A Piecewise-Linear Moment-Matching Approach to Parameterized Model-Order Reduction for Highly Nonlinear Systems

Bradley N. Bond, *Student Member, IEEE*, and Luca Daniel, *Member, IEEE*

***Abstract*—This paper presents a parameterized reduction technique for highly nonlinear systems. In our approach, we first approximate the nonlinear system with a convex combination of parameterized linear models created by linearizing the nonlinear system at points along training trajectories. Each of these linear models is then projected using a moment-matching scheme into a low-order subspace, resulting in a parameterized reduced-order nonlinear system. Several options for selecting the linear models and constructing the projection matrix are presented and analyzed. In addition, we propose a training scheme which automatically selects parameter-space training points by approximating parameter sensitivities. Results and comparisons are presented for three examples which contain distributed strong nonlinearities: a diode transmission line, a microelectromechanical switch, and a pulse-narrowing nonlinear transmission line. In most cases, we are able to accurately capture the parameter dependence over the parameter ranges of $\pm 50\%$ from the nominal values and to achieve an average simulation speedup of about $10\times$.**

***Index Terms*—Model-order reduction (MOR), nonlinear systems, parameterized reduced-order models (PROMs).**

I. Introduction

THE AUTOMATIC extraction of parameterized macromodels for modern mixed-signal system-on-chips is an extremely challenging task due to the presence of several nonlinear analog circuits and microelectromechanical (MEM) components. The ability to generate 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.

Several parameterized model-order reduction (PMOR) techniques have been introduced in literature in the past few years. Some are based on statistical performance evaluation [1]–[4], and others are based on moment-matching techniques [5]–[11], on truncated-balance-realization (TBR) techniques [12], and on quasi-convex optimization techniques [13]. Very few, such as [4], also apply to the nonlinear systems.

Several non-PMOR approaches are available for nonlinear systems. For example, the reduction of weakly nonlinear systems has been shown using Volterra series and moment-matching techniques [14]–[19]. The reduction of strongly nonlinear systems has been shown using trajectory piecewise-linear (TPWL) methods combined with the moment-matching techniques [20]–[22], TPWL combined with the TBR techniques [23], and trajectory piecewise-polynomial combined with the moment-matching techniques [24], [25].

In [26], we proposed a PMOR technique for nonlinear systems by exploiting ideas from the nonparameterized TPWL method for nonlinear systems [22] and from a parameterized moment-matching technique for linear systems [9]. In such a method, the nonlinear system is approximated by a collection of parameterized linear models, which is obtained by linearizing the nonlinear system at important regions of the state space. Each linear model is then projected into a reduced space by application of a projection matrix. The procedure is completed by selecting a set of weighting functions which combine the parameterized reduced-order linear models. Here, we expand upon our previous work by generalizing the nonlinear PMOR (NLPMOR) algorithm and by examining how the parameter dependence of the reduced model is affected by the method with which linearization points and columns for the projection matrix are chosen. In addition, we propose an adaptive training scheme which selects parameter-space points for training by using available trajectory information to approximate the system sensitivity to the parameters. Requiring fewer training trajectories reduces the model generation cost and potentially eliminates redundant linear models. The proposed approach has been fully tested on the diode circuit and the MEM switch used in [26] and also on a new highly nonlinear distributed circuit which is used to propagate soliton waves.

The rest of this paper is organized as follows. Section II briefly reviews moment matching for linear systems, moment matching for parameterized linear systems, and a TPWL method for nonlinear nonparameterized systems. Our new approach is presented in Section III along with an algorithm for its implementation. Three parameterized nonlinear-system examples were chosen to test the proposed method and are described in detail in Section IV. Results from these examples along with the algorithm and parameter-space-accuracy analyses are presented in Section V.

Manuscript received September 12, 2006; revised March 18, 2007. This work was supported in part by Microelectronics Advanced Research Corporation (MARCO), by the National Science Foundation (NSF), and by the Defense Advanced Research Projects Agency (DARPA). This paper was recommended by Associate Editor L. M. Silveira.

The authors are with the Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139 USA (e-mail: bnbond@mit.edu).

Digital Object Identifier 10.1109/TCAD.2007.907258

II. Background

A. Moment-Matching MOR for Linear Systems

Consider a linear system

$$\frac{d\mathbf{x}}{dt} = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t), \quad \mathbf{y}(t) = \mathbf{C}^{\mathrm{T}}\mathbf{x}(t) \tag{1}$$

where $\mathbf{A} \in \mathbb{R}^{N\times N}$, $\mathbf{B} \in \mathbb{R}^{N\times r_{\mathrm{i}}}$, and $\mathbf{C} \in \mathbb{R}^{N\times r_{\mathrm{o}}}$. The state $\mathbf{x} \in \mathbb{R}^N$ has a very large order N, and $\mathbf{u}(t) \in \mathbb{R}^{r_{\mathrm{i}}}$ and $\mathbf{y}(t) \in \mathbb{R}^{r_{\mathrm{o}}}$ are vectors containing the r_{i} inputs and r_{o} outputs, respectively. One approach to reduce the order of such a system involves employing an orthonormal projection matrix $\mathbf{V} \in \mathbb{R}^{N\times q}$ such that $\mathbf{x} \approx \mathbf{V}\hat{\mathbf{x}}$, where $\hat{\mathbf{x}} \in \mathbb{R}^q$ is the reduced state with $q \ll N$, obtaining the ROM

$$\frac{d\hat{\mathbf{x}}}{dt} = \hat{\mathbf{A}}\hat{\mathbf{x}}(t) + \hat{\mathbf{B}}\mathbf{u}(t), \quad \hat{\mathbf{y}}(t) = \hat{\mathbf{C}}^{\mathrm{T}}\hat{\mathbf{x}}(t) \tag{2}$$

where $\hat{\mathbf{A}} \in \mathbb{R}^{q\times q} = \mathbf{V}^{\mathrm{T}}\mathbf{A}\mathbf{V}$, $\hat{\mathbf{B}} \in \mathbb{R}^{q\times r_{\mathrm{i}}} = \mathbf{V}^{\mathrm{T}}\mathbf{B}$, and $\hat{\mathbf{C}} \in \mathbb{R}^{q\times r_{\mathrm{o}}} = \mathbf{V}^{\mathrm{T}}\mathbf{C}$ [27]. For the remainder of this paper, we will use the convention that vectors are denoted by bold-font lowercase letters (e.g., $\mathbf{x}$), matrices are denoted by bold-font capital letters (e.g., $\mathbf{A}$), and scalars are denoted by plain-font symbols (e.g., t).

The projection matrix is carefully constructed to preserve the input/output relationship (e.g., transfer function) of the system. If the projection matrix is chosen such that

$$\{\tilde{\mathbf{B}}, \mathbf{M}\tilde{\mathbf{B}}, \mathbf{M}^2\tilde{\mathbf{B}}, \ldots, \mathbf{M}^{q-1}\tilde{\mathbf{B}}\} \subseteq \mathrm{range}(\mathbf{V}) \tag{3}$$

where $\mathrm{range}(\mathbf{V}) = \{\mathbf{x} \in \mathbb{R}^N | \mathbf{x} = \mathbf{V}\hat{\mathbf{x}}, \hat{\mathbf{x}} \in \mathbb{R}^q, \mathbf{V} \in \mathbb{R}^{N\times q}\}$, $\mathbf{M} = \mathbf{A}^{-1}$, and $\tilde{\mathbf{B}} = -\mathbf{A}^{-1}\mathbf{B}$, then the resulting reduced-system transfer function will match the first q moments of the Taylor series expansion in the Laplace variable s of the large-system transfer function [28]

$$\mathbf{x} = [\mathbf{I} - s\mathbf{M}]^{-1}\tilde{\mathbf{B}}\mathbf{u} = \sum_{n=0}^{\infty} s^n\mathbf{M}^n\tilde{\mathbf{B}}\mathbf{u}. \tag{4}$$

B. Moment-Matching PMOR for Linear Systems

Consider a linear system whose dynamical descriptor matrices in the Laplace domain are functions of the Laplace frequency variable s and of some other geometrical parameters $s_1, \ldots, s_\mu$

$$\mathbf{E}(s, s_1, \ldots, s_\mu)\mathbf{x} = \mathbf{B}\mathbf{u} \tag{5}$$

where $\mathbf{E} \in \mathbb{R}^{N\times N}$. By using a polynomial fitting technique and introducing additional parameters $\tilde{s}$, as shown in [8] and [9], one can approximate the parameterized system as

$$[\tilde{\mathbf{E}}_0 + \tilde{s}_1\tilde{\mathbf{E}}_1 + \cdots + \tilde{s}_P\tilde{\mathbf{E}}_P]\mathbf{x} = \mathbf{B}\mathbf{u} \tag{6}$$

where $\tilde{\mathbf{E}}_i \in \mathbb{R}^{N\times N}$.

System (6) can be rearranged and expanded in a Taylor series to obtain

$$\mathbf{x} = [\mathbf{I} - \tilde{s}_1\mathbf{M}_1 - \cdots - \tilde{s}_P\mathbf{M}_P]^{-1}\tilde{\mathbf{B}}\mathbf{u} \tag{7}$$

$$= \sum_n (\tilde{s}_1\mathbf{M}_1 + \cdots + \tilde{s}_P\mathbf{M}_P)^n\tilde{\mathbf{B}}\mathbf{u} \tag{8}$$

$$= \sum_n \sum_{k_1} \cdots \sum_{k_P} \left[\mathbf{F}^n_{k_1,\ldots k_P}(\mathbf{M}_1, \ldots, \mathbf{M}_P)\tilde{\mathbf{B}}\mathbf{u}\right] s_1^{k_1}, \ldots, s_P^{k_P} \tag{9}$$

where $\tilde{\mathbf{B}} = \tilde{\mathbf{E}}_0^{-1}\mathbf{B}$, $\mathbf{M}_i = -\tilde{\mathbf{E}}_0^{-1}\tilde{\mathbf{E}}_i$, and $\mathbf{F}_i \in \mathbb{R}^{N\times N}$. The formulas for $\mathbf{F}_i$ are long and convoluted. Here, we present only the formula for the simplified case $P = 2$ where the pattern of the vectors $\mathbf{F}_i$ is more perceptible. Detailed recursive formulas for the calculation of $\mathbf{F}_i$ with an arbitrary P can be found in [9]. If $P = 2$, system (6) becomes

$$[\tilde{\mathbf{E}}_0 + \tilde{s}_1\tilde{\mathbf{E}}_1 + \tilde{s}_2\tilde{\mathbf{E}}_2]x = \mathbf{B}\mathbf{u} \tag{10}$$

and (9) becomes

$$\sum_n \sum_k \left[\mathbf{F}^n_k(\mathbf{M}_1, \mathbf{M}_2)\tilde{\mathbf{B}}\mathbf{u}\right] s_1^{n-k}s_2^k \tag{11}$$

where $\tilde{\mathbf{B}} = \tilde{\mathbf{E}}_0^{-1}\mathbf{B}$, $\mathbf{M}_i = -\tilde{\mathbf{E}}_0^{-1}\tilde{\mathbf{E}}_i$, and $\mathbf{F}^n_k(\mathbf{M}_1, \mathbf{M}_2)$ is shown at the bottom of the page. This recursive formula generates vectors of the form

$$\tilde{\mathbf{B}}, \mathbf{M}_1\tilde{\mathbf{B}}, \mathbf{M}_2\tilde{\mathbf{B}}, \mathbf{M}_1^2\tilde{\mathbf{B}}, (\mathbf{M}_1\mathbf{M}_2 + \mathbf{M}_2\mathbf{M}_1)\tilde{\mathbf{B}}, \mathbf{M}_2^2\tilde{\mathbf{B}}, \ldots$$

If we now construct the projection matrix $\mathbf{V}$ such that

$$\left[\tilde{\mathbf{B}}, \mathbf{M}_1\tilde{\mathbf{B}}, \mathbf{M}_2\tilde{\mathbf{B}}, \mathbf{M}_1^2\tilde{\mathbf{B}}, (\mathbf{M}_1\mathbf{M}_2 + \mathbf{M}_2\mathbf{M}_1)\tilde{\mathbf{B}}, \ldots\right] \subseteq \mathrm{range}(\mathbf{V}) \tag{12}$$

for the $P = 2$ case, and

$$\{\mathbf{F}_0\tilde{\mathbf{B}}, \mathbf{F}_1\tilde{\mathbf{B}}, \ldots\} \subseteq \mathrm{range}(\mathbf{V}) \tag{13}$$

for arbitrary P, then the P-variable Taylor series expansion of the transfer function of the reduced-order system

$$[\hat{\mathbf{E}}_0 + \tilde{s}_1\hat{\mathbf{E}}_1 +, \ldots, \tilde{s}_P\hat{\mathbf{E}}_P]\hat{x} = \hat{\mathbf{B}}u \tag{14}$$

will match exactly the P-variable Taylor series transfer-function moments of the original system (6), where $\hat{\mathbf{E}}_i \in \mathbb{R}^{q\times q} = \mathbf{V}^{\mathrm{T}}\tilde{\mathbf{E}}_i\mathbf{V}$, and $\hat{\mathbf{B}} \in \mathbb{R}^{q\times r_i} = \mathbf{V}^{\mathrm{T}}\tilde{\mathbf{B}}$.

$$\mathbf{F}^n_k(\mathbf{M}_1, \mathbf{M}_2) = \begin{cases} 0, & \text{if } k \notin 0, 1, \ldots, n \\ I, & \text{if } m = 0 \\ \mathbf{M}_1\mathbf{F}^{n-1}_k(\mathbf{M}_1, \mathbf{M}_2) + \mathbf{M}_2\mathbf{F}^{n-1}_{k-1}(\mathbf{M}_1, \mathbf{M}_2), & \text{otherwise} \end{cases}$$

It is noted in [9] that for a large number of parameters P and a modest number of moments m matched for each parameter, this method may generate systems of substantial order $O(P^m)$.

