Model Reduction via Projection onto Nonlinear Manifolds, with Applications to Analog Circuits and Biochemical Systems

Chenjie Gu and Jaijeet Roychowdhury
{gcj,jr}@umn.edu
ECE Department, University of Minnesota, Twin Cities

***Abstract*— Previous model order reduction methods fit into the framework of identifying the low-order linear subspace and using the linear projection to project the full state space into the low-order subspace. Despite its simplicity, the macromodel might automatically include redundancies.**

In this paper, we present a model order reduction approach, named *maniMOR*, which extends the linear projection framework to a general nonlinear projection framework. The two key ideas of *maniMOR* are (1) it explicitly separates the construction of the low-order subspace and projection operation; (2) it constructs a nonlinear manifold which captures important system responses and defines the corresponding nonlinear projection operator.

The low-order manifold subspace in *maniMOR* is identified by stitching together the low-order linear subspaces around a set of sample points on the manifold. After the manifold is determined, it is embedded into a global nonlinear coordinate system. The projection function is defined in a piece-wise linear manner, and the model evaluation is conducted directly in the manifold subspace using cheap matrix-vector product computations. As a result, a compact model is generated by pre-computing all the functions and Jacobians and storing them in a look-up table.

We apply *maniMOR* on two analog circuits and a bio-chemical system to validate its correctness. Extensive comparisons with the results of the full model and other macromodels are provided. Experimental results show that *maniMOR* manages to obtain a huge reduction – *e.g.*, from 52 to 5 for the I/O buffer circuit and from 304 to 30 for yeast pheromone pathway system. This is less than half of the size of the TPWL model with the same accuracy. With great promise to capture important system responses, *maniMOR* presents a novel and powerful paradigm for nonlinear model reduction, and casts inspirations for further researches.

I. Introduction

Model order reduction (MOR) has been an active research topic in CAD area for a long time. The goal of model order reduction is to automatically extract a smaller macromodel for a subcircuit/subsystem, thus enabling fast simulations/verifications of large complex systems. MOR for bio-chemical systems is also of great importance and has received much more concern than ever. Those systems have the feature that even a very small system has hundreds of reactants reacting with each other. Therefore the size of the corresponding differential equations describing the chemical reaction kinetics [1] is extremely large. In this context, macromodels enable the simulation of large higher-level chemical systems.

So far, MOR techniques for linear time invariant systems have been well-developed and widely used, such as Krylov subspace methods [2], [3], TBR methods [4], [5], and the combination of the two [6], [7]. On the other hand, nonlinear systems present a lot of challenges for MOR, and much less robust, efficient, and generally-applicable methods are available. Some of the most influential works include the TVP method [8] for time varying systems, projection-based methods based on linearization or bilinearization [9] for weakly nonlinear time-varying systems, and trajectory-based methods [10]–[13]. Almost all existing methods are based on the idea of projecting the full state space on a linear subspace, where the system dynamics evolve. However, macromodels generated under this framework seem not to achieve the most reduction in size.

In this paper, we present an approach, named *maniMOR*, to nonlinear model order reduction, based on nonlinear manifold projection. Employing *nonlinear* projection of the full state space into a low-order *nonlinear* manifold/subspace, instead of *linear* projection into *linear* subspace in previous works, *maniMOR* is able to extract a smaller macromodel of a nonlinear system. *maniMOR* first constructs the nonlinear manifold in a way such that it captures the main behavior of the original system. (*e.g.*, asymptotic DC response, small signal AC response, frequency conversion and distortion effects, *etc.*) In the course of the manifold construction, a coordinate system in the manifold subspace is built up. Accordingly, the nonlinear projection/mapping between the original space and the manifold subspace is defined – since the projection operator becomes nonlinear, it is defined and stored in a piece-wise linear fashion, rather than a single projection matrix in the linear case. Finally, in order to achieve computation reduction of the model, compact computation of the macromodel directly in the manifold coordinate is ensured by simplifying the function computation to cheap matrix-vector product operations.

The first main contribution of this paper is to extend the *linear* projection framework of previous methods to a general *nonlinear* projection framework, and to demonstrate that through nonlinear projection into a nonlinear manifold, more reduction can be obtained than its linear counterpart. In this view point, methods based on linear projection can be regarded as a special case in the nonlinear projection framework.

The second main contribution of this paper is that we propose to explicitly split the MOR procedure into two sub-problems, *i.e.*, (1) the construction/parameterization of the nonlinear manifold where system dynamics evolve; (2) projection between the original state space and the manifold subspace. To the best of our knowledge, previous literatures have not brought into attention the concept of separating the construction of subspace and projection into the subspace, although this is implicitly done in linear-projection-based methods (where the projector defines the low-order subspace). This separation gives a better understanding of MOR methods.

The third main contribution of this paper is that we present a procedure to identify the nonlinear manifold, define the manifold coordinate system, and generate a compact model for fast simulation. The procedure identifies the nonlinear manifold and the coordinate system in a correct-by-construction manner – *i.e.*, we ensure that the major system responses do lie on the nonlinear manifold we construct. Since the manifold is nonlinear, it is approximated by stitching together the local tangent subspaces around a set of sample points on the manifold, and the coordinate system is built along *nonlinear axes*. Accordingly, the nonlinear projection/mapping function is also defined by a piece-wise linear approximation around several nearest sample points. The generated compact model enables us to convert function and Jacobian computations into efficient matrix-vector products, by employing a piece-wise linear approximation. This is similar but distinguishable to the technique used in previous trajectory-based approaches.

The rest of the paper is organized as follows. In Section II, we briefly review the projection framework of existing MOR methods, and the trajectory-based methods. In Section III, we discuss in detail the procedure of *maniMOR*, together with some implementation details. Three examples of application of *maniMOR* are presented in Section IV. Finally, Section V concludes the paper, and proposes future research directions.

978-1-4244-2820-5/08/$25.00 ©2008 IEEE

II. Background

In this section, we first review the general MOR framework based on linear projection of the full state space into a low-order linear subspace, and the trajectory-based methods. Then we discuss problems/limitations in those methods, to motivate ideas in *maniMOR*, which are presented in Section III.

