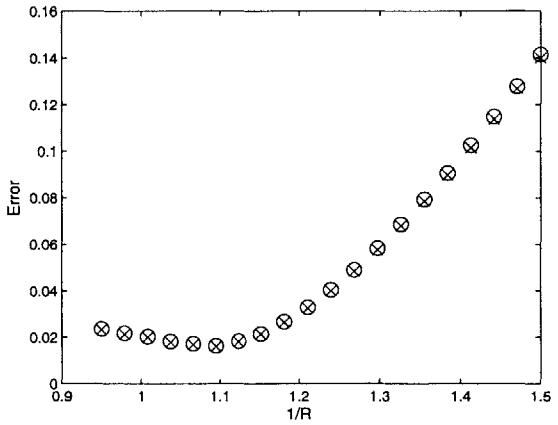

Figure 5-8: Two different PROMs of the diode transmission linear parameterized in $p = \frac{1}{R}$. Both models were projected from large order $N = 200$ down to reduced order $q = 6$. The circles correspond to the model using moments only from the $k = 166$ linear models, while the crosses correspond to the models created using moments from every point along the trajectory.

Note that the system is still nonlinear in the state. A model was created using sinusoids as training inputs and a nominal parameter value of $\alpha = 40$ for linearization and training. As mentioned previously, linearizing at α_0 introduces some initial error as the parameter varies away from α_0. An important question to ask is whether the limiting factor in parameter-space accuracy of the PROM is a result of projecting the system into a low-order subspace, or a result of this original linearization in the parameters. Figure 5-9 compares the error of the PROM with the error of the large piecewise-linear non-reduced system (3.7) at several different parameter values. The figure shows that in this case the error from the PROM is no worse than the error from the piecewise linear large model. This indicates the majority of the error in these simulations may be a result of the original linearization in the parameter.

5.3 Cost And Timing

Each algorithm presented in Section 3.2 follows a slightly different path to create theirs model, resulting in four different computational costs for the four algorithms.

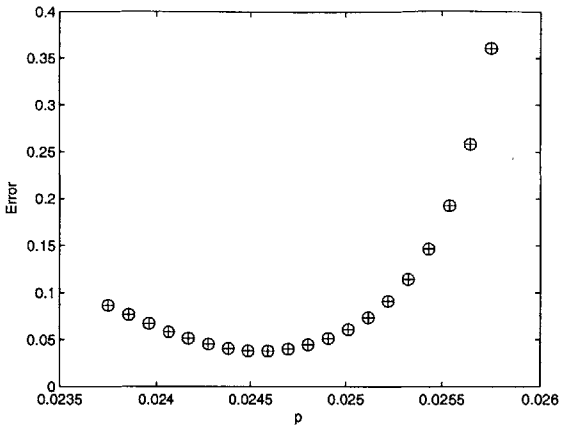

Figure 5-9: Comparison of an order $q = 20$ PROM for the diode transmission line parameterized in $p = \frac{1}{v_t}$ created using TPWL algorithm (plusses) with a large piecewise linear model with order $q = 200$ (circles). The two models were simulated over a range of parameter values using sinusoidal inputs. The error plotted is $|e(t)|$ as defined in (4.5).

This is mainly based on how the projection matrix V is constructed and how many training trajectories are created. Table 5.1 compares the cost of these two parts of the four algorithms.

The cost of constructing the projection matrix is measured in system solves because each Krylov vector requires one system solve to compute. Compared to TPWL, the new cost of Algorithm 1 results from creating additional vectors for V. The projection matrix will require $O(kp^m)$ vectors to match m moments with respect to each of the p parameters for k linear models.

The training trajectories are created by solving the large nonlinear system at each time step. If there are T total time steps for the trajectory and each nonlinear solve requires w Newton iterations, each trajectory will cost $O(wT)$ system solves on average. Note that if Krylov vectors are to be taken from every trajectory step instead of just the points where linear models are created, the cost of building V increases from $O(km)$ to $O(Tm)$ with T being the number of time steps in the trajectory. This is where the majority of the cost in Algorithm 2 lies. Training the system for p different parameters at r values requires r^p trajectories. The projection matrix now

	Building Projection Matrix V (in system solves)	**Training trajectories**
TPWL	$O(km)$	1
Alg 1	$O(kp^m)$	1
Alg 2	$O(kr^p m)$	r^p
Alg 3	$O(kr^p p^m)$	r^p

Table 5.1: Cost of training, measured in number of training trajectories requires, and cost of constructing projection matrix V, measured in system solves, for the four algorithms. In this table k represents the number of linear models in the PROM, m is the number of moments matched for each parameter, p is the total number of parameters, and r is the number of expansion points in the parameter space.

requires just $O(kr^p m)$ system solves.