C. TPWL for MOR of Nonlinear Systems

Consider a nonlinear system in the form

$$\frac{d\mathbf{x}}{dt} = \mathbf{f}\left(\mathbf{x}(t)\right) + \mathbf{B}\mathbf{u}(t), \quad \mathbf{y} = \mathbf{C}^{\mathrm{T}}\mathbf{x}(t) \tag{15}$$

where $\mathbf{f} : \mathbb{R}^N \mapsto \mathbb{R}^N$. The TPWL method uses local linear approximations to represent the nonlinear function $\mathbf{f}(\mathbf{x})$, resulting in a collection of linear models [22]. The nonlinear system can then be approximated with a weighted combination of the linear models

$$\frac{d\mathbf{x}}{dt} = \sum_{i=0}^{\kappa-1} w_i(\mathbf{x}, \mathbf{X}) \left[\mathbf{A}_i\mathbf{x} + \mathbf{k}_i\right] + \mathbf{B}\mathbf{u}(t) \tag{16}$$

where $w_i(\mathbf{x}, \mathbf{X})$ denotes some weighting functions which depend on the state $\mathbf{x}$ and the κ linearization points $\mathbf{X} \in \mathbb{R}^{N\times\kappa} = [\mathbf{x}_0, \mathbf{x}_2, \ldots, \mathbf{x}_{\kappa-1}]$, and $(\mathbf{A}_i, \mathbf{k}_i)$ represents the linearized model at state $\mathbf{x}_i$

$$\mathbf{A}_i = \frac{\partial \mathbf{f}(\mathbf{x})}{\partial \mathbf{x}}\Big|_{\mathbf{x}_i}$$
$$\mathbf{k}_i = \mathbf{f}(\mathbf{x}_i) - \frac{\partial \mathbf{f}(\mathbf{x})}{\partial \mathbf{x}}\Big|_{\mathbf{x}_i} \mathbf{x}_i.$$

Since it would be too expensive to uniformly cover the entire state space with linearizations, models are only created in important regions of the space where the vector field $\mathbf{f}(\mathbf{x})$ is very nonlinear and where the state is likely to evolve during simulation. Such important regions can be found for instance by simulating the nonlinear system with some "typical" training inputs and restricting the choice of linear models to these resulting trajectories. One major drawback of this method is that the accuracy of the reduced model can be highly dependent on the "richness" of the inputs chosen for the training.

A projection matrix $\mathbf{V}$ can now be created by assembling all the Krylov vectors (3) from each of the linearized models. The reduced system becomes

$$\frac{d\hat{\mathbf{x}}}{dt} = \sum_{i=0}^{\kappa-1} w_i(\hat{\mathbf{x}}, \hat{\mathbf{X}}) \left[\hat{\mathbf{A}}_i\hat{\mathbf{x}}(t) + \hat{\mathbf{k}}_i\right] + \hat{\mathbf{B}}\mathbf{u}(t) \tag{17}$$

where $\hat{\mathbf{A}}_i = \mathbf{V}^{\mathrm{T}}\mathbf{A}_i\mathbf{V}$, $\hat{\mathbf{k}}_i = \mathbf{V}^{\mathrm{T}}\mathbf{k}_i$, $\hat{\mathbf{B}} = \mathbf{V}^{\mathrm{T}}\mathbf{B}$, $\hat{\mathbf{x}} = \mathbf{V}^{\mathrm{T}}\mathbf{x}$, $\hat{\mathbf{X}} = [\mathbf{V}^{\mathrm{T}}\mathbf{x}_1, \ldots, \mathbf{V}^{\mathrm{T}}\mathbf{x}_\kappa]$, and κ is the number of linear models. The relative weights $w_i(\hat{\mathbf{x}}, \hat{\mathbf{X}})$ of each linear model vary dynamically as the state evolves. One possible weighting scheme is

$$w_i(\hat{\mathbf{x}}, \hat{\mathbf{X}}) = \frac{\exp\left[\frac{-\beta d_i^2}{m^2}\right]}{\sum_j \exp\left[\frac{-\beta d_j^2}{m^2}\right]}$$

where β is some constant (typically, we used $\beta = 25$), $d_i = \|\hat{\mathbf{x}} - \hat{\mathbf{x}}_i\|$, and $m = \min_i(d_i)$. Other possible weighting schemes have been proposed in [29] and [30].

III. PMOR for Nonlinear Systems

A. PROM Description Derivation

Let us consider a system possessing nonlinear dependence on both the state $\mathbf{x}(t)$ and some parameters p_i

$$\frac{d\mathbf{x}}{dt} = \mathbf{f}\left(\mathbf{x}(t), p_1, p_2, \ldots, p_\mu\right) + \mathbf{B}(p_1, \ldots, p_\mu)\mathbf{u}(t)$$
$$\mathbf{y} = \mathbf{C}^{\mathrm{T}}\mathbf{x} \tag{18}$$

where $\mathbf{x} \in \mathbb{R}^N$, $\mathbf{f} : \mathbb{R}^N \mapsto \mathbb{R}^N$, $\mathbf{B} \in \mathbb{R}^{N\times r_i}$, and $\mathbf{C} \in \mathbb{R}^{N\times r_o}$. By using, for instance, a polynomial fitting scheme or a Taylor series approximation in the parameters, we can extract the parameter dependence from the nonlinear functions by writing

$$\mathbf{f}(\mathbf{x}, p_1, \ldots, p_\mu) \approx \sum_{j=0}^{P-1} g_j(p_1, \ldots, p_\mu)\mathbf{f}_j(\mathbf{x})$$
$$\mathbf{B}(p_1, \ldots, p_\mu)\mathbf{u}(t) \approx \sum_{j=0}^{P-1} g_j(p_1, \ldots, p_\mu)\mathbf{B}_j\mathbf{u}(t) \tag{19}$$

where $g_j(p_1, \ldots, p_\mu)$ denotes the scalar functions of the parameters, $\mathbf{f}_j(\mathbf{x})$ denotes the vector-valued functions of the state, and $\mathbf{B}_j$ denotes the constant input matrices. For example, a first-order Taylor series expansion on $\mathbf{f}(\mathbf{x}, p)$ would yield

$$\mathbf{f}_j(\mathbf{x}) = \frac{\partial \mathbf{f}(\mathbf{x}, p)}{\partial p_j}\Big|_{p_0} \tag{20}$$
$$\mathbf{f}_0(\mathbf{x}) = \mathbf{f}(\mathbf{x}, p_0) - \sum_j p_{j0}\frac{\partial \mathbf{f}(\mathbf{x}, p)}{\partial p_j}\Big|_{p_0}. \tag{21}$$

By introducing a new set of parameters $\tilde{p}_j = g_j(p_1, \ldots, p_\mu)$, it is finally possible to make the system affine in the new parameters

$$\frac{d\mathbf{x}}{dt} = \sum_{j=0}^{P-1} \tilde{p}_j \left[\mathbf{f}_j(\mathbf{x}) + \mathbf{B}_j\mathbf{u}(t)\right] \tag{22}$$

while retaining the nonlinear dependence on the original parameters. In order to keep the equations concise, we choose $\mathbf{f}_0(\mathbf{x})$ and $\mathbf{B}_0$ to be the terms with no parameter dependence, and thus, we define $\tilde{p}_0 = 1$.

It is desirable to write the system as (22) because such form permits approximating each nonlinear function $\mathbf{f}_j(\mathbf{x})$ as an affine function without affecting the parameter dependence of the system

$$\tilde{p}_j\mathbf{f}_j(\mathbf{x}) \approx \tilde{p}_j \left[\mathbf{f}_j(\mathbf{x}_i) + \frac{\partial \mathbf{f}_j}{\partial \mathbf{x}}(\mathbf{x} - \mathbf{x}_i)\right]$$
$$= \tilde{p}_j[\mathbf{A}_{ij}\mathbf{x} + \mathbf{k}_{ij}]$$
$$\mathbf{A}_{ij} = \frac{\partial \mathbf{f}_j(\mathbf{x})}{\partial \mathbf{x}}\Big|_{\mathbf{x}_i}$$
$$\mathbf{k}_{ij} = \mathbf{f}_j(\mathbf{x}_i) - \frac{\partial \mathbf{f}_j(\mathbf{x})}{\partial \mathbf{x}}\Big|_{\mathbf{x}_i}\mathbf{x}_i$$

where $\mathbf{A}_{ij} \in \mathbb{R}^{N\times N}$, and $\mathbf{k}_{ij} \in \mathbb{R}^N$.

This allows, as in [22], an approximation of the nonlinear functions $\mathbf{f}_j(\mathbf{x})$ as a collection of local linearizations around different points $\mathbf{x}_i$ in the state space

$$\frac{d\mathbf{x}}{dt} = \sum_{i=0}^{\kappa-1}\sum_{j=0}^{P-1} w_i(\mathbf{x},\mathbf{X})\tilde{p_j}[\mathbf{A}_{ij}\mathbf{x} + \mathbf{k}_{ij} + \mathbf{B}_j\mathbf{u}] \tag{23}$$

where $w_i(\mathbf{x},\mathbf{X})$ denotes the weighting functions which vary dynamically with the state.

Now that the system matrices $\mathbf{A}_{ij}$ have no implicit parameter dependence, standard projection techniques can be applied to each of the linear systems in (23). For example, by using a projection matrix $\mathbf{V} \in \mathbb{R}^{N\times q}$, the system becomes

$$\begin{aligned}\frac{d\hat{\mathbf{x}}}{dt} &= \sum_{i=0}^{\kappa-1}\sum_{j=0}^{P-1} w_i(\hat{\mathbf{x}},\hat{\mathbf{X}})\tilde{p_j}[\hat{\mathbf{A}}_{ij}\hat{\mathbf{x}} + \hat{\mathbf{k}}_{ij} + \hat{\mathbf{B}}_j\mathbf{u}]\\ \mathbf{y} &= \hat{\mathbf{C}}^{\mathrm{T}}\hat{\mathbf{x}}\end{aligned} \tag{24}$$

where $\hat{\mathbf{x}} \in \mathbb{R}^q$, $\hat{\mathbf{A}}_{ij} \in \mathbb{R}^{q\times q} = \mathbf{V}^{\mathrm{T}}\mathbf{A}_{ij}\mathbf{V}$, $\hat{\mathbf{k}}_{ij} \in \mathbb{R}^q = \mathbf{V}^{\mathrm{T}}\mathbf{k}_{ij}$, $\hat{\mathbf{B}}_j \in \mathbb{R}^{q\times r_i} = \mathbf{V}^{\mathrm{T}}\mathbf{B}_j$, $\hat{\mathbf{C}} \in \mathbb{R}^{q\times r_o} = \mathbf{V}^{\mathrm{T}}\mathbf{C}$, $\mathbf{x} = \mathbf{V}\hat{\mathbf{x}}$, and $\hat{\mathbf{X}} \in \mathbb{R}^{q\times\kappa} = [\mathbf{V}^{\mathrm{T}}\mathbf{x}_0,\ldots,\mathbf{V}^{\mathrm{T}}\mathbf{x}_{\kappa-1}]$, resulting in a ROM which possesses a parameter dependence similar to that of the original model.