A. Projection Framework

Consider a system described by differential equations (1), where $\vec{x} \in \mathbb{R}^n$ are state variables[1], and $\vec{u}(t)$ are the inputs. For linear systems, $q(\vec{x}) = C\vec{x}$, $f(\vec{x}) = G\vec{x}$.

$$\frac{d}{dt} q(\vec{x}) + f(\vec{x}) + B\vec{u}(t) = 0 \tag{1}$$

Most previous methods boil down to identifying a low-order linear subspace of size q ($q \ll n$), which is defined by q bases $V = [\vec{v}_1 \vec{v}_2 \cdots \vec{v}_q]$, such that the main system response lie on this linear subspace. After the projection, the state variable in the low-order subspace $\vec{z} \in \mathbb{R}^q$ satisfies equation $\vec{z} = V^T \vec{x}$. The differential equations for the reduced system, shown in (2), are obtained by projecting the differential equations into another linear subspace spanned by columns of $W = [\vec{w}_1 \vec{w}_2 \cdots \vec{w}_q]$, so that the residual, defined by $\vec{r} \equiv \frac{d}{dt} q(V\vec{z}) + f(V\vec{z}) + B\vec{u}(t)$, is orthogonal to this linear subspace, *i.e.*, $W^T \vec{r} = 0$. [9]

$$W^T \left(\frac{d}{dt} q(V\vec{z}) + f(V\vec{z}) + B\vec{u}(t) \right) = 0 \tag{2}$$

Different projection-based methods differ by using different projection matrices V and W, and in many cases, $V = W$ is used. Without loss of generality, we assume $V = W$ throughout this paper.

The projection-based methods are efficient for linear systems since the linear function, $f(\vec{x}) = G\vec{x}$, can also be written in the form of matrix-vector products in the reduced system, *i.e.*, $\hat{f}(\vec{z}) = \hat{G}\vec{z}$, where $\hat{G} = V^T G V$.[2] This avoids projections during the simulation, and reduces computational cost.

B. Trajectory-based Methods

However, for nonlinear functions, there is no easy way to express $\hat{f}(\vec{z}) = V^T (f(V\vec{z}))$. Trajectory-based methods come into play in the sense that they express the nonlinear function in a piece-wise linear (polynomial) fashion, so that compact model can be generated to achieve computation reduction.

In trajectory-based methods, as shown in (3), the nonlinear function is first linearized at several sample points $\vec{x}_i, (i = 1, 2, \cdots)$, which are sampled along the trajectories corresponding to a set of "training" inputs. This enables the compact model for each linearized system to be generated in the same way as in linear systems, as shown in (4).

$$f(\vec{x}) = f(\vec{x}_i) + G_i(\vec{x} - \vec{x}_i) \tag{3}$$

$$\begin{aligned} \hat{f}(\vec{z}) &= \hat{f}(\vec{z}_i) + \hat{G}_i(\vec{z} - \vec{z}_i) \\ &= V^T f(V\vec{z}_i) + V^T G_i V(\vec{z} - \vec{z}_i) \end{aligned} \tag{4}$$

Finally, the nonlinear function is approximated by a weighted summation (interpolation) of compact models for all linearized systems, as shown in (5). [3]

$$\hat{f}(\vec{z}) = \sum_i w_i(\vec{z}) \left(\hat{f}(\vec{z}_i) + \hat{G}(\vec{z} - \vec{z}_i) \right) \tag{5}$$

[1]Specifically, $\vec{x}$ represent node voltages and branch currents in circuit differential algebraic equations.

[2]The same operations are done for function $q(\vec{x})$. Therefore, in the rest of the paper, we provide derivations only in terms of $f(\vec{x})$.

[3]Several improvements on robust and efficient function computation have been proposed by several authors, such as kernel methods [14], interpolation among k-nearest neighbors [11], [12], *etc.*.

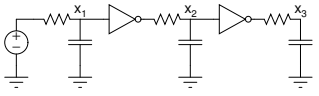
Since the low-order subspace should include the low-order subspaces for all the linearized models, the projection matrix V in trajectory-based methods is generated by (1) calculating the projection matrix V_i for each linearized system; (2) aggregating all the projection matrices $V_{agg} = \text{Union}(V_i)(i = 1, 2, \cdots)$; (3) performing an SVD on the aggregated projection matrix $V = \text{SVD}(V_{agg})$; (4) picking up the q column vectors in V that correspond to q largest singular values. The key idea behind this procedure is that if the low-order subspace for each linearized system can capture system responses around the linearization point, then the union of all those low-order subspaces should capture all system responses.

C. Limitations of Linear-projection-based Trajectory-based Methods

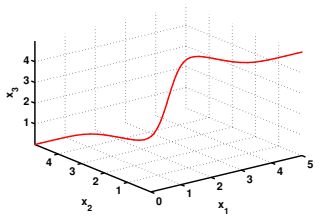
While the trajectory-based methods show reasonable reduction of the system and good match to the full model, they also cast a variety of questions. Several authors have paid attention to several aspects of the method [11]–[14], and we just list the problems that have not been noticed/published/well-solved.

Firstly, and maybe the most importantly, the DC operating points are modeled in trajectory-based methods by a simple heuristic, which can be problematic in practice. It includes the state variables, which are sampled from the training trajectories, into the projection matrix. This is exactly to force those states to be embedded in the low-order subspace. There are three main problems with this heuristic:

- For the sample states that *are not* DC operating points, they do not need to be added into the projection matrix, because they could travel in the full space, and not essentially in the low-order subspace we want to construct. Even for a LTI system, the trajectory could go off the reduced subspace, such as its low-order Krylov-subspace. So incorporating those states into the low-order subspace is unreasonable, and will lead to a much larger model.
- For the sample states that *are* DC operating points, the heuristic is good since it exactly matches the DC response. However, it is not perfect, and sometimes bad, since the DC operating points could potentially be anywhere in the state space. An illustrative example is a chain of two inverters, as shown in Fig. 1. Fig. 1(b) shows the DC operating points plotted in the state space – merely DC operating points span the full state space. Using this heuristic, no reduction can be obtained in this particular example. Another obvious example is an N-bit ADC. Since the N outputs need to span at least an N-D linear subspace, we could at most reduce the system size to N, while only capture DC response. Therefore, linear projections can lead to a large model.



(a) Circuit diagram.



(b) DC operating points.

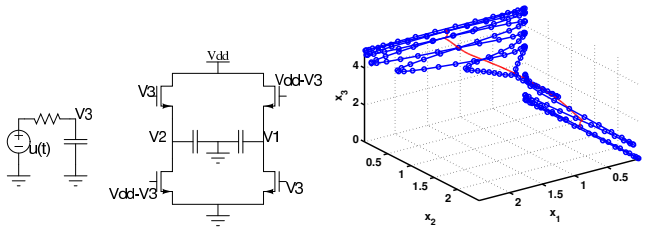
Fig. 1. A chain of two inverters.