The final algorithm, Algorithm 3, creates the most expensive models. As in Algorithm 2, the cost of training is $O(hTr^p)$, but the projection matrix now requires $O(kp^m r^p)$ system solves.

In the end, apart from accuracy, the most important model trait for the user is the time required to run a simulation with that model. The obvious things that may affect simulation time are the reduced order q and the number of linear models present k. It is difficult to precict accurately the effect of k and q because the simulation time depends on each piece in multiple ways. For example, when k and q are small there are fewer weights to compute and fewer nonlinear equations to solve, which should speed up simulation. But, if k or q is too small, the model may not have the proper pieces needed to approximate the nonlinear function for some input, or the state may want to leave the subspace chosen, both of which could cause trouble for the nonlinear solver resulting in many more Newton steps at each iteration, which would slow down the simulation. However the latter adverse effects are probably only significant for extremely small k and q.

To see the effects of k and q on simulation time, Table 5.2 examines the speedup of the PROM by comparing the simulation time of the reduced model to the simulation time of the full nonlinear model as these two factors are varied.

The data was collected by creating a PROM with some k and some q, simulating

k	$q = N/4$	$q = N/10$	$q = N/40$
42	9.0 X	17.1 X	20.7 X
95	7.5 X	9.82 X	8.7 X
96	6.52 X	11.0 X	11.52 X
252	3.5 X	4.6 X	5.1 X

Table 5.2: Examination of the ***speed-up*** factor for simulation time for a PROM compared to the full nonlinear system based on the reduction from large order N to reduced order q, and the number of linear models k in the PROM.

the PROM at ten different parameter values, and then simulating the full system at those same ten parameter values. If the time required for the full system simulation is T_{full} and the time required for the reduced system simulation is T_{red}, the speed up factor is defined as the ratio of these two times, T_{full}/T_{red}. Letting N be the order of the original system and q the order of the reduced system, the reduction factor is simply N/q.

From the table, it appears there is a relationship between the simulation speed and the order of the reduced model. It also appears they may be a relationship between simulation speed and k. It is certainly true that a larger k will increase the time required and cost of constructing the model.

Chapter 6

Conclusion

6.1 Conclusion

This work has shown that it is possible to create reduced order nonlinear models which accurately approximate large nonlinear systems over a practical range of geometrical parameter variations. Several approaches were tested on three examples: a diode transmission line, a MEM switch, and a pulse narrowing transmission line. Furthermore, based on the needs of the user, it is possible to construct models which are more or less accurate in different regions of the parameter space.

The PROMs were constructed as follows. Using the trajectory piecewise linear method, a collection of linear models was created by linearizing the nonlinear system at important regions of the state-space. Transfer function moments were then computed for each linear system to find a basis for a subspace in which the nonlinear system can be well approximated. These moments, which can be computed from an expansion of the transfer function in all of the parameters (Section 2.2.2) or from an expansion in only the Laplace variable (Section 2.2.1), can be used to assemble a projection matrix. This projection matrix was then used to project each linear system into the reduced space. A model of order q constructed in this manner is able to match a total of q moments of the large linearized system transfer functions. In the case where only Laplace variable moments are matched (TPWL and Algorithm 2 in Table 3.1), the resulting model will have extremely high accuracy very close to

the nominal parameter value, but it will sacrifice accuracy at other parameter values. On the other hand, when parameter moments are matched (Algorithms 1 and 3 in Table 3.1), the resulting model will be accurate over a larger range of parameter values, but it will sacrifice some accuracy at the nominal parameter value.

For a model with k linear models, there are k linearization points to be distributed across the state space. If the points are concentrated in regions where a single parameter value takes the trajectory (TPWL and Algorithm 1), then the resulting model will be more accurate for inputs very close to the nominal parameter value. If the k linearization points are spread out over regions of the space visited by trajectories at different parameter values (Algorithms 2 and 3), then the resulting model will be accurate for inputs over a wider range of parameter values, but may sacrifice some accuracy near the nominal parameter value.