In order to complete the procedure, two algorithms need to be specified: how to choose the linearization points $\mathbf{x}_i$ and how to construct a projection matrix $\mathbf{V}$. These two methods will be discussed in detail in the following sections, and then, they will be combined to create the proposed NLPMOR algorithm.

B. Selecting Linearization Points

In the standard TPWL [20]–[25], linearization points are chosen along the state trajectories generated by typical training inputs. By using a similar idea, additional trajectories can be created by training with system (22) at a set of points in the parameter space $\{\tilde{p_j}\}$. This additional training produces linear models in new state-space regions where variations in the parameter are likely to drive the state. As with the training inputs, if we know a range of practical parameter values over which the system will be evaluated, we can restrict the parameter training points to that set. Additionally, if we have an information about the sensitivity of the system to each parameter, training should be performed in regions where the system is most sensitive to the parameter. Section III-D presents a method for approximating these sensitivities using this information to select the training points.

Computing the exact training trajectories requires a simulation of the full nonlinear system, which may be prohibitively expensive. Alternatively, one could use "approximate training trajectories." In this case, rather than simulating the full nonlinear system, we simulate a linearized model. It is assumed that this linearized model is accurate as long as the current simulated state stays in some neighborhood of the linearization state. Once the current simulated state leaves such a neighborhood, a new linearized model is created at the current state, and the procedure continues on in this manner.

The additional trajectories created by parameter-space training increase the cost of constructing the model but do not significantly affect the cost of simulating the ROM. Since the weighting functions in (24) 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 the simulation time [29]. Thus, by holding the order of the reduced system fixed and adding additional models from the new trajectories, the interpolation of the nonlinearity $\mathbf{f}_j(\mathbf{x})$ in (22) can be improved without increasing the simulation time.

C. Constructing the Projection Matrix

As in PMOR for linear systems [5], [8], [9], the columns of the projection matrix $\mathbf{V}$ can be chosen to span the subspace generated by the vectors from a multivariable Taylor series expansion about each parameter $\tilde{p}_j$ in (23). This is similar to the scheme used in Section II-B except that, in this case, the model is nonlinear. Therefore, the projection vectors are constructed using the vectors produced by a multivariable Taylor series expansion (with respect to the frequency and all of the parameters) of the transfer functions of each of the κ-linearized models created during the training. Constructing $\mathbf{V}$ in this manner ensures that the PROM will match the moments of the transfer functions of each of the linearized systems with respect to both the frequency and parameter values.

It is important to note here that it would be possible to generate parameterized projection vectors using other projection-based PMOR methods (for example, [6], [7], [10]–[12]). However, moment matching is suitable for this method because it is relatively cheap to compute a few moments from each linearization while it is being generated, and the parameterized moments allow us to more carefully fit the transfer functions around the frequencies and the parameter values at which the training trajectories were created.

The training procedure produces κ linear models that capture the nonlinear effects of the original system. In addition to creating Krylov vectors from these κ models, it may be beneficial to also create the Krylov vectors at additional points along the training trajectories. This does not significantly increase the computational cost because when solving the nonlinear system with an implicit time integration scheme (e.g., Newton's method with the backward Euler method), we produce linearizations at every time step; hence, the additional cost is merely a few system solves per additional Krylov vector.

One additional difference between the linear case in Section II-B and the nonlinear case is the constant vector $\mathbf{k}$ in (23)—an artifact of the state linearizations. This term can be treated as a second input vector $\mathbf{b}_2$ with a constant input $u_2(t) = 1$. Thus, (23) becomes

$$\frac{d\mathbf{x}}{dt} = \sum_{i=0}^{\kappa-1}\sum_{j=0}^{P-1} w_i(\mathbf{x},\mathbf{X})\tilde{p_j}\left[\mathbf{A}_{ij}\mathbf{x}(t) + \mathbf{b}_{2ij}u_2(t) + \mathbf{B}_j\mathbf{u}(t)\right]. \tag{25}$$

To account for this term, several Krylov vectors should also be generated as in Section II-B for each linear model with $\mathbf{k}$ in place of $\mathbf{B}$.

The matching of moments about multiple expansion points for every linear model may quickly increase the number of columns in the projection matrix. As $\mathbf{V}$ becomes large, simulation of the ROM will become costly. One way to keep the size of the reduced system small is to perform a singular value decomposition (SVD) on the projection matrix [12], [31], [32]. The SVD is relatively inexpensive because the projection matrix is very tall and also relatively "skinny." After SVD, only vectors corresponding to the largest q singular values are selected as columns for the new projection matrix $\mathbf{V}$, resulting in a reduced system of small order q.

D. Selecting Parameter-Space Training Points

One possible method of selecting the parameter values for training in the parameter space is to predict whether a change in parameter value will cause the state to visit new regions of the state space, which are not supported by the current projection operation subspace. Let us define $\mathbf{x}(t, \tilde{\mathbf{p}}_\mathrm{a}, \omega_\mathrm{a}) \in \mathbb{R}^N$ for $t \in [0, T]$ as a trajectory which solves (22) at $\tilde{\mathbf{p}}_\mathrm{a} = [\tilde{p}_{0\mathrm{a}}, \tilde{p}_{1\mathrm{a}}, \ldots, \tilde{p}_{P-1\mathrm{a}}]^\mathrm{T}$ driven by a sinusoidal input at frequency ω_a, and $\mathbb{V}$ as the subspace spanned by the columns of the projection matrix $\mathbf{V}$, which was constructed such that $\mathbf{x}(t, \tilde{\mathbf{p}}_\mathrm{a}, \omega_\mathrm{a}) \in \mathbb{V}$. If it can be shown that $\mathbf{x}(t, \tilde{\mathbf{p}}_\mathrm{b}, \omega_\mathrm{a}) \in \mathbb{V}$ for some $\tilde{\mathbf{p}}_\mathrm{b} = \tilde{\mathbf{p}}_\mathrm{a} + \Delta\tilde{\mathbf{p}}$ without computing $\mathbf{x}(t, \tilde{\mathbf{p}}_\mathrm{b}, \omega_\mathrm{a})$, then there is no need to train at $\tilde{\mathbf{p}}_\mathrm{b}$ to generate more projection vectors.

The solution $\mathbf{x}(t, \tilde{\mathbf{p}}_\mathrm{b}, \omega_\mathrm{a})$ can be approximated with a first-order Taylor series expansion in $\tilde{\mathbf{p}}$ as

$$\mathbf{x}(t, \tilde{\mathbf{p}}_\mathrm{b}, \omega_\mathrm{a}) \approx \mathbf{x}(t, \tilde{\mathbf{p}}_\mathrm{a}, \omega_\mathrm{a}) + \frac{\partial \mathbf{x}}{\partial \tilde{\mathbf{p}}}\Big|_{\tilde{\mathbf{p}}_\mathrm{a}} \Delta\tilde{\mathbf{p}} \tag{26}$$

where $\Delta\tilde{\mathbf{p}} = [\Delta\tilde{p}_0, \ldots, \Delta\tilde{p}_{P-1}]^\mathrm{T} \in \mathbb{R}^P$, and $(\partial\mathbf{x}/\partial\tilde{\mathbf{p}}) \in \mathbb{R}^{N\times P}$. If $(\partial\mathbf{x}/\partial\tilde{\mathbf{p}}) \in \mathbb{V}$, then $\mathbf{x}(t, \tilde{\mathbf{p}}_\mathrm{b}, \omega_\mathrm{a}) \in \mathbb{V}$ because $\mathbb{V}$ is a linear subspace; therefore, linear combinations of elements in $\mathbb{V}$ are also in $\mathbb{V}$.

To compute $\partial\mathbf{x}/\partial\tilde{\mathbf{p}}$, let us first denote $\mathbf{x}_k = \mathbf{x}(t_k, \tilde{\mathbf{p}}_\mathrm{a}, \omega_\mathrm{a})$ for $1 \le k \le \tau$ as a sample of trajectory $\mathbf{x}(t, \tilde{\mathbf{p}}_\mathrm{a}, \omega_\mathrm{a})$ at t_k and, then, define $\bar{\mathbf{x}} \in \mathbb{R}^{N\tau} = [\mathbf{x}_1^\mathrm{T}, \ldots, \mathbf{x}_\tau^\mathrm{T}]^\mathrm{T}$ as a stack of the τ trajectory samples into one long vector. Since $\mathbf{x}$ solves (22), this new variable $\bar{\mathbf{x}}$ approximately solves the system

$$\bar{\mathbf{D}}\bar{\mathbf{x}} = \sum_{j=0}^{P-1} \tilde{p}_j \left[\bar{\mathbf{f}}_j(\bar{\mathbf{x}}) + \bar{\mathbf{B}}_j\right] \tag{27}$$

where $\bar{\mathbf{D}} \in \mathbb{R}^{N\tau\times N\tau}$ is a finite difference time-derivative operator, $\bar{\mathbf{f}}_j : \mathbb{R}^{N\tau} \mapsto \mathbb{R}^{N\tau} = [\mathbf{f}_j^\mathrm{T}(\mathbf{x}_1), \ldots, \mathbf{f}_j^\mathrm{T}(\mathbf{x}_\tau)]^\mathrm{T}$, and $\bar{\mathbf{B}}_j \in \mathbb{R}^{N\tau} = [(\mathbf{B}\mathbf{u}(t_1))^\mathrm{T}, \ldots, (\mathbf{B}\mathbf{u}(t_\tau))^\mathrm{T}]^\mathrm{T}$. Differentiating this system with respect to each of the parameters yields

$$\bar{\mathbf{D}}\frac{\partial\bar{\mathbf{x}}}{\partial\tilde{\mathbf{p}}} = \tilde{\bar{\mathbf{f}}}(\bar{\mathbf{x}}) + \sum_{j=0}^{P-1} \tilde{p}_j \frac{\partial\bar{\mathbf{f}}_j}{\partial\bar{\mathbf{x}}}\frac{\partial\bar{\mathbf{x}}}{\partial\tilde{\mathbf{p}}} \tag{28}$$

where $(\partial\bar{\mathbf{f}}/\partial\bar{\mathbf{x}}) \in \mathbb{R}^{N\tau\times N\tau}$, $(\partial\bar{\mathbf{x}}/\partial\tilde{\mathbf{p}}) \in \mathbb{R}^{N\tau\times P}$, and $\tilde{\bar{\mathbf{f}}} \in \mathbb{R}^{N\tau\times P}$ is

$$\tilde{\bar{\mathbf{f}}}(\bar{\mathbf{x}}) = \left[\left(\bar{\mathbf{f}}_0(\bar{\mathbf{x}}) + \bar{\mathbf{B}}_0\right), \ldots, \left(\bar{\mathbf{f}}_{P-1}(\bar{\mathbf{x}}) + \bar{\mathbf{B}}_{P-1}\right)\right]. \tag{29}$$

This can be rearranged into the linear system

$$\left[\bar{\mathbf{D}} - \sum_{j=0}^{P-1} \tilde{p}_j \frac{\partial\bar{\mathbf{f}}_j}{\partial\bar{\mathbf{x}}}\right] \frac{\partial\bar{\mathbf{x}}}{\partial\tilde{\mathbf{p}}} = \tilde{\bar{\mathbf{f}}}(\bar{\mathbf{x}}) \tag{30}$$

whose solution is a narrow matrix $\partial\bar{\mathbf{x}}/\partial\tilde{\mathbf{p}}$ such that

$$\frac{\partial\bar{\mathbf{x}}}{\partial\tilde{\mathbf{p}}} = \begin{bmatrix} \frac{\partial\mathbf{x}}{\partial\tilde{p}_0}\Big|_{t=t_1} & \cdots & \frac{\partial\mathbf{x}}{\partial\tilde{p}_{P-1}}\Big|_{t=t_1} \\ \vdots & \ddots & \vdots \\ \frac{\partial\mathbf{x}}{\partial\tilde{p}_0}\Big|_{t=t_\tau} & \cdots & \frac{\partial\mathbf{x}}{\partial\tilde{p}_{P-1}}\Big|_{t=t_\tau} \end{bmatrix}. \tag{31}$$

System (30) is large; however, $\partial\bar{\mathbf{f}}/\partial\bar{\mathbf{x}}$ is very sparse, and $\tilde{\bar{\mathbf{f}}}$ is narrow and sparse. Assembling the system requires no extra work because both the Jacobians and the function evaluations in (30) were already computed at every time step during the training process.