Secondly, the generation of the final projection matrix is questionable. In the trajectory-based methods, the final projection matrix is generated by aggregating the projection matrices for all linearized systems, followed by an SVD, and selection of column vectors corresponding to singular values larger than some threshold ε. Two arguments can be made to this heuristic:

- The aggregation leads to large model. Consider the circuit of a ring mixer shown in Fig. 2(a). At different DC bias, different

transistors in the circuit are on/off. So the local subspaces at different DC operating points do not share a common linear subspace. This can be observed in Fig. 2(b), which plots a trajectory obtained from a transient simulation of the circuit. In Fig. 2(b), locally around a DC operating point, the trajectory is confined in a low-order subspace (plane in this case). However, the size of the common linear subspace is larger than each local low-order subspace.

- The aggregation could kill important bases generated for linearized systems. The first few bases in each V_i are always important – they correspond to the first few derivatives of the transfer function (moments) in Krylov-subspace methods, or the largest few Hankel singular values in TBR methods. However, after the aggregation, this information is lost! What is worse, the less useful bases, which correspond to higher-order derivatives of the transfer function or smaller Hankel singular values, could duplicate or be highly-correlated in different linearized systems. As a result, the SVD of the aggregated projection matrix will first pick up those unimportant columns, and filter out important ones, thus losing important system characteristics. Also highly possibly, the DC operating points, characterized using the previous heuristic, can again be lost in this step. This is observed in our experiments – specifically, in the yeast pheromone pathway example, as shown in Section IV-C, almost no reduction can be obtained. We also demonstrate later in Section IV-C another heuristic trying to get around this problem, but the model generated is still not efficient.



(a) Circuit diagram. (b) Trajectory in state space.

Fig. 2. A simple CMOS ring mixer. The red curve is the DC operating points. The blue trajectory is from a transient simulation.

Thirdly, one recent big improvement of trajectory-based methods, based on localized linear reductions [13] was proposed. The local linear subspaces are identified beforehand for clusters of points. This approach could outperform far more better than previous variants. However, still some problems need to be addressed carefully.

- Since all the computations are computed locally, the algorithm does not have a global view of where the state variable is unless it is projected back to the aggregated subspace. Therefore, although the model evaluations are performed in the local smaller reduced subspace, the equation solver is applied in the common aggregated subspace. Otherwise, the local reduced subspace must be chosen more conservatively such that two consecutive states (in transient analysis, for example) fall in the same local subspace.
- The search for the nearest neighbor is performed in the aggregated subspace, whose size is much larger than local subspaces. The computational cost for evaluating pair-wise distance metric and searching the nearest neighbors might become comparable to function computations.
- Generally, it also suffers from some common problems of trajectory-based methods, as mentioned before.

In a brief summary, the linear-projection-based methods and the aggregation in trajectory-based methods generally lead to redundant models, and trajectory-based methods intrinsically suffer from some fragility. We try to address those problems in *maniMOR* model in section Section III.

III. maniMOR

In this section, we present our *maniMOR* model order reduction procedure. We start by introducing the notion of separating the subspace construction and projecting operation in MOR procedure. Under this framework, we give a high-level algorithm description of *maniMOR*, followed by detailed explanations and derivations in several sub-sections.

The key difference of *maniMOR* algorithm from existing algorithms is that *maniMOR* constructs a low-order nonlinear manifold subspace and uses nonlinear projection to project the full state space into this manifold subspace, while previously almost all methods are based on low-order linear subspace and linear projection operation. As we will demonstrate, the low-order nonlinear manifold we build does remove many redundancies that are present in its linear counterpart, as well as capturing important system responses. We also show that the the nonlinear projection could be performed in a piece-wise linear manner, which enables fast computation of the model. Equally important is that the pre-computation and storage of the *maniMOR* model does not grow too high – a compact nonlinear macromodel will finally be generated.

A. Separation of Subspace Construction and Projection

As discussed in Section II, previous MOR methods try to find a projection matrix V, whose columns span a low-order subspace, and define the projection operation $\vec{z} = V^T\vec{x}$. Implicitly in this equation, the construction of the low-order subspace and projection operation are mixed – both of them are are defined by V.

By definition, the low-order subspace is the subspace that captures main system responses. The projection is generally defined by the point in the subspace that is closest to the point in the original space. For example, in Krylov subspace method, the bases of the subspace are chosen such that the first few moments of the transfer function are matched. Then the projection operation coincides with the left multiplication of V^T. In retrospect, we find that existing MOR methods do fit into this concept of separation. Table I lists the subspace construction schemes and projection methods used in Krylov-subspace, TBR and trajectory-based methods.

The separation of the two procedures helps us better understand MOR methods, and provides us more freedom and insights to do model order reduction – they do not need to be defined by the linear projection matrix. Next, we presents an attempt to construct a nonlinear reduced-order manifold and to employ nonlinear projections in the *maniMOR* model.

B. maniMOR Outline

The outline of *maniMOR* macromodel generation is shown below, followed by several subsections discussing the details.

1) Trajectory sampling
   - For different training inputs, perform simulations of the full system, such as DC, TRAN, HB, *etc.*.
   - Sample a few points on those trajectories, using certain criterion. (*e.g.*, assign a minimum distance between any two samples.)
2) Manifold construction and coordinate assignment
   - Construct the nonlinear manifold in a correct-by-construction manner, *i.e.*, try to match the system responses observed in training trajectories.
   - Embed the low-order nonlinear manifold in a nonlinear coordinate system.
3) Reduced-order model generation
   - For each sampled point on training trajectories, find its projection in the manifold coordinate.
   - For each projected sample point, calculate the low-order subspace of the system linearized at that point. (*e.g.*, the Krylov-subspace)

TABLE I

SEPARATION OF SUBSPACE CONSTRUCTION AND PROJECTION

Methods	Subspace construction guideline	Projection
Krylov-subspace	matches first few derivatives of the transfer function	$\vec{x} = V\vec{z}$
TBR	matches largest few Hankel singular values	$\vec{x} = V\vec{z}$
Trajectory-based	the union of subspaces at all sampled points	$\vec{x} = V\vec{z}$

- Precompute the functions, Jacobians, local projection matrices at all samples, and store the model, which is essentially a look-up table.

C. Construction of the Nonlinear Manifold and Its Coordinate System: From Linear Subspace to Nonlinear Manifold

Intuitively motivated in Section II-C and Fig. 1(b), we see that a nonlinear manifold contains more information than linear subspaces, and therefore has the potential ability to exploit more reduction of the model. Here we present the detailed procedure to construct such a nonlinear manifold.