It was also shown that using extra Krylov vectors for the projection matrix combined with an SVD (Section 5.2.3) to find the optimum subspace can produce more accurate PROMs in some cases. Since this method will always produce a PROM at least as accurate as the method considering fewer Krylov vectors, such an approach should always be considered if the additional cost is manageable.

Finally, Section 5.2.4 showed that in some cases the majority of the PROM error arises not from projecting the system into a low order space, but rather from the parameter linearization arising in the transition from (3.1) to (3.4). If this is the case, a more accurate approximation scheme of the parameter dependence should be employed (e.g. third order approximation instead of first order) to ensure optimum results from the reduction process.

6.2 Future Work

There are three main areas for improvement in this method: The piecewise linear approximation of a nonlinear function, the approximation of the function to obtain linear parameter dependence, and the choice of reduced order subspace for projection.

The first issue is related to picking linearization points. If there were a better

method to find important linear models, the piecewise linear approximation could better approximate the original nonlinear function, resulting in a more accurate PROM. This is also the most expensive part of the algorithm because it requires simulation of the full nonlinear system. An alternative method which creates fewer models and does not require solving the full nonlinear system would drastically reduce the cost of creating PROMs. This could be done by examining the nonlinearities in the original system to predict regions of the space where the model will behave very nonlinearly.

The second concern can be addressed by using higher order parameter dependence approximations or by using linearizations at multiple parameter values. This could possibly be achieved by building parameter dependence into the weighting functions to avoid the problem of different linear models at the same state.

The final area for improvement is related to the construction of the projection matrix. If there was a way to determine which models produce the most important Krylov vectors without linearizing and creating the vectors, it would be much cheaper computationally to construct the projection matrix.

Bibliography

[1] M. Rewienski and J. White. A trajectory piecewise-linear approach to model order reduction and fast simulation of nonlinear circuits and micromachined devices. *IEEE Trans. Computer-Aided Design*, 22(2):155–70, Feb 2003.

[2] D. Vasilyev, M. Rewienski, and J. White. A tbr-based trajectory piecewise-linear algorithm for generating accurate low-order models for nonlinear analog circuits and mems. In *Proc. of the ACM/IEEE Design Automation Conference*, pages 490–5, June 2003.

[3] E. Afshari and A. Hajimiri. A non-linear transmission line for pulse shaping in silicon. *IEEE Journal Solid-State Circuits*, 40(3):744–52, 2005.

[4] S. Pullela, N. Menezes, and L.T. Pileggi. Moment-sensitivity-based wire sizing for skew reduction in on-chip clock nets. *IEEE Trans. Computer-Aided Design*, 16(2):210–215, February 1997.

[5] H. Liu, A. Singhee, R. A. Rutenbar, and L. R. Carley. Remembrance of circuits past: macromodeling by data mining in large analog design spaces. In *Proc. of the ACM/IEEE Design Automation Conference*, pages 437–42, June 2002.

[6] K. C. Suo, A. Megretski, and L. Daniel. A quasi-convex optimization approach to parameterized model order reduction. In *Proc. of the IEEE/ACM Design Automation Conference*, CA, June 2005.

[7] Y Liu, Lawrence T. Pileggi, and Andrzej J. Strojwas. Model order-reduction of RCL interconnect including variational analysis. In *Proc. of the ACM/IEEE*

Design Automation Conference, pages 201–206, New Orleans, Louisiana, June 1999.

[8] C. Prud'homme, D. Rovas, K. Veroy, Y. Maday, A.T. Patera, and G. Turinici. Reliable real-time solution of parametrized partial differential equations: Reduced-basis output bounds methods. *Journal of Fluids Engineering*, 2002.

[9] D. S. Weile, E. Michielssen, Eric Grimme, and K. Gallivan. A method for generating rational interpolant reduced order models of two-parameter linear systems. *Applied Mathematics Letters*, 12:93–102, 1999.

[10] L. Daniel, C. S. Ong, S. C. Low, K. H. Lee, and J. K. White. A multiparameter moment matching model reduction approach for generating geometrically parameterized interconnect performance models. *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, 23(5):678–93, May 2004.