If $\partial\mathbf{x}/\partial\tilde{\mathbf{p}}$ is well approximated by vectors in $\mathbb{V}$, then its largest singular vectors, defined as $\tilde{\partial}\mathbf{x}/\partial\mathbf{p}$, are orthogonal to the null space of $\mathbf{V}^\mathrm{T}$, defined as $\mathcal{N}(\mathbf{V}^\mathrm{T})$, i.e.,

$$\left\| \left(\frac{\tilde{\partial}\mathbf{x}}{\partial\mathbf{p}}\right)^\mathrm{T} \mathcal{N}(\mathbf{V}^\mathrm{T}) \right\| \le \epsilon \tag{32}$$

where ϵ is a small tolerance.

Note that even if solution $\tilde{\partial}\mathbf{x}/\partial\mathbf{p}$ is in $\mathbb{V}$, it may still be beneficial to add new linearization points from the trajectory $\mathbf{x}(t, \tilde{\mathbf{p}}_\mathrm{b}, \omega_\mathrm{a})$. If each linearized model is assumed to be accurate in some δ-ball around its linearization point, then no models are needed if

$$\|\mathbf{x}(t, \tilde{\mathbf{p}}_\mathrm{a}, \omega_\mathrm{a}) - \mathbf{x}(t, \tilde{\mathbf{p}}_\mathrm{b}, \omega_\mathrm{a})\| < \delta \tag{33}$$

for all t's. From (26), this is equivalent to

$$\left\| \frac{\partial\mathbf{x}(t)}{\partial\mathbf{p}} \right\| \le \frac{\delta}{\|\Delta p\|} \tag{34}$$

for all values of t.

Thus, one could perform the aforementioned checks while the trajectory at $\tilde{\mathbf{p}}_\mathrm{a}$ is being created, and one would know at the end of the trajectory whether it is necessary to train at a nearby parameter-space point $\tilde{\mathbf{p}}_\mathrm{b}$.

E. Proposed NLPMOR Algorithm

An algorithm for NLPMOR is constructed by defining both a linearization scheme (i.e., a method to choose linearization points for linear models) and a projection scheme (i.e., a method to construct the projection matrix $\mathbf{V}$). By combining the parameterization options in Sections III-B and C, we obtain four different schemes for training, presented in Table I, and four different schemes for constructing $\mathbf{V}$, presented in Table II.

TABLE I
FOUR OPTIONS IN SELECTING LINEARIZATION POINTS FROM THE TRAINING TRAJECTORIES

	Training at Single Point in Parameter Space	Training at Multiple Points in Parameter Space
Exact Trajectories	T_{es}	T_{em}
Approximate Trajectories	T_{as}	T_{am}

TABLE II
FOUR OPTIONS IN CONSTRUCTING THE PROJECTION MATRIX

	Krylov Vectors From All States	Krylov Vectors Only From Linearized Systems
MOR V Moment Matching in V	V_{ma}	V_{ml}
PMOR V Moment Matching in V	V_{pa}	V_{pl}

A generic NLPMOR algorithm which incorporates each of these options is presented in Algorithm 1.

In order to select a linearization scheme, two decisions need to be made: whether to exactly compute the trajectories or to merely approximate the trajectories and whether to train in the parameter space. If exact trajectories are used, the nonlinear system and the current linear model are solved to obtain $\mathbf{x}_t$ and $\mathbf{x}_i$, which are the states of the nonlinear system and linearized model, respectively, at time t. By defining $\Delta = \|\mathbf{x}_t - \mathbf{x}_i\|$ as the distance between the solution of the nonlinear system and the solution of the linearized system, a new model is created whenever $\Delta > \delta$, where δ is some preset tolerance. A linearization of the original nonlinear system consists of a pair $\{\mathbf{A}_{tj}, \mathbf{k}_{tj}\}$ as in (23). If approximate training trajectories are used, rather than comparing the linear-system solution $\mathbf{x}_i$ to the nonlinear-system solution, we compare $\mathbf{x}_i$ to the previous linearization points $\mathbf{x}_{L_j}$. By setting $\Delta = \min_j \|\mathbf{x}_i - \mathbf{x}_{L_j}\|$, linearizations are created when the state $\mathbf{x}_i$ strays too far from the closest of all the precalculated linearization points x_{L_j}, i.e., $\Delta > \delta$.

In order to select a projection scheme, we need to decide which linear models the Krylov vectors are generated from and which Taylor expansion is used to compute the vectors. When the system is trained with exact trajectories, the linear models are available at every time step, and it is cheap to create several Krylov vectors at each step (achieved in Algorithm 1 by setting KrAll $= 1$). If approximate trajectories are used for training, or if the cost of creating Krylov vectors at every time step is prohibitive, then the Krylov vectors are computed only from the linear models (obtained in Algorithm 1 by setting KrAll $= 0$).

Finally, the Krylov vectors could be created either with a single variable Taylor series expansion about the Laplace variable s, which is referred to in Algorithm 1 as MORV, or with a multivariable Taylor series expansion about s and all of the parameters, which is referred to in Algorithm 1 as PMORV.

Algorithm 1 NLPMOR

for each Training Input Signal **do**
  for each Training Point in the Parameter Space **do**
    Linearize nonlinear system at initial state x_{L_0}
    while $t < t_{\text{final}}$ **do**
      Simulate linearized model to compute its next state $\mathbf{x}_i$
      Set KrLin $= 0$
      if Exact Training Trajectories **then**
        Simulate nonlinear system to compute its next state $\mathbf{x}_t$
        Compute $\Delta = \|\mathbf{x}_t - \mathbf{x}_i\|$
        Set $\mathbf{x}_n = \mathbf{x}_t$
      else if Approximate Training Trajectories **then**
        Compute $\Delta = \min_j \|\mathbf{x}_{L_j} - \mathbf{x}_i\|$
        Set $\mathbf{x}_n = \mathbf{x}_i$
      end if
      if $\Delta > \delta$ **then**
        Linearize nonlinear system at current state $\mathbf{x}_n$
        $j \leftarrow j + 1$
        $x_{L_j} = x_n$
        KrLin $= 1$
      end if
      if (KrAll‖KrLin) **then**
        if MORV **then**
          Use (3) to compute $\mathbf{V}_{\text{new}}$
        else if PMORV **then**
          Use (13) to compute $\mathbf{V}_{\text{new}}$
        end if
        $\tilde{\mathbf{V}} = [\tilde{\mathbf{V}}\ \mathbf{V}_{\text{new}}]$
      end if
    end while
  end for
end for
Construct a new projection matrix $\mathbf{V}$ using only the dominant singular vectors of $\tilde{\mathbf{V}}$
Project systems using $\mathbf{V}$
Select weighting functions $w(\mathbf{x}, \mathbf{X})$

The notation in Tables I and II will be used in this paper to identify different kinds of model reduction algorithms. For instance, when we write a $T_{\text{em}}V_{\text{pl}}$ PROM, we mean that the reduced model is created by training with exact trajectories at multiple points in the parameter space and is reduced with a PMOR projection matrix with vectors taken only from the linear models created. As another example of our notation, when we compare $T_{\text{xx}}V_{\text{mx}}$ and $T_{\text{xx}}V_{\text{px}}$ models in Section V, we mean that we intend to examine only the effects of the MOR moment matching versus the PMOR moment matching.

F. Algorithm Costs

Since computing each Krylov vector requires one system solve, the cost of constructing the projection matrix can be measured in a number of system solves. Such costs are summarized in Table III. For a projection matrix created by generating the m MORV Krylov vectors from the κ linear models, the cost

TABLE III
COSTS OF CONSTRUCTING THE PROJECTION MATRIX USING THE FOUR AVAILABLE OPTIONS, MEASURED IN SYSTEM SOLVES PER TRAJECTORY

	Krylov Vectors From All States	Krylov Vectors Only From Linearized Systems
MOR V Moment Matching in $\mathbf{V}$	$O(Tm)$	$O(\kappa m)$
PMOR V Moment Matching in $\mathbf{V}$	$O(TP^m)$	$O(\kappa P^m)$

TABLE IV
COSTS OF TRAINING THE SYSTEM USING THE FOUR AVAILABLE OPTIONS, MEASURED IN SYSTEM SOLVES PER INPUT

	Training at Single Point in Parameter Space	Training at Multiple Points in Parameter Space
Exact Trajectories	$O(\gamma T)$	$O(\gamma T r^P)$
Approximate Trajectories	$O(T)$	$O(T r^P)$

of constructing the projection matrix is $O(\kappa m)$. If instead the PMORV vectors are chosen and the system has P parameters, then the cost of constructing $\mathbf{V}$ becomes $O(\kappa P^m)$. When the Krylov vectors are generated from every trajectory step, the costs become $O(Tm)$ and $O(TP^m)$, respectively, where T is the total number of time steps in a trajectory.

The exact training trajectories are created by solving the large nonlinear system at each time step. If each trajectory contains T points and each nonlinear solve requires γ Newton iterations, a single trajectory will cost $O(\gamma T)$ system solves. For the approximate trajectory algorithms, the cost of a single trajectory is reduced to $O(T)$ solves, as shown in Table IV. Finally, for a system with P parameters and r training values for each parameter, a single input will generate r^P different training trajectories.

IV. EXAMPLES

Three example systems were chosen to help illustrate the advantages of NLPMOR. All three examples are physical systems which contain strong nonlinearities that are distributed throughout the devices and possess dependence on several geometrical parameters. For each example, a derivation of the original system model is presented, followed by the results from our different algorithms.

A. Diode Transmission Line

The first example considered is a diode transmission line, which was used in the original TPWL papers [22], [23]. This allows for some relative accuracy comparisons between our new method and a well-established result in literature. The transmission line, shown in Fig. 1, is a nonlinear analog circuit containing a chain of strongly nonlinear diodes, resistors, and capacitors.