1) The first dimension: The first intuition to construct our nonlinear manifold is motivated by the inverter example we mentioned in Section II-C, *i.e.*, a 1-D (size 1) model should be enough to capture the DC response of the system. In the linear case, the DC operating points constitutes a straight line going through the origin in the state space, which is defined by $G\vec{x}+Bu=0$. For a nonlinear system, the DC operating points are x's that satisfy $f(\vec{x})+Bu=0$, which represent a curve in the state space. This curve can be easily computed by performing a DC sweep simulation.

Now that we have obtained a "DC curve", we regard it as a nonlinear axis. Suppose the state variable in the reduced space is $\vec{z}=[z_1,z_2,\cdots,z_q]^T$, we designate the coordinates of z_1 along this "DC curve". Doing so, we ensure the DC response of the full system be exactly matched in the reduced-order model. This simple coordinate designation scheme is a natural generalization of the linear case to the nonlinear case. In linear-projection methods, z_1 is along the direction of $\vec{v}_1$, the first column of the projection matrix V. Specifically, in Krylov-subspace methods, $\vec{v}_1 = G^{-1}B$, which exactly matches the equation for DC operating points $G\vec{x}+Bu=0$.

One important question is that what should the value z_1 be at different points on the "DC curve". We formulate the procedure as follows: (1) arbitrarily choose a DC operating point, and let z_1 for this point to be 0; (2) choose the point $\vec{x}_j$ that is closest to points that have been designated a coordinate, among which the closest point to $\vec{x}_j$ is $\vec{x}_i$, choose its z_1 value z_{1j} such that (6) is satisfied, where the sign is determined by their relative position along the "DC curve".

$$||z_{1j}-z_{1i}||_2 = ||\vec{x}_j-\vec{x}_i||_2 \tag{6}$$

Since locally a curve can be approximated by a straight line, the distance between two nearby points *along the curve* could reasonably approximated by the distance between the two points. We claim this procedure is valid because it preserves the distance information between nearby points – in other words, it preserves the *geodesic* distance information between pairs of points.

2) Beyond the first dimension: The second thing to be captured in the nonlinear manifold is the small signal response at each DC operating point. Since locally around each DC operating point, the low-order subspace is linear, and this linear subspace can be efficiently calculated, (*e.g.*, Krylov-subspace methods), we approximate the nonlinear manifold by stitching the low-order linear subspace around each DC operating point together. Thus, all the small signal responses are captured exactly in the model.

In this case, all the axes except for the first one are linear, and the coordinates are trivially chosen, just as the linear case.

This idea could be examined in view of model order reduction for time-varying systems. Suppose the input of a time varying system is decomposed into a large signal $u_L(t)$ and a small signal $u_S(t)$, *i.e.*, $u(t)=u_L(t)+u_S(t)$. Each value of u_L corresponds to an equilibrium point and a local low-order linear subspace. Suppose q is size of the reduced model, and Krylov-subspace methods are used for each linearized system. The first q moments of the transfer function are matched. In trajectory-based methods, all the q-D linear subspaces at different equilibrium points are aggregated together. Therefore they render a more redundant model than *maniMOR*.

The last thing to be included in the manifold is the large signal response. It can naturally be extrapolated from the previous two ideas that we need to make the rest $q-1$ axes *nonlinear*. Obviously, this is a hard problem since there is no easy simulation such as the DC sweep simulation that could give us the manifold.

Based on the fact that at each point on the nonlinear manifold, the tangent linear subspace is a good approximation to the local manifold, we could say that *a point on this tangent linear subspace and close to the expansion point is also on the manifold.* Therefore, starting at any DC operating point x_0, we first calculate its Krylov subspace, and the corresponding projection matrix $V_0=[v_1,v_2,\cdots,v_q]$. Then, we can go an Euler step along any of the $q-1$ directions from x_0, and claim that the new point is also on the manifold.

For example, the simplest way is the forward Euler method in (7), where x_{i-1} is the known point on the manifold, x_i is the new point, $\vec{v}_j$ is the j-th Krylov basis, h is the step size.

$$x_i = x_{i-1} + h\vec{v}_j \tag{7}$$

Accordingly, the z coordinate is chosen by increasing the corresponding z element by h. For example, if a size h step is performed along v_j direction, z_j is increased by h. When the Euler step is small enough, our assumption that the new point does lie on the nonlinear manifold is correct.[4]

D. Projection Between the Nonlinear Manifold and the Full State Space

Similar to the linear-projection-based methods [10], there are two projection operations: projection of the state space $\vec{x}=f_V(\vec{z})$ and the projection of the differential equations $W^T(\vec{z})$. After the projection, the differential equations in the reduced-order subspace is (8).

$$W^T(\vec{z})\left[\frac{d}{dt}\vec{q}(f_V(\vec{z}))+\vec{f}(f_V(\vec{z}))+B\vec{u}(t)\right]=0 \tag{8}$$

1) Nonlinear projection of the state space: We express the nonlinear projection between $\vec{x}\in\mathbb{R}^n$ and $\vec{z}\in\mathbb{R}^q$ by a nonlinear equation $\vec{x}=f_V(\vec{z})$. However, since there is no particular form (*e.g.*, polynomial) for this nonlinear equation, we use a piece-wise linear function as an approximation.

At any sample point $\vec{z}_i$, we write the first-order Taylor expansion of $f_V(\cdot)$ in (9). In (9), $f_V(\vec{z}_i)$ is exactly $\vec{x}_i$, and the term $\frac{\partial f_V}{\partial \vec{z}}$ is essentially the projection matrix V_i of the linearized system at $\vec{x}_i$, which can be efficiently computed (*e.g.*, by Krylov subspace methods). So (9) can be simplified to (10).

$$\vec{x}=f_V(\vec{z})\simeq f_V(\vec{z}_i)+\frac{\partial f_V}{\partial \vec{z}}(\vec{z}-\vec{z}_i) \tag{9}$$

$$\vec{x}\simeq \vec{x}_i+V_i(\vec{z}-\vec{z}_i) \tag{10}$$

[4]The problems are also obvious in this heuristic: (1) the number of points needed to be visited on the manifold could increase exponentially; (2) there is no theoretical proof that such a nonlinear manifold could capture large signal responses. We have been trying some other heuristics combined with this one, and some of them do give good experimental results. Though, we seek more theoretically sound manifolds to be used here.