[11] L. Daniel and J. White. Automatic generation of geometrically parameterized reduced order models for integrated spiral rf-inductors. In *IEEE Intern. Workshop on Behavioral Modeling and Simulation*, San Jose, CA, September 2003.

[12] P. Li, F. Liu, S. Nassif, and L. Pileggi. Modeling interconnect variability using efficient parametric model order reduction. In *Design, Automation and Test Conference in Europe*, March 2005.

[13] P. Gunupudi and M. Nakhla. Multi-dimensional model reduction of vlsi interconnects. In *Proc. of the Custom Integrated Circuits Conference*, pages 499–502, Orlando, FL, 2000.

[14] P. Heydari and M. Pedram. Model reduction of variable-geometry interconnects using variational spectrally-weighted balanced truncation. In *Proc. of IEEE/ACM International Conference on Computer Aided-Design*, San Jose, CA, November 2001.

[15] J.R. Phillips. Variational interconnect analysis via PMTBR. In *Proc. of IEEE/ACM International Conference on Computer Aided-Design*, pages 872–9, November 2004.

[16] R. Gustavsen and A. Semlyen. Rational approximation of frequency domain responses by vector fitting. *IEEE Trans. Power Delivery*, 14:1052–1061, July 1993.

[17] Carlos P. Coelho, Joel R. Phillips, and L. Miguel Silveira. Robust rational function approximation algorithm for model generation. In 36^{th} *ACM/IEEE Design Automation Conference*, pages 207–212, New Orleans, Louisiana, June 1999.

[18] Carlos P. Coelho, Joel R. Phillips, and L. Miguel Silveira. A convex programming approach to positive real rational approximation. In *Proc. of IEEE/ACM International Conference on Computer Aided-Design*, pages 245–251, San Jose, CA, November 2001.

[19] M. Celik, A. Atalar, and M. A. Tan. Transient analysis of nonlinear circuits by combining asymptotic waveform evaluation with volterra series. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, 42(8):470–473, August 1995.

[20] J. Chen and S. M. Kang. An algorithm for automatic model-order reduction of nonlinear MEMS devices. In *Proceedings of ISCAS 2000*, pages 445–448, 2000.

[21] J.R. Phillips. Projection frameworks for model reduction of weakly nonlinear systems. In *Proc. of the ACM/IEEE Design Automation Conference*, pages 184–9, June 2000.

[22] J. R. Phillips. Automated extraction of nonlinear circuit macromodels. In *Proceedings of the Custom Integrated Circuit Conference*, pages 451–454, Orlando, FL, May 2000.

[23] J.R. Phillips. Projection-based approaches for model reduction of weakly nonlinear, time-varying systems. *IEEE Trans. Computer-Aided Design*, 22(2):171–87, 2003.

[24] P. Li and L. T. Pileggi. Norm: Compact model order reduction of weakly nonlinear systems. In *Proc. of the ACM/IEEE Design Automation Conference*, June 2003.

[25] Y. Chen. Model order reduction for nonlinear systems. M.S. Thesis, Massachusetts Institute of Technology, September 1999.

[26] M. Rewienski and J. K. White. A trajectory piecewise-linear approach to model order reduction and fast simulation of nonlinear circuits and micromachined devices. In *Proc. of IEEE/ACM International Conference on Computer Aided-Design*, pages 252–7, San Jose, CA, USA, November 2001.

[27] N. Dong and J. Roychowdhury. Piecewise polynomial nonlinear model reduction. In *Proc. of the ACM/IEEE Design Automation Conference*, June 2003.

[28] N. Dong and J. Roychowdhury. Automated extraction of broadly applicable nonlinear analog macromodels from spice-level descriptions. In *Proc. of the IEEE Custom Integrated Circuits Conference*, October 2004.

[29] A. Odabasioglu, M. Celik, and L. T. Pileggi. PRIMA: passive reduced-order interconnect macromodeling algorithm. *IEEE Trans. Computer-Aided Design*, 17(8):645–654, August 1998.

[30] S. Tiwary and R. Rutenbar. Scalable trajectory methods for on-demand analog macromodel extraction. In 42^{nd} *ACM/IEEE Design Automation Conference*, June 2005.

[31] N. Dong and J. Roychowdhury. Automated nonlinear macromodelling of output buffers for high-speed digital applications. In 42^{nd} *ACM/IEEE Design Automation Conference*, June 2005.