We chose the nodal voltages as the system state and derived the system equations using Kirchoff's current law and nodal

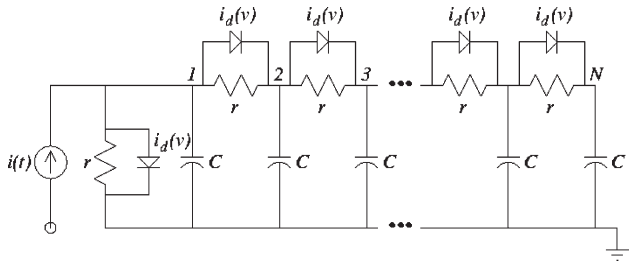

Fig. 1. Nonlinear transmission line circuit containing diodes [22], [23].

analysis. An equation for interior node j has the form

$$C\frac{dx_j}{dt} = \frac{x_{j-1} - 2x_j + x_{j+1}}{r} + I_d\left[e^{\frac{1}{v_\mathrm{T}}(x_{j-1}-x_j)} - e^{\frac{1}{v_\mathrm{T}}(x_j - x_{j+1})}\right] \quad (35)$$

leading to a state-space system of the form

$$\mathbf{E}\frac{d\mathbf{x}}{dt} = -\frac{1}{r}\mathbf{Q}^\mathrm{T}\mathbf{Q}\mathbf{x} - I_d\mathbf{Q}^\mathrm{T}\mathbf{d}(\mathbf{x}, v_\mathrm{T}) + \mathbf{b}u(t). \quad (36)$$

Here, $\mathbf{Q} \in \mathbb{R}^{N\times N}$ is the adjacency matrix for the resistor and diode networks, and $\mathbf{E} \in \mathbb{R}^{N\times N}$ is the capacitance matrix. Vector $\mathbf{d}(\mathbf{x}, v_\mathrm{T}) = -\mathbf{Q}^T\phi(\mathbf{x}, v_\mathrm{T})$, where $\phi(\mathbf{x}, v_\mathrm{T}) : \mathbb{R}\times\mathbb{R}^N \mapsto \mathbb{R}^N$. Its jth row is $\phi_j(\mathbf{x}, v_\mathrm{T}) = e^{(1/v_\mathrm{T})\mathbf{q}_j^T\mathbf{x}} - 1$, where $\mathbf{q}_j$ is the jth column of $\mathbf{Q}$. Vector $\mathbf{b} \in \mathbb{R}^N$ relates the state equations to the input which is an ideal current source $u(t) = i(t)$. All resistors have a value of 1 Ω, whereas all capacitors have a value of 10 pF. The constitutive relation for the diodes is $\phi(v) = I_d(e^{(1/v_\mathrm{T})v} - 1)$, where v_T is the threshold voltage, and v is the voltage across the device. Values of $I_d = 0.1$ nA and $v_t = 25$ mV were used as nominal values. Three parameters were considered for the diode transmission line: the resistor values r, the diode threshold voltage v_T, and the diode saturation current I_d. The situation is simplified if the parameters are defined as $p_\mathrm{G} = 1/r$, $p_V = 1/v_\mathrm{T}$, and $p_I = I_d$. Since (36) possesses nonlinear dependence on p_V, the system must first be expanded in powers of p_V. We chose to use a second-order expansion about the nominal value $p_V = (1/25\ \mathrm{mV}) = 40\ \mathrm{V}^{-1}$

$$\mathbf{E}\frac{d\mathbf{x}}{dt} = p_\mathrm{G}\mathbf{G}\mathbf{x} + p_I\mathbf{d}_0(\mathbf{x}) + p_I p_V\mathbf{d}_1(\mathbf{x}) + p_I p_V^2\mathbf{d}_2(\mathbf{x}) + \mathbf{b}u(t) \quad (37)$$

where

$$\begin{aligned}
\mathbf{G} &= -\mathbf{Q}^\mathrm{T}\mathbf{Q}\\
\mathbf{d}_0(\mathbf{x}) &= -\mathbf{Q}^\mathrm{T}\left[\mathbf{d}(\mathbf{x}, v_{\mathrm{T}_0}) - \frac{1}{v_{\mathrm{T}_0}}\frac{\partial\mathbf{d}(\mathbf{x}, v_{\mathrm{T}_0})}{\partial(\frac{1}{v_{\mathrm{T}_0}})} + \frac{1}{2v_{\mathrm{T}_0}^2}\frac{\partial^2\mathbf{d}(\mathbf{x}, v_{\mathrm{T}_0})}{\partial(\frac{1}{v_{\mathrm{T}_0}})^2}\right]\\
\mathbf{d}_1(\mathbf{x}) &= -\mathbf{Q}^\mathrm{T}\left[\frac{\partial\mathbf{d}(\mathbf{x}, v_{\mathrm{T}_0})}{\partial(\frac{1}{v_{\mathrm{T}_0}})} - \frac{1}{v_{\mathrm{T}_0}}\frac{\partial^2\mathbf{d}(\mathbf{x}, v_{\mathrm{T}_0})}{\partial(\frac{1}{v_{\mathrm{T}_0}})^2}\right]\\
\mathbf{d}_2(\mathbf{x}) &= -\mathbf{Q}^\mathrm{T}\frac{1}{2}\frac{\partial^2\mathbf{d}(\mathbf{x}, v_{\mathrm{T}_0})}{\partial(\frac{1}{v_{\mathrm{T}_0}})^2}.
\end{aligned}$$

Note that the system is still nonlinear in the state. To test our reduction algorithms, we created a reduced model

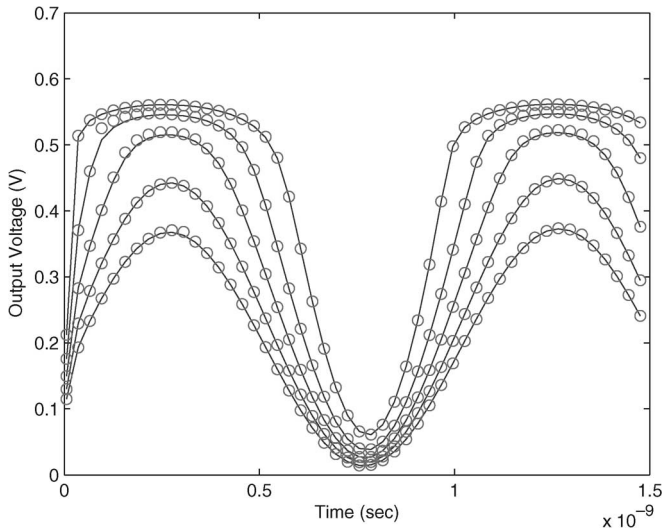

Fig. 2. Model created for the diode transmission line with original size $N = 200$ parameterized in $p_{\rm G} = 1/r$ by training at $p_{G0} = \{0.75p_{G0}, p_{G0}, 2p_{G0}\}$ with $p_{G0} = 1$. The PROM has size $q = 10$ and was simulated at a range of $p_{\rm G}$ values in the interval $[0.7p_{G0}, 2.5p_{G0}]$, resulting in a speedup of about $10\times$.

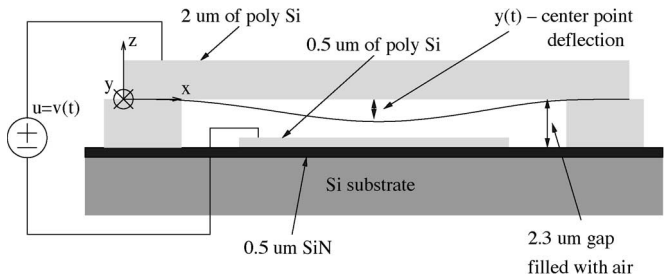

Fig. 3. MEM switch is a polysilicon beam fixed at both ends and suspended over a semiconducting pad and substrate [22], [23].

parameterized in $p_{\rm G}$. Fig. 2 compares the simulation output of the full nonlinear system with that of the PROM over a large range of parameter values which vary from the nominal value by as much as -30% and $+150\%$.

B. Micromachined Switch

The second example is a micromachined switch [22], [23]. The switch consists of a polysilicon fixed–fixed beam suspended over a polysilicon pad on a silicon substrate, as shown in Fig. 3. When a voltage is applied between the beam and the substrate, the electrostatic force generated pulls the beam down toward the pad. If the force is large enough, the beam will come into contact with the pad closing the circuit. In addition to being used as a switch, this device can be used as a pressure sensor due to its extreme sensitivity to the surrounding atmospheric conditions. The unknowns of interest in this system are the deflection of the beam $z(x,t)$ and the air pressure between the beam and the substrate $P(x,y,t)$. The system of equations is assembled by discretizing the coupled 1-D Euler's beam (38) and the 2-D Reynold's squeeze-film damping (39), taken from [22]. A finite difference scheme was used for the discretization, using m points for the length and n points for the width, 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 the pressure was discretized in the width

$$\hat{E}I_0h^3w\frac{\partial^4 z}{\partial x^4} - S_0hw\frac{\partial^2 z}{\partial x^2} = F_{\rm elec} + \int_0^w (P - P_{\rm a})dy - \rho_0 hw\frac{\partial^2 z}{\partial t^2} \tag{38}$$

$$\nabla\cdot\left((1+6K)z^3P\nabla P\right) = 12\mu\frac{\partial(Pz)}{\partial t}. \tag{39}$$

Here, $F_{\rm elec} = -(\epsilon_0 wv^2/wu^2)$ is the electrostatic force across the plates resulting from the applied voltage v, where v^2 is the input to the system. The beam is 2.2 μm above the substrate ($z_0 = 2.2$ μm), 610 μm in length, and has a width of 40 μm. The other constants are permittivity of free space $\epsilon_0 = 8.854\times 10^{-6}$ F/m, permeability $\mu = 1.82\times 10^{-5}$ kg/m $\cdot$ s, moment of inertia $I_0 = 1/12$, Young's modulus $\hat{E} = 149$ GPa, Knudsen number $K = \lambda/z_0$, $\lambda = 0.064$, stress coefficient $S_0 = -3.7$, and density $\rho_0 = 2300$ kg/m^3. The aforementioned equations can be separated into three partial differential equations

$$\begin{aligned}
\frac{\partial z}{\partial t} &= \frac{\partial^3 z}{\partial t^3}\frac{1}{3z^2}\\
\frac{\partial^4 z}{\partial t^4} &= \left(\frac{\partial^3 z}{\partial t^3}\right)^2\frac{2}{3z^3} - \frac{3\epsilon_0}{2\rho_0 h}v^2 + \frac{3z^2}{\rho_0 hw}S_0hw\frac{\partial^2 z}{\partial x^2}\\
&\quad + \frac{3z^2}{\rho_0 hw}\left[\int_0^w (P-P_{\rm a})dy - EIh^3w\frac{\partial^4 z}{\partial x^4}\right]\\
\frac{\partial P}{\partial t} &= -\frac{\partial^3 z}{\partial t^3}\frac{P}{3z^3} + \frac{1}{12\,\mu z}\nabla\left(\left(1+6\frac{\lambda}{z}\right)z^3P\nabla P\right).
\end{aligned}$$

We choose the state-space variables to be $\mathbf{x}_1 \in \mathbb{R}^m = z$, $\mathbf{x}_2 \in \mathbb{R}^m = \partial u^3/\partial t$, and $\mathbf{x}_3 \in \mathbb{R}^{mn} = P$, and the parameters to be Young's modulus $p_E = E$ and stress coefficient $p_S = S$. Rearranging the discretized system equations to obtain linearity in each parameter results in the system

$$\begin{aligned}
\frac{\partial \mathbf{x}_1}{\partial t} &= \mathbf{f}_{10}(\mathbf{x}_1, \mathbf{x}_2)\\
\frac{\partial \mathbf{x}_2}{\partial t} &= \mathbf{f}_{20}(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3) + p_S\mathbf{f}_{21}(\mathbf{x}_1, \mathbf{x}_2) + p_E\mathbf{f}_{22}(\mathbf{x}_1) + \mathbf{b}u(t)\\
\frac{\partial \mathbf{x}_3}{\partial t} &= \mathbf{f}_{30}(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3)
\end{aligned}$$

where the total system has an order $N = m(n+2)$, $\mathbf{f}_{10} \in \mathbb{R}^m$, $\mathbf{f}_{20}, \mathbf{f}_{21}, \mathbf{f}_{22} \in \mathbb{R}^m$, and $\mathbf{f}_{30} \in \mathbb{R}^{mn}$. A detailed description of these functions can be found in [33]. The beam is fixed at both ends and initially in equilibrium; therefore, the applied boundary conditions are

$$z(x,0) = z_0, \quad P(x,y,0) = P_{\rm a}, \quad z(0,t) = z(l,t) = z_0. \tag{40}$$

Other constraints enforced are

$$\frac{\partial P(0,y,t)}{\partial x} = \frac{\partial P(l,y,t)}{\partial x} = 0, \quad P(x,0,t) = P(x,w,t) = P_{\rm a} \tag{41}$$

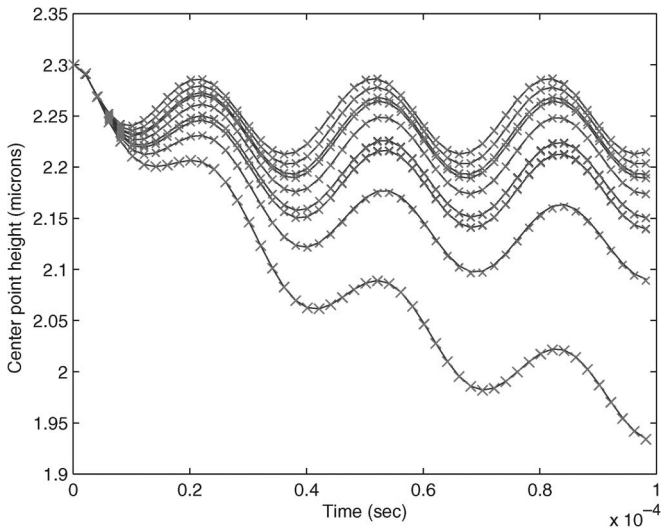

Fig. 4. Output of a micromachined switch model parameterized in p_E and p_S and simulated at nine different sets of parameter values on an evenly spaced grid where $p_E \in [0.6p_{E0}, 1.4p_{E0}]$, and $p_S \in [0.6p_{S0}, 1.4p_{S0}]$. The solid lines represent the original model with order $N = 144$, the crosses represent the reduced model of order $q = 20$, the resulting speedup in simulation was about $15\times$, and the nominal parameter values are $[p_{E0}, p_{S0}] = [1.49 \times 10^5, -3.7]$.