So, the nonlinear function can be approximated by (11), a weighted summation of the functions linearized at different samples, where the weight function $w_i(\vec{z})$ represents certain distance metric (kernel) between $\vec{z}$ and $\vec{z}_i$, and sum up to unity, *i.e.*, $\sum_i w_i(\vec{z}) = 1$. The parameter k in (11) is used to select k nearest samples around $\vec{z}$, so that the computation does not increase with the number of samples. Similarly, the projection from $\vec{z}$ to $\vec{x}$ is derived in (12).

$$\vec{x} \simeq \sum_{i=1}^{k} w_i(\vec{z}) \left(\vec{x}_i + V_i(\vec{z} - \vec{z}_i)\right) \tag{11}$$

$$\vec{z} \simeq \sum_{i=1}^{k} w_i(\vec{x}) \left(\vec{z}_i + V_i^T (\vec{x} - \vec{x}_i)\right) \tag{12}$$

2) Nonlinear projection of the differential equations: Generally we could define any function of the projection matrix $W(\vec{z})$ which depends on $\vec{z}$ (or $\vec{x}$). For simplicity, we could just use the term $\frac{\partial \vec{x}}{\partial \vec{z}}$ as the projection matrix, as shown in (13).[5] Since the second term in (13) is always small, we could reasonably remove that term for computational speedups.

$$\begin{aligned} W(\vec{z}) \simeq \frac{\partial \vec{x}}{\partial \vec{z}} &= \sum_{i=1}^{k} w_i(\vec{z}) V_i + \sum_{i=1}^{k} \frac{\partial w_i(\vec{z})}{\partial \vec{z}} \left(\vec{x}_i + V_i(\vec{z} - \vec{z}_i)\right) \\ &\simeq \sum_{i=1}^{k} w_i(\vec{z}) V_i \end{aligned} \tag{13}$$

We stress that (11) and (12) represent *nonlinear projections* between the original state $\vec{x}$ and reduced state $\vec{z}$, and (13) represents the nonlinear projection of the residual/differential equations. In comparison, previous works such as [10], use a linear projection $\vec{z} = V^T \vec{x}$ for both the state space and residual. This important difference enables the nonlinear manifold concept, and leads to a different model generation scheme, as described in Section III-E.

E. Compact Model Generation

From previous two subsections, we see that the nonlinear function in the reduced manifold subspace is in the form of $W^T(\vec{z}) f(f_V(\vec{z}))$. Given a state variable $\vec{z}$, the naive way to compute this function is to project $\vec{z}$ to $\vec{x}$, evaluate the function $f(\vec{x})$, and project back to $f(\vec{z})$. However, this leads to prohibitively expensive computational cost, such that the reduction does not bring speed-ups.

As a recipe for that, we seek to use simple matrix-vector product computations to calculate $W^T(\vec{z}) f(f_V(\vec{z}))$ directly in the nonlinear manifold subspace. Similar to what is mentioned in trajectory-based methods [10], $f(\vec{x})$ is approximated by the weighted summation of linearized functions at different samples $\vec{x}_i$, as shown in (14).

$$f(f_V(\vec{z})) = f(\vec{x}) \simeq \sum_i w_i(\vec{z}) (f(\vec{x}_i) + G_i V_i(\vec{z} - \vec{z}_i)) \tag{14}$$

Therefore, multiplying (13) with (14), we obtain (15).

$$\begin{aligned} f(z) &= W^T(\vec{z}) f(f_V(\vec{z})) \\ &= \sum_{j=1}^{k} w_j(\vec{z}) V_j^T \sum_{i=1}^{k} w_i(\vec{z}) (f(\vec{x}_i) + G_i V_i(\vec{z} - \vec{z}_i)) \\ &= \sum_{j=1}^{k} \sum_{i=1}^{k} w_j(\vec{z}) w_i(\vec{z}) (V_j^T f(\vec{x}_i) + V_j^T G_i V_i(\vec{z} - \vec{z}_i)) \end{aligned} \tag{15}$$

Similar equations can be derived for $q(\vec{z})$ and $B(\vec{z})$. Finally, for each pair i, j, we store the matrices/vectors in the macromodel, as shown in table Table II. This means that if we take m samples to construct the manifold, $O(m^2)$ matrices/vectors are required for storage. However, it is worthwhile to mention that in practice we do not need to generate a size $O(m^2)$ LUT, because a pair of two points far away from each other will never be used for function evaluation. Thus we can eliminate many such pairs. Hence the actual size of the LUT is of the order of m. For example, if we only store pairs which are in the k-nearest neighbor, k times of the storage in TPWL method is required in *maniMOR*. Although the total storage is larger than previous trajectory-based methods, it grows gracefully, and is affordable in practice.

TABLE II
LOOK-UP TABLE STORED IN THE REDUCED MODEL

Variable	Meaning	Size
$\hat{f}_{ij}$	$V_j^T f(\vec{x}_i)$	q x 1
$\hat{q}_{ij}$	$V_j^T q(\vec{x}_i)$	q x 1
$\hat{B}_i$	$V_i^T B$	q x 1
$\hat{G}_{ij}$	$V_j^T G_i V_i$	q x q
$\hat{C}_{ij}$	$V_j^T C_i V_i$	q x q

F. Simulation of maniMOR model

We discuss two most common simulations, DC and TRAN, of the *maniMOR* model. Similar procedures can be adapted to other simulations, such as harmonic balance simulation.

1) DC simulation: As mentioned before, the DC analysis could be performed directly using the 1-D model. Doing so, not only huge speedups is obtained, but also can we avoid some DC convergence problems, as a side benefit. Since *maniMOR* has trained all the possible equilibrium points, it also automatically eliminates possible multiple unrealistic/non-physical DC solutions. Continuation methods [15], which is well known to deliver global DC convergence, is also easier to apply because of the fact that we are only search in a 1-D subspace.

2) Transient simulation: The transient simulation of the *maniMOR* model is summarized below.

- Project the initial condition $\vec{x}_0$ into the nonlinear manifold subspace, and get $\vec{z}_0$ as the new initial condition of reduced system;
- Search the k nearest points around $\vec{z}_0$, among sampled points stored in the LUT in *maniMOR*;
- Compute the function values using equation (15), feed into any nonlinear solver to solve the transient integration equations, *e.g.*, backward Euler equations;
- Repeat the above two steps until time $t = t_{\text{end}}$;
- Project the solutions back to the full state space.

IV. VALIDATION

In this section, we apply *maniMOR* method to macromodel a nonlinear transmission line circuit, an I/O buffer circuit, and the yeast pheromone signaling pathway system. We validate the *maniMOR* model by comparing the results of simulations using *maniMOR* model and the full model. We also show comparisons of *maniMOR* to existing methods, mainly against trajectory-based methods, to gauge the accuracy and efficiency of *maniMOR*. All simulations were performed using a MATLAB/C/C++-based circuit/system simulation environment, on a 2.4GHz Athlon XP-based PC running Linux.

A. Nonlinear Transmission Line

An early example in TPWL papers is the nonlinear transmission line circuit [16], [17]. To verify *maniMOR*, we apply both *maniMOR* and TPWL on the nonlinear transmission line circuit with 100 nodes.

Using the same training set and model size in [16] (*i.e.*, TPWL model is of order 10; training input is the step input $i(t) = u(t-3)$), we apply a test input of $i(t) = 0.8u(t-3)$ to see how TPWL model behaves. As shown in Fig. 3, since the DC solution corresponding

[5]This approximation is valid because when x is near x_i, w_i is nearly 1, and it matches the projection matrix at x_i. However, it is noticed that when x is not near any sample point, this approximation will render a projection matrix whose columns are not orthonormal. So, some modifications should be applied to handle this problem. But our current implementation uses (13).

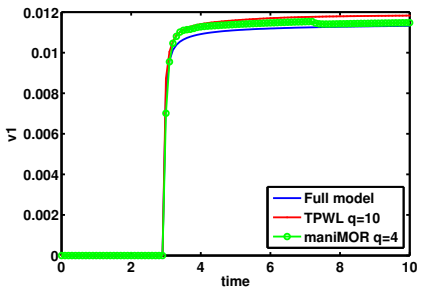


Fig. 3. Comparison of of the transient simulation of *maniMOR* model and TPWL model. (Nonlinear transmission line circuit [16])

to $i = 0.8$ is not trained, an observable error, compared to the full simulation, is present in the result.

Accordingly, a *maniMOR* model of order 4 is generated, and the results are plotted in Fig. 3. We can see that (1) the error is less than that of TPWL model, even if the model size is almost halved; (2) despite the initial transient error, the solution converges to the exact steady state solution finally.

B. CML Buffer

The second example is a current-mode logic (CML) buffer chain [18] (Vdd = 1.8V), the circuit diagram of which is shown in Fig. 4. We use the BSIM3 [19] model for MOSFETs through the simulation, and the size of differential algebraic equations for the full circuit is 52.

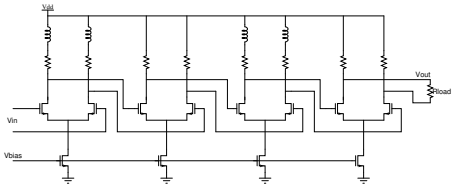


Fig. 4. Circuit diagram of a current-model logic (CML) buffer.

1) Macromodel generation: Applying *maniMOR*, a size 5 model is generated for this circuit. When generating the reduced-order model, the local Krylov-subspace projector is calculated by applying Arnoldi algorithm on the linearized system at each sample point. This means that for each DC operating point, 5 moments of the transfer function of the linearized system is matched.

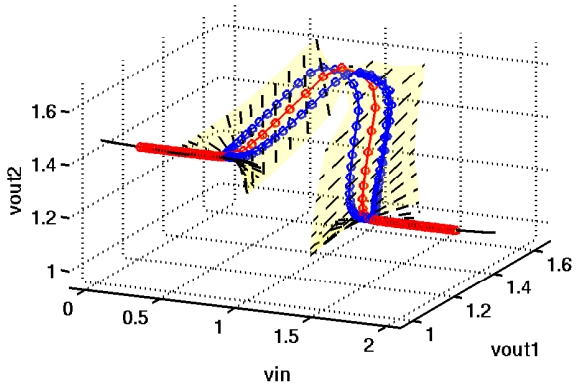


Fig. 5. Visualization of the low-order manifold subspace generated by *maniMOR*. The axes represent three node voltages in the circuit. The **red** curve consists of the DC operating points; the **black** dashed lines are the second local subspace bases; the **transparent** yellow manifold is generated by stitching together all the local linear subspaces. The **blue** circled points are on a transient trajectory simulated using the full model.

The nonlinear manifold we constructed is visualized in Fig. 5 (only first two bases are plotted) by picking three node voltages as axes. Clearly, the manifold is nonlinear, and blue circled points, which are sampled on a transient trajectory of the full model, do stay close to this nonlinear manifold, which confirms that the manifold is the low-order subspace that captures system responses.

In contrast, TPWL [10] aggregates all the samples and all projection matrices into the final projection matrix. We used 20 DC operating points for training – by aggregating those state variables, we obtain a matrix of 20 columns. The singular values of this matrix are plotted in Fig. 6. Therefore, to get a set of bases which approximate all the DC operating points with an overall accuracy of 0.01, first 8 bases must be used! Only to capture the DC response of the circuit, TPWL has to use a model of size at least 8.

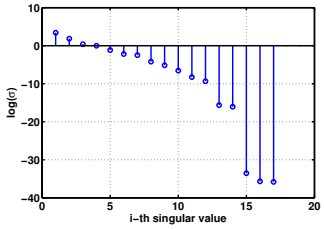
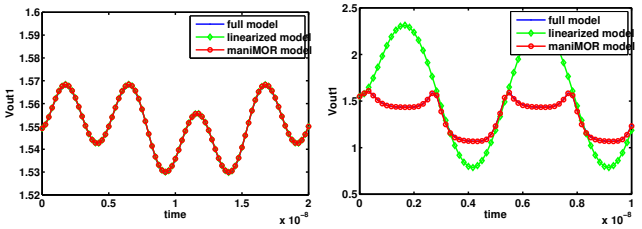


Fig. 6: Singular values of the matrix containing 20 equilibrium points.

When generating the TPWL model, we use the same method (*i.e.*, Arnoldi algorithm), the same linear subspace size (*i.e.*, 5), to calculate the local linear projector as in *maniMOR*. As a result, a size 12 TPWL model is generated in order to match all the DC operating points, and the first 5 moments of the transfer functions for all the linearized systems.

2) Transient simulation with small signal inputs: We first test the case when the input is a small signal superimposed on a DC bias. A two tone input $u(t) = 0.005\sin(2\pi 0.5\times 10^9 t) + 0.0025\sin(2/3\pi 0.5\times 10^9 t)$ is applied. Not surprisingly, since the amplitude of the input is small, the transient trajectory is always near the DC operating point, and the reduced-order model of the system linearized at the DC operating point captures the small signal response quite well, as shown in Fig. 7(a). The result of the *maniMOR* model also matches that of the full model – when inspecting the simulation of the model, the local region chosen at each transient step is always the DC operating point.