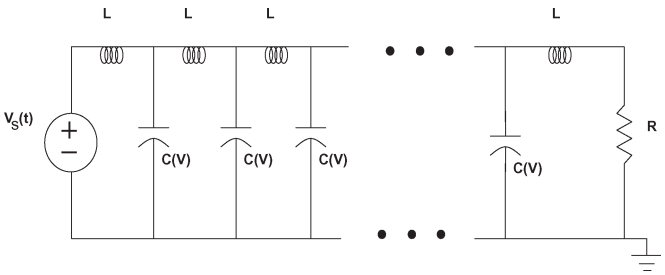

Fig. 5. Pulse-narrowing transmission line circuit containing nonlinear capacitors [34].

where the initial height and pressure are $z_0 = 2.3\ \mu$m and $P_{\rm a} = 1.103 \times 10^5$ Pa, respectively. Typical inputs for this system are sinusoids $u(t) = (v\cos(\omega t))^2$ with $\omega = (10\pi/30)$ MHz and $v = 7$ or a step input $u(t) = v^2$ for $t > 0$ with $v = 7$. The system output is the deflection of the beam center point.

For this example, a reduced model parameterized in p_E and p_S was created by training with the sinusoidal inputs. The model was then simulated at nine different sets of parameter values on an evenly spaced grid with each parameter varying up to $\pm 40\%$ from the nominal values. The outputs from the simulation along with the output of the full nonlinear system are shown in Fig. 4.

C. Pulse-Narrowing Transmission Line

The final example considered is a nonlinear transmission line used for signal shaping. One example of such a line, shown in Fig. 5, contains distributed nonlinear capacitors. The resulting wave equation for this transmission line contains a nonlinear term which sharpens the peaks in a wave traveling down the line. Hence, these devices may be useful in pulse-narrowing applications. A thorough analysis of this line can be found in [34]. The nonlinearity arises from the voltage dependence of the capacitors $C_n = C(V_n) \approx C_0(1 - b_{\rm c}V_n)$. By setting the system state to the node voltages and branch currents, the 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 $y(t) = V_m(t)$. By using this formulation, the system equations for an interior node n would be of the form

$$C_n(V_n)\frac{dV_n}{dt} = I_{n-1} - I_n \tag{42}$$

$$L_n\frac{dI_n}{dt} = V_n - V_{n+1} \tag{43}$$

leading to the state-space model

$$\begin{bmatrix}\dot{\mathbf{x}}\\ \dot{\mathbf{z}}\end{bmatrix} = \begin{bmatrix}\frac{1}{C_0}\mathbf{f}_V(\mathbf{x},\mathbf{z})\\ \frac{1}{L}\mathbf{f}_I(\mathbf{x},\mathbf{z})\end{bmatrix} + \begin{bmatrix}0\\ \frac{1}{L}\mathbf{b}\end{bmatrix}u(t) \tag{44}$$

where the nth equations of $\mathbf{f}_V$ and $\mathbf{f}_I$ are

$$\mathbf{f}_{Vn}(\mathbf{x},\mathbf{z}) = \frac{z_{n-1} - z_n}{1 - b_{\rm c}x_n} \tag{45}$$

$$\mathbf{f}_{In}(\mathbf{x},\mathbf{z}) = x_n - x_{n+1}. \tag{46}$$

Here, $\mathbf{b}$ is the vector of voltage-source inputs. Typical capacitor and inductor values are 100 pF and 100 pH, respectively. Parameters of interest for the pulse-narrowing transmission line are the inductor values, the capacitor values, and $b_{\rm c}$ which is a parameter that adjusts the nonlinearity of the line. These three parameters all affect the shaping of the wave as it travels down the line. For this example, PROMs were created by training with a sinusoidal input with $u(t) = v\sin(\omega t)$ at a frequency of 5 GHz. The PMOR moment matching generated moments about parameter expansion points equal to the parameter values used in the training. To test this example, we parameterized the system in $p_C = 1/C$ and $p_L = 1/L$, resulting in a system of the form

$$\begin{bmatrix}\dot{\mathbf{x}}\\ \dot{\mathbf{z}}\end{bmatrix} = p_L\left(\begin{bmatrix}0\\ \mathbf{f}_I(\mathbf{x},\mathbf{z})\end{bmatrix} + \begin{bmatrix}0\\ \mathbf{b}_1\end{bmatrix}u(t)\right) + p_C\begin{bmatrix}\mathbf{f}_V(\mathbf{x},\mathbf{z})\\ 0\end{bmatrix}. \tag{47}$$

Fig. 6 compares the output of the full system and a PROM for the pulse-narrowing transmission line simulated at five different parameter values varying as much as -90% and $+100\%$.

V. Algorithm Comparative Analysis

In this section, we examine the accuracy of models created with the different linearization and projection schemes from Tables I and II. Specifically considered is how the different linearization and projection options affect the accuracy of the PROM in the parameter space. We also wish to determine whether the parameter-space accuracy of the PROM is limited by the original linearization of the nonlinear system with respect to the parameters.

A. Training in the Parameter Space

The effects of training at different points in the parameter space (described in Section III-B) can be seen by comparing the $T_{\rm xm}V_{\rm xx}$ models with the $T_{\rm xs}V_{\rm xx}$ models. As explained at

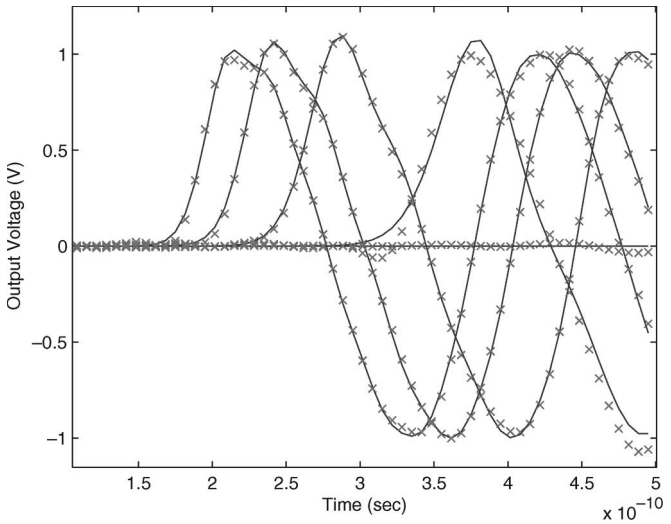

Fig. 6. Output from a model of the pulse-narrowing transmission line simulated at five different values of $p_L = 1/L$ on the interval $[0.1p_{L0}, 2p_{L0}]$, where $p_{L0} = 10^{11}$. The model was reduced from the large order $N = 200$ to the reduced order $q = 50$ which resulted in a speedup of $\sim 5\times$.

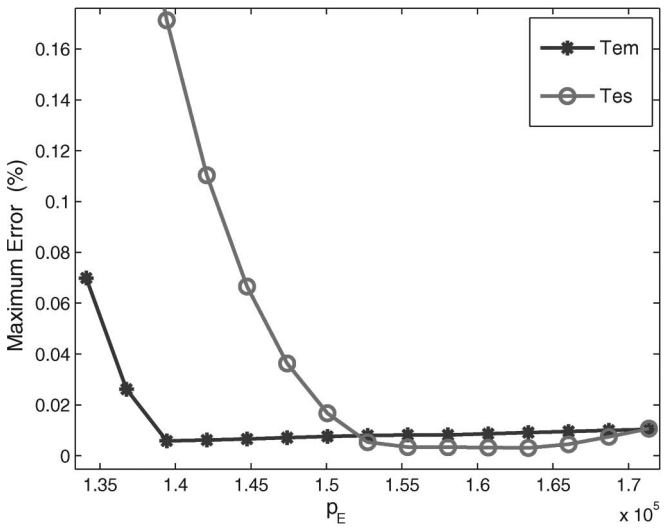

Fig. 7. $T_{\rm em}V_{\rm pl}$ and $T_{\rm es}V_{\rm pl}$ models of the micromachined switch example parameterized in p_E and simulated over a range of parameter values. Each model was reduced from the large order $N = 150$ to the reduced order $q = 30$.

the end of Section III-E, we use the notations from Tables I and II to identify different kinds of models. Trajectories created with different parameter values will likely evolve in different regions of the state space, thus resulting in different collections of linear models. The first test compares the $T_{\rm em}V_{\rm pl}$ and $T_{\rm es}V_{\rm pl}$ models of the micromachined switch. By considering p_E as the parameter, one model was created by training at $p_E = p_{E0} = 149$ GPa, and the other was created by training at $p_E = [0.95p_{E0}, 1.05p_{E0}]$. The projection matrices for both models were created by matching the parameter moments at E_0 and the frequency moments at the input frequency $f = 1$ GHz. The models were simulated at a set of parameter values in the range $[0.9p_{E0}, 1.1p_{E0}]$. Fig. 7 compares the maximum percent error for each model, which is defined as

$$\max_t \left(\frac{|y(t) - \hat{y}(t)|}{|y(t)|} \right) \times 100. \tag{48}$$

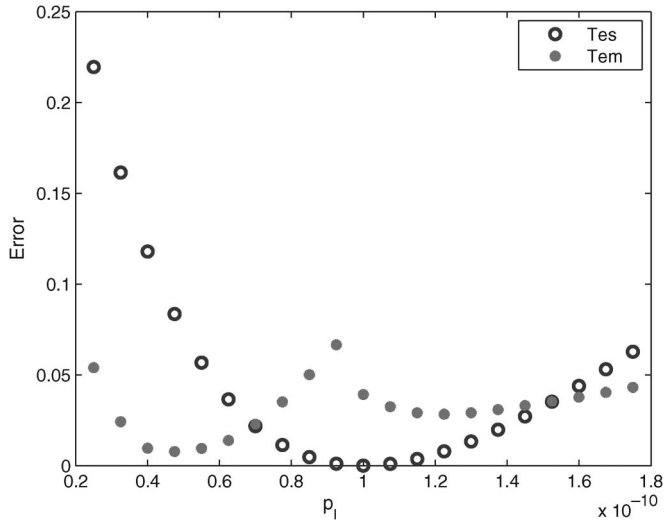

Fig. 8. Norm of the error, as defined in (49), over time for a $T_{\rm es}V_{\rm ml}$ model trained at $p_I = 10^{-10}A$ and a $T_{\rm em}V_{\rm ml}$ model trained at $[0.5p_{I_0}, 1.3p_{I_0}]$. The system was reduced from the original order $N = 100$ to the reduced order $q = 50$.

A similar comparison is made in Fig. 8 with $T_{\rm es}V_{\rm ml}$ and $T_{\rm em}V_{\rm ml}$ models of the diode transmission line parameterized in p_I. In this figure, the error plotted is the norm of $e(t)$, where

$$e(t) = |y(t) - \hat{y}(t)|. \tag{49}$$

These models were constructed by training at $p_I = p_{I_0} = 0.1$ nA for $T_{\rm es}$ and $p_I = [0.5p_{I_0}, 1.3p_{I_0}]$ for $T_{\rm em}$.

Both Figs. 7 and 8 show that the greatest accuracy occurs close to 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 is more accurate in a larger region around the training values.

B. Parameterizing the Projection Matrix

The benefits of parameterizing the projection matrix via PMOR moment matching, as in Section III-C, can be examined by comparing the $T_{\rm xx}V_{\rm mx}$ models with the $T_{\rm xx}V_{\rm px}$ models.

Fig. 9 compares the total simulation error at different parameter values for the $T_{\rm es}V_{\rm ml}$ and $T_{\rm es}V_{\rm pl}$ models of the diode transmission line parameterized in p_I. As with the parameter-space training, this figure suggests that the $V_{\rm ml}$ model is more accurate close to the nominal parameter value, whereas the $V_{\rm pl}$ model is less accurate at the nominal value but more accurate over a larger range of parameter values.

C. Krylov Vectors From Extra Models

To determine whether the linear models created during the training produce the Krylov vectors which span a near-optimal-reduced space, we compare the $T_{\rm xx}V_{\rm xl}$ models with the $T_{\rm xx}V_{\rm xp}$ models. Both PROMs contain the same number of linear models κ and have the same reduced order q. Fig. 10 compares the output from these two models. The results, however, are system-dependent.

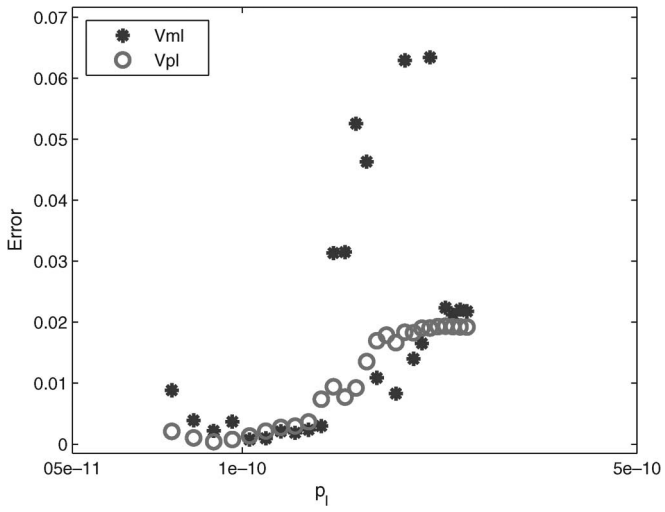

Fig. 9. Norm of the error, as defined in (49), for two ROMs of the diode transmission line parameterized in p_I and simulated at a range of parameter values using sinusoidal inputs. The models are $T_{\rm es}V_{\rm ml}$ and $T_{\rm es}V_{\rm pl}$ and were reduced from the large order $N = 100$ to the reduced order $q = 40$.

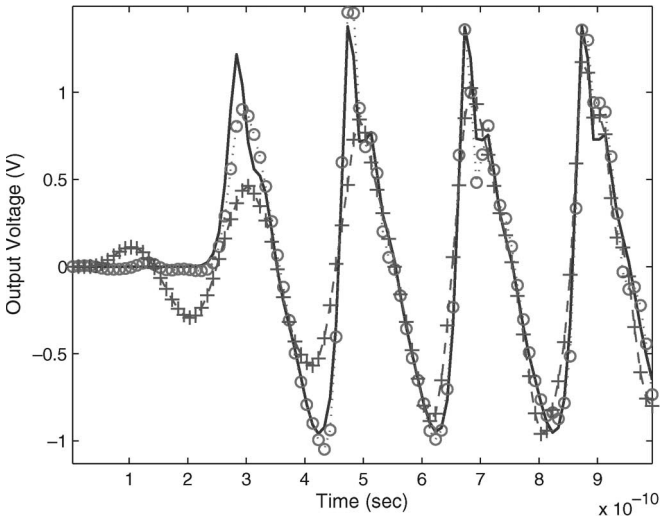

Fig. 10. Two models of the pulse-narrowing transmission line parameterized in p_L. The circles use vectors from every point of the trajectories, and the crosses use vectors only from the $k = 181$ linear models. In both cases, an SVD was used on $\mathbf{V}$, and both models were projected from the large order $N = 200$ down to the same reduced order $q = 50$.

We also considered the diode transmission line parameterized in $p_{\rm R}$, and in this case, there is no discernable advantage to the $V_{\rm xa}$ model. In general, we suspect that the $V_{\rm xa}$ model will not be less accurate than the $V_{\rm xl}$ model. Before the SVD in step 32 of Algorithm 1, the $V_{\rm xa}$ projection matrix contains all of the columns in the $V_{\rm xl}$ projection matrix. Therefore, from a practical point of view, after SVD, the $V_{\rm xa}$ projection matrix will correspond to a subspace at least approximately as good as the projection matrix from the $V_{\rm xl}$ model. However, theoretically, it is important to note here that using a projection matrix constructed using an SVD in this manner can no longer guarantee an absolutely exact match of transfer-function moments between the original linearized models and the reduced models.

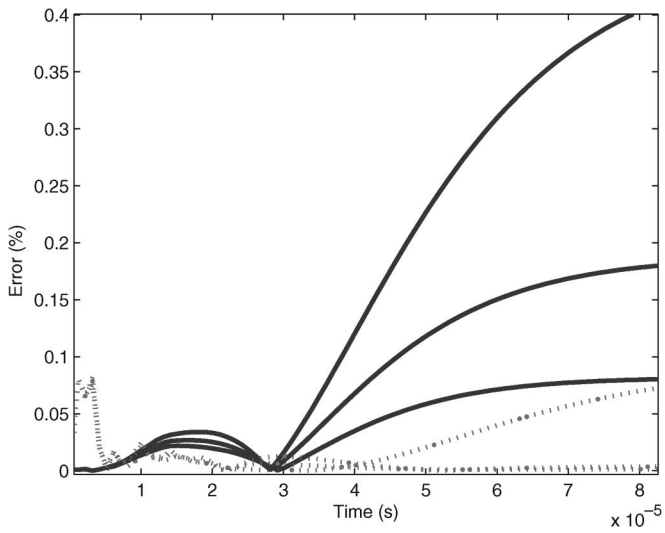

Fig. 11. Percent error in the output of the reduced models of micromachined switch. The solid curves correspond to the models constructed with approximate training trajectories ($T_{\rm ax}$ models), whereas the dashed curves correspond to the models constructed with exact training trajectories ($T_{\rm ex}$ models). Both models were then simulated at three different parameter values.

D. Approximate Training Trajectories

Generating exact training trajectories can be often very expensive. Alternatively, one could instead use approximate training trajectories. In this section, we compare the two approaches examining the $T_{\rm ax}V_{\rm xx}$ models and the $T_{\rm ex}V_{\rm xx}$ models.

By using the micromachined switch example parameterized in p_E, the $T_{\rm es}V_{\rm ml}$ and $T_{\rm as}V_{\rm ml}$ models were created. The two models were then simulated at three different parameter values close to the training values. Fig. 11 compares the percent error of the PROM output for the two models.

Although the model created with exact trajectories is more accurate, both models still produce outputs with a maximum error smaller than 0.5%.

E. Effects of Linearizing in Parameters

Lastly, we consider the effects of linearizing the original nonlinear system (18) with respect to the parameters (22). An important question to ask is whether the dominant factor in determining the accuracy of the PROM is a result of projecting the system into a low-order subspace or a result of this linearization in the parameters.

To investigate this, we considered the diode transmission line with parameter p_V. Since the original system was nonlinear in p_V, (36) was expanded to second order about some nominal value p_{V_0} to obtain system (37) which is linear in powers of p_V but still nonlinear in the state $\mathbf{x}$. A model was created using sinusoids as training inputs and a nominal parameter value $p_{V_0} = 40$ for expansion and training. Fig. 12 compares the output error at different parameter values between the original system (18), the model expanded in powers of the parameters (22), and the PROM (24). In this case, we define the error $e_m(\mathbf{p})$ as

$$e_m(\mathbf{p}) = \left(\frac{\max_t |y(t) - y_0(t)|}{\max_t |y_0(t)|} \right) \times 100 \qquad (50)$$

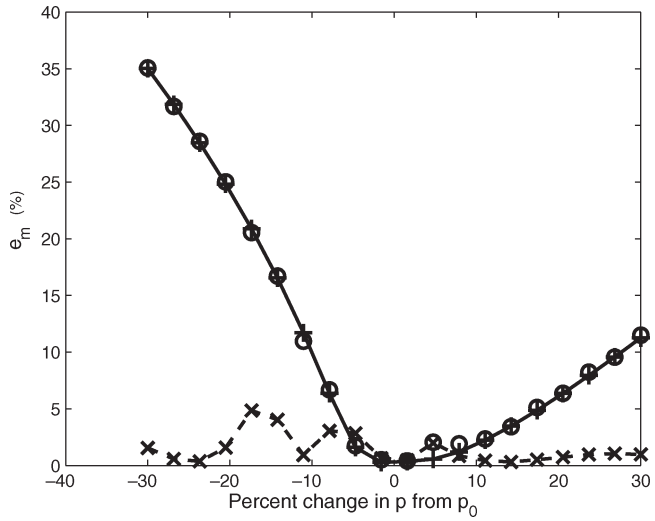

Fig. 12. Output error, as defined in (50), between three different diode line models parameterized in p_V and simulated over a range of parameter values. The pluses correspond to the error between systems (18) and (24), the circles correspond to the error between systems (18) and (22), and the crosses correspond to the error between systems (22) and (24).

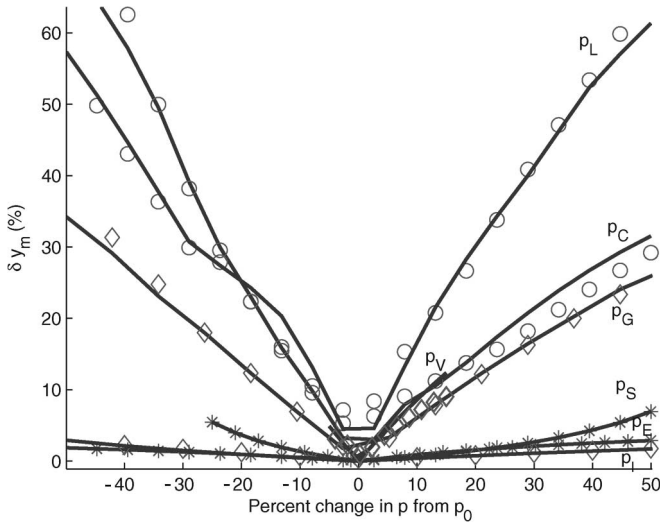

Fig. 13. Output sensitivity of models to parameter changes, as defined in (50). The solid lines represent the original systems, and the symbols represent the associated PROMs. Several parameters were considered for each example system, with the circles corresponding to the pulse-narrowing transmission line, the stars corresponding to the MEM switch, and the diamonds corresponding to the diode line.

where $y(t)$ is the output of one system at parameter value p, and $y_0(t)$ is the output of the other system at parameter value p.

It can be seen that for this particular case, the PROM error is not significantly worse than the error from the large nonlinear system which was expanded in powers of the parameters (22). This indicates that both aspects of the reduction process, i.e., finding a good subspace and selecting linearization points, worked well for this example. However, the accuracy of both models compared with the original system (18) declines rapidly as the parameter value moves beyond $\pm 10\%$. For this example, we can conclude that if an accuracy over a larger range of parameter values is needed, a higher order expansion in the parameter would be required.

F. Sensitivity to Parameters

To determine how accurately the parameter dependence of the original system is captured in the PROM, we can compare output sensitivity with the changes in the parameters for both the original system and the PROMs. Fig. 13 compares these sensitivities for several parameters for each of the three example systems. We define the sensitivity $\delta y(\mathbf{p})$ as

$$\delta y(\mathbf{p}) = \left(\frac{\max_t |y(t) - y_0(t)|}{\max_t |y_0(t)|} \right) \times 100 \tag{51}$$

where $y(t)$ is the system output at parameter value p, and $y_0(t)$ is the system output at nominal parameter value p_0.

The figure shows that the PROMs do, in fact, capture the parameter dependence of the original system over a significant range of parameter values. The exact range of values depends on the system and the parameter considered, as the system sensitivity is different for each parameter.