(a) Small signal input. (b) Large signal input.

Fig. 7. Comparison of *maniMOR* model with linearized reduced-order model. The waveforms of one output voltage using different models are plotted.

3) Transient simulation with large signal inputs: When the input includes a large signal, the trajectory will not be only around a DC operating point, but travel among different regions. So the small signal model, which works fine in the previous experiment, fails to match the waveforms of the full model.

In this experiment, we first apply a sinusoidal input $u(t) = 0.25\sin(2\pi 0.5\times 10^9 t)$. As shown in Fig. 7(b), there is some distortion happening in the output waveform (The waveform of the full model (blue) is almost overlapped with that of *maniMOR* model (red)). The small signal model, because it takes the linearity assumption, still generates a sinusoidal output waveform, and fails to capture the distortion effect. Though, *maniMOR* model matches the true waveform, since it uses different local models when the state variable travels to different regions in the state space.

We then apply a step input as shown in Fig. 8. We manually choose the input so that the output voltage will traverse four DC operating points.

As shown in Fig. 9, without the heuristic to handle the equilibrium points, TPWL model with model size 10, fails to converge back to the correct DC operating points. After including the DC operating

points into the projection matrix, The TPWL model of size 10 still leads to some error. During experiments, we find that the model size must be at least 12 in order to match the original trajectory. And this result is obtained by using a training input to be the same as the test input – potentially an even larger model will be needed.

In comparison, *maniMOR* model with size 5 renders an output waveform almost indistinguishable from that of the full model – less than half of the size of the TPWL model to reach the same accuracy. We conclude that the redundancy in TPWL model is because of the "aggregation" and SVD step. *The aggregation and SVD step has to embed all the DC operating points and all the local Krylov subspaces into a linear subspace.* Under the linear subspace constraint, it is very possible that much less reduction can be obtained.

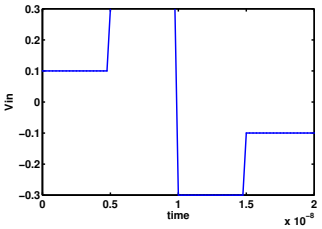


Fig. 8: Input signal.

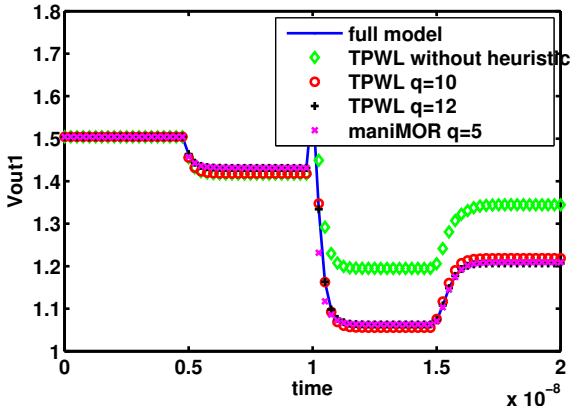


Fig. 9. Comparison of *maniMOR* model with TPWL model. The waveforms of one output voltage using different models are plotted.

C. Yeast Pheromone Signaling Pathway

In this example, we show the application of *maniMOR* macromodelling method on a Bio-chemical system. We have found that *maniMOR* model reduces the system size by a large amount for this example, and works far more better than TPWL model. We provide the simulation results of *maniMOR* and TPWL model, and explain why TPWL model fail to achieve any reduction in this example.

$$\frac{d}{dt}\vec{x} = f(\vec{x},\vec{k},u(t)) \tag{16}$$

The chemical kinetics are modeled accurately by a set of differential equations, which are in the form of (16). In (16), $\vec{x}$ are the concentrations of all reactants in the system, $\vec{k}$ are constants representing the reaction rates, $u(t)$ is a time-varying input to the system, which is usually controllable in experiments. The function $f(\vec{x},\vec{k})$ usually contains terms such as $k_{ij}x_i^a x_j^b \quad (i,j,a,b \in \mathbb{N})$. As a result, chemical reactions always show a lot of *nonlinearities*.

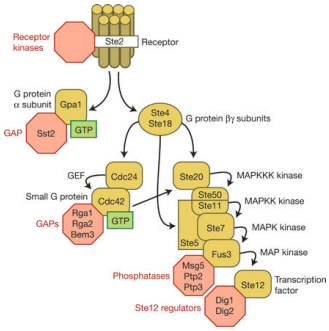
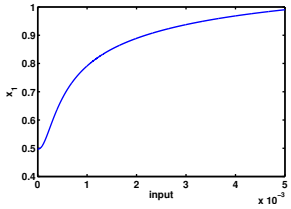


Fig. 10: Components of the pheromone response pathway in yeast. [20] Each arrow represents a type of reaction.

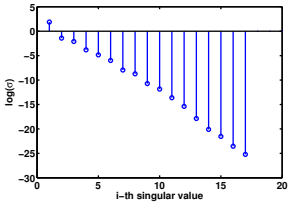
In a chemical reaction, the number of reactants is usually huge. For example, yeast pheromone signaling pathway [20] contains a chain of chemical reactions, as shown in Fig. 10. For each reaction, several reactants participate. Shown in [21], [22], the differential equation model models 304 reactants in this pathway. But bio-chemical engineers almost only care how input changes can affect certain output of the system, and this is where MOR can play an important role. Reduced-order models also makes it possible to simulate "very large scale" bio-chemical systems.

1) Macromodel generation: The equilibrium states in chemical systems are extremely important. Some engineers/scientists in biological domain run a very long time transient simulation to get the equilibrium state. Although it is a smart idea to suggest performing a "DC analysis" on the chemical systems, we find that it is very hard to make the DC analysis converge, unless continuation/homotopy/limiting methods, which are computationally expensive, are applied. We also find that even if the "DC analysis" converges, since there are a lot of DC solutions to those equations, most of which unrealistic (negative concentration makes no sense in reality), we have difficulty to identify the real solution.

In the training course of *maniMOR* model, we carefully simulate the DC operating points due to a wide range of input values, as plotted in Fig. 11(a). The *maniMOR* model then captures all those equilibrium points exactly, based only on a 1-D model! In the process of building the *maniMOR* model, we also implicitly eliminates all the useless equilibriums, which makes the DC convergence of the macromodel much better than the full model.



(a) Concentrations of x_1.



(b) Singular values.