To validate our parameter-selecting training scheme in Section III-D, we approximate the gradient of the state with respect to the parameters $\partial \mathbf{x}/\partial \mathbf{p}$ and examine whether it lies in the subspace spanned by the columns of the projection matrix $\mathbf{V}$. By the fundamental theorem of linear algebra, this can be determined by checking if $\partial \mathbf{x}/\partial \mathbf{p}$ is orthogonal to the left null space of $\mathbf{V}$, which we define as

$$e_p(\mathbf{p}) = \left\| \left(\frac{\partial \tilde{\mathbf{x}}}{\partial \mathbf{p}} \right)^{\mathrm{T}} \mathcal{N}(\mathbf{V}^{\mathrm{T}}) \right\|. \tag{52}$$

Here, $\partial \tilde{\mathbf{x}}/\partial \mathbf{p}$ is the largest three singular vectors of $\partial \mathbf{x}/\partial \mathbf{p}$ as computed in (30). Both the singular vectors and $\mathcal{N}(\mathbf{V}^{\mathrm{T}})$ are normalized; hence, $e_p(\mathbf{p}) \in [0, 1]$ with $e_p(\mathbf{p}) = 0$ corresponding to $\partial \tilde{\mathbf{x}}/\partial \mathbf{p}$ exactly in the subspace spanned by the columns of $\mathbf{V}$, and $e_p(\mathbf{p}) = 1$ corresponding to $\partial \tilde{\mathbf{x}}/\partial \mathbf{p}$ exactly orthogonal to the subspace spanned by the columns of $\mathbf{V}$. We have computed this quantity at three different parameter values for each parameter in each system and compared the results in Table V.

TABLE V
EQUATION (52) COMPUTED ON THREE DIFFERENT TRAJECTORIES, CORRESPONDING TO $0.5p_0$, p_0, AND $1.5p_0$, FOR EACH PARAMETER IN OUR THREE EXAMPLES

	$0.5p_0$	p_0	$1.5p_0$
p_G	0.55	0.58	0.56
p_V	0.31	0.06	0.46
p_I	0.031	0.027	0.025
p_E	0.17	0.18	0.070
p_S	0.17	0.23	0.18
p_L	0.81	0.86	0.90
p_C	0.66	0.52	0.47

The results of this test indicate that in some cases, such as for parameters p_E and p_S in the MEM switch, we do not need to train at additional parameter values to capture the parameter dependence of the original system. It also shows that in other cases, such as for parameters p_L and p_C in the pulse-narrowing transmission line, increasing the range of the parameter values

will take the trajectory to a significantly different subspace, and the reduced model would need to be updated by training that system at the additional parameter values. In general, these results match with what we experienced in the training process, as we found the pulse-narrowing transmission line to be the most difficult system to model and the MEM switch to be the easiest. As a matter of fact, we can observe a correlation between the results in Table V and the sensitivities shown in Fig. 13. Both tests indicate that the pulse-narrowing transmission line is the most sensitive to changes in the parameters and that the MEM switch is least sensitive to the parameter changes.

VI. Conclusion

We have presented an algorithm for the parameterized model-order reduction of highly nonlinear systems, which consists of a linearization scheme and a projection scheme, each of which is chosen from four possible options. The different approaches were tested on three examples: a diode transmission line, a MEM switch, and a pulse-narrowing transmission line.

For a PROM of order q containing κ linear models, the parameter-space accuracy of the model is characterized by the distribution of the κ linearization points across the state space and by the q vectors which define the column span of the projection matrix. We have found that high local parameter-space accuracy can be achieved by placing all κ linearization points on the trajectories resulting from training at a single parameter-space point or by creating the Krylov vectors using a single variable Taylor series expansion. Alternatively, higher global parameter-space accuracy can be attained by placing the linearization points on trajectories created by training at multiple points in the parameter space and by using a multivariable Taylor series expansion to create vectors for the projection matrix.

Finally, we have shown that the generated reduced models can capture the system sensitivity to the different parameters, and these sensitivities can be approximated and used for the purpose of the parameter-space training point selection.

References

[1] S. Pullela, N. Menezes, and L. T. Pileggi, "Moment-sensitivity-based wire sizing for skew reduction in on-chip clock nets," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 16, no. 2, pp. 210–215, Feb. 1997.

[2] Y. Liu, L. T. Pileggi, and A. J. Strojwas, "Model order-reduction of RCL interconnect including variational analysis," in *Proc. ACM/IEEE Des. Autom. Conf.*, New Orleans, LA, Jun. 1999, pp. 201–206.

[3] P. Heydari and M. Pedram, "Model reduction of variable-geometry interconnects using variational spectrally-weighted balanced truncation," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Des.*, San Jose, CA, Nov. 2001, pp. 586–591.

[4] 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. ACM/IEEE Des. Autom. Conf.*, Jun. 2002, pp. 437–442.

[5] D. S. Weile, E. Michielssen, E. Grimme, and K. Gallivan, "A method for generating rational interpolant reduced order models of two-parameter linear systems," *Appl. Math. Lett.*, vol. 12, no. 5, pp. 93–102, Jul. 1999.

[6] P. Gunupudi and M. Nakhla, "Multi-dimensional model reduction of VLSI interconnects," in *Proc. Custom Integr. Circuits Conf.*, Orlando, FL, 2000, pp. 499–502.

[7] 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," *J. Fluids Eng.*, vol. 124, no. 1, pp. 70–80, Mar. 2002.

[8] L. Daniel and J. White, "Automatic generation of geometrically parameterized reduced order models for integrated spiral RF-inductors," in *Proc. Int. Workshop Behav. Modeling Simul.*, San Jose, CA, Sep. 2003, pp. 18–23.

[9] 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. Comput.-Aided Design Integr. Circuits Syst.*, vol. 23, no. 5, pp. 678–693, May 2004.

[10] P. Li, F. Liu, S. Nassif, and L. Pileggi, "Modeling interconnect variability using efficient parametric model order reduction," in *Proc. Des., Autom. Test Eur.*, Mar. 2005, pp. 958–963.

[11] X. Li, P. Li, and L. T. Pileggi, "Parameterized interconnect order reduction with explicit-and-implicit multi-parameter moment matching for inter/intra-die variations," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Des.*, San Jose, CA, 2005, pp. 806–812.

[12] J. R. Phillips, "Variational interconnect analysis via PMTBR," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Des.*, Nov. 2004, pp. 872–879.

[13] K. C. Suo, A. Megretski, and L. Daniel, "A quasi-convex optimization approach to parameterized model order reduction," in *Proc. ACM/IEEE Des. Autom. Conf.*, Anaheim, CA, Jun. 2005, pp. 933–938.

[14] M. Celik, A. Atalar, and M. A. Tan, "Transient analysis of nonlinear circuits by combining asymptotic waveform evaluation with Volterra series," *IEEE Trans. Circuits Syst. I, Fundam. Theory Appl.*, vol. 42, no. 8, pp. 470–473, Aug. 1995.

[15] J. Chen and S. M. Kang, "An algorithm for automatic model-order reduction of nonlinear MEMS devices," in *Proc. ISCAS*, 2000, pp. 445–448.

[16] J. R. Phillips, "Projection frameworks for model reduction of weakly nonlinear systems," in *Proc. ACM/IEEE Des. Autom. Conf.*, Jun. 2000, pp. 184–189.

[17] J. R. Phillips, "Automated extraction of nonlinear circuit macromodels," in *Proc. Custom Integr. Circuit Conf.*, Orlando, FL, May 2000, pp. 451–454.

[18] J. R. Phillips, "Projection-based approaches for model reduction of weakly nonlinear, time-varying systems," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 22, no. 2, pp. 171–187, Feb. 2003.

[19] P. Li and L. T. Pileggi, "Norm: Compact model order reduction of weakly nonlinear systems," in *Proc. ACM/IEEE Des. Autom. Conf.*, Jun. 2003, pp. 472–477.

[20] Y. Chen, "Model order reduction for nonlinear systems," M.S. thesis, MIT, Cambridge, MA, Sep. 1999.

[21] 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. IEEE/ACM Int. Conf. Comput.-Aided Des.*, San Jose, CA, Nov. 2001, pp. 252–257.

[22] 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. Comput.-Aided Design Integr. Circuits Syst.*, vol. 22, no. 2, pp. 155–170, Feb. 2003.

[23] 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. ACM/IEEE Des. Autom. Conf.*, Jun. 2003, pp. 490–495.

[24] N. Dong and J. Roychowdhury, "Piecewise polynomial nonlinear model reduction," in *Proc. ACM/IEEE Des. Autom. Conf.*, Jun. 2003, pp. 484–489.

[25] N. Dong and J. Roychowdhury, "Automated extraction of broadly applicable nonlinear analog macromodels from spice-level descriptions," in *Proc. IEEE Custom Integr. Circuits Conf.*, Oct. 2004, pp. 117–120.

[26] B. Bond and L. Daniel, "Parameterized model order reduction for nonlinear dynamical systems," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Des.*, San Jose, CA, 2005, pp. 487–494.

[27] A. Odabasioglu, M. Celik, and L. T. Pileggi, "PRIMA: Passive reduced-order interconnect macromodeling algorithm," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 17, no. 8, pp. 645–654, Aug. 1998.

[28] E. Grimme, "Krylov projection methods for model reduction," Ph.D. dissertation, Coordinated-Sci. Lab., Univ. Illinois Urbana–Champaign, Urbana, IL, 1997.

[29] S. Tiwary and R. Rutenbar, "Scalable trajectory methods for on-demand analog macromodel extraction," in *Proc. 42nd ACM/IEEE Des. Autom. Conf.*, Jun. 2005, pp. 403–408.

[30] N. Dong and J. Roychowdhury, "Automated nonlinear macromodelling of output buffers for high-speed digital applications," in *Proc. 42nd ACM/IEEE Des. Autom. Conf.*, Jun. 2005, pp. 51–56.

[31] B. Moore, "Principal component analysis in linear systems: Controllability, observability, and model reduction," *IEEE Trans. Autom. Control*, vol. AC-26, no. 1, pp. 17–32, Feb. 1981.

[32] K. Willcox and J. Peraire, "Balanced model reduction via the proper orthogonal decomposition," in *Proc. 15th AIAA Comput. Fluid Dyn. Conf.*, Anaheim, CA, Jun. 2001. Paper 2001–2611.
[33] M. Rewienski, "A trajectory piecewise-linear approach to model order reduction of nonlinear dynamical systems," Ph.D. dissertation, MIT, Cambridge, MA, Jun. 2003.
[34] E. Afshari and A. Hajimiri, "A non-linear transmission line for pulse shaping in silicon," *IEEE J. Solid-State Circuits*, vol. 40, no. 3, pp. 744–752, Mar. 2005.

Bradley N. Bond (S'07) received the B.S. degree in engineering science and mechanics from The Pennsylvania State University, University Park, in 2004, and the M.S. degree in electrical engineering from the Massachusetts Institute of Technology, Cambridge, in 2006, where he is currently working toward the Ph.D. degree in the Computational Prototyping Group, Department of Electrical Engineering and Computer Science.

His research interests include model-order reduction and nonlinear systems.

Luca Daniel (S'98–M'04) received the Laurea degree in electronic engineering (*summa cum laude*) from the Universita di Padova, Padova, Italy, in 1996, and the Ph.D. degree in electrical engineering from the University of California (UC), Berkeley, in 2003.

In 1997, he was with the STMicroelectronics Berkeley Laboratories, Berkeley, CA. In 1998, he was with the HP Research Laboratories, Palo Alto, CA, and in 2001, he was with the Cadence Berkeley Laboratories, Berkeley. He is currently an Assistant Professor with the Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA, and a Principal Investigator of the Research Laboratory of Electronics Computational Prototyping Group. His research interests include parameterized model-order reduction of linear and nonlinear dynamical systems, mixed signal, RF and millimeter-wave circuit design and robust optimization, power electronics, microelectromechanical design and fabrication, and parasitic extraction and accelerated integral equation solvers.

Dr. Daniel was the recipient of the 2003 ACM Outstanding Ph.D. Dissertation Award in Electronic Design Automation and the Best Thesis Awards from both the Department of Electrical Engineering and Computer Science and the Department of the Applied Math at UC Berkeley. He was also the recipient of four Best Paper Awards in conferences and the IEEE Power Electronic Society Prize Paper Award for the best paper published on the IEEE TRANSACTIONS ON POWER ELECTRONICS in 1999.