Fig. 11. Concentrations of x_1 at equilibrium corresponding to different inputs, and the singular values for the DC matrix.

In contrast, TPWL model has to generate a matrix containing all the equilibrium points. We picked up 20 equilibrium points into a matrix. The singular values of this matrix is again plotted in Fig. 11(b). Setting the threshold of the singular value to be 1×10^{-10}, we have to include 8 bases into the projection matrix, which renders at least a size 8 model. And, this only ensures the overall error among all equilibrium points – it is extremely possible that large error occurs for a certain equilibrium point.

2) DC and small signal analysis: The results are very similar to the CML buffer example, so we omit the simulation results here.

3) Transient simulation with large signal inputs: We also test the *maniMOR* model using a large signal input $u(t) = 0.003 + 0.002\sin(2\pi\frac{1}{72000}t)$. As plotted in Fig. 12, we observe a relatively good match between the *maniMOR* model (size 30) and the full model. Even with a size 3 model, the error is quite small.

In contrast, TPWL method is quite inefficient in this case. A plain implementation of TPWL (including DC operating points, and the local subspace size is 6, set the training input to be the test input) renders a model of size 300 (almost the same as the full model). Even though, it cannot match correct system dynamics. Our explanation is that it results from the problems brought by aggregation of projection matrices – trivial bases kill the non-trivial ones. By

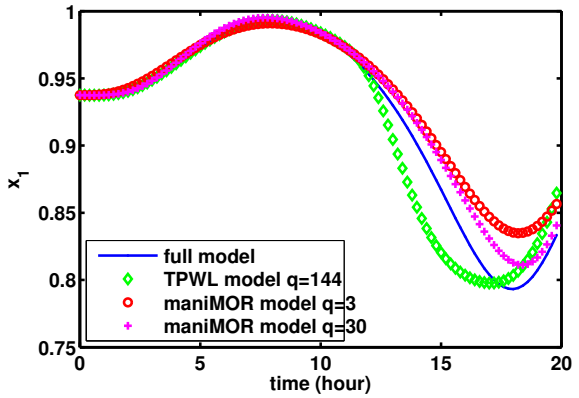


Fig. 12. Transient simulation of the yeast pheromone pathway. The concentration of one reactant is plotted.

employing a heuristic,[6] TPWL method works for this example, with order reduction from 304 to 144. But even with model size 144, the simulation results of TPWL model, as shown in Fig. 12, is not better than *maniMOR* model with $q = 3$.

Obviously, *maniMOR* model out-performs TPWL model in this example. We also see that the drawbacks in TPWL method, as discussed in Section II, can sometimes be significant in practice, and those problems are very hard to be tackled if the *common aggregated linear subspace* assumption is applied on the reduction procedure.

V. Conclusion

In this paper, we have presented *maniMOR*, a nonlinear model reduction approach via nonlinear projection framework. We believe that the framework introduced here is a natural extrapolation of a sequence of model reduction formulations proposed by various authors, starting from methods for LTI systems, through methods for time-varying weakly-nonlinear systems, to methods for highly-nonlinear systems. In this light, *maniMOR* uses the nonlinear manifold and nonlinear projection to integrate together the advantages of many methods – local tangent subspace is constructed using Krylov-subspace methods; the "DC curve" in the manifold construction is similar to methods for time-varying systems; the compact model is stored in a similar way to trajectory-based methods.

We have applied *maniMOR* to three practical systems. Through the three examples, we have shown several drawbacks of previous methods, and demonstrate how *maniMOR* attacks those problems and achieves more reduction.

Obviously, many questions may arise with regard to many details of our algorithm. For example, is there any error bound estimation of this model? How to make sure that there is a unique projection of a state into the manifold subspace? We have been working on those problems, and have some preliminary results, which are not included in this paper because of the page limit. With this paper, we hope to motivate further researches in the model order reduction community.

Acknowledgments

The authors would like to acknowledge Anirvan Sengupta for providing the fundamental insight about arrows sticking out of a 1-D trajectory. The authors would also like to acknowledge Ty Thomson for allowing the access to his unpublished yeast pheromone model.

Computational and infrastructural resources from the Digital Technology Center and the Supercomputing Institute of the University of Minnesota are gratefully acknowledged.

[6]Since the first few bases in the local projection matrix V_i are more important than others, we construct $\hat{V}_k$, whose columns are the aggregation of the k-th bases from all V_i at different samples. Now we do an SVD to each V_k, and pick up bases corresponding to singular values that are larger than a threshold ε. For different k, the threshold changes due to their importance. After that, we simply aggregate all $\hat{V}_k$ into $\hat{V}_{agg}$, and do an SVD on $\hat{V}_{agg} = U\Lambda V^T$ just for ortho-normalization. Finally the projection matrix is set to U. Using this heuristic, we ensure that important bases will not be filtered in the aggregation and SVD step.

References

[1] http://en.wikipedia.org/wiki/Chemical_kinetics.
[2] A. Odabasioglu, M. Celik, and L.T. Pileggi. PRIMA: passive reduced-order interconnect macromodellingalgorithm. In Proceedings of the IEEE International Conference on Computer Aided Design, pages 58–65, November 1997.
[3] P. Feldmann and R.W. Freund. Circuit noise evaluation by Pade approximation basedmodel-reduction techniques. In Proceedings of the IEEE International Conference on Computer Aided Design, pages 132–138, November 1997.
[4] Joel R. Phillips, Luca Daniel, and Miguel Silveira. Guaranteed Passive Balancing Transformations for Model Order Reduction. Proceedings of the IEEE Design Automation Conference, 2002.
[5] Payam Rabiei and Massoud Pedram. Model Order Reduction of Large Circuits Using Balanced Truncation. Proc. IEEE Asia and South Pacific Design Automation Conference, 1999.
[6] M. Kamon, F. Wang, J. White. Generating Nearly Optimally Compact Models from Krylov-Subspace Based Reduced-Order Models . IEEE Transactions on Circuits and Systems, 2000.
[7] J. R. Phillips, C. P. Coelho, L. Miguel Silveira. On Generating Compact, Passive Models of Frequency-Described Systems. In 15th Symposium on Integrated Circuits and Systems, 2002.
[8] J. Roychowdhury. Reduced-order modelling of linear time-varying systems. In Proceedings of the IEEE International Conference on Computer Aided Design, November 1998.
[9] Joel R. Phillips. Projection-Based Approaches for Model Reduction of WeaklyNonlinear Time-Varying Systems. IEEE Transactions on Computer-Aided Design, 22(2):171–187, 2003.
[10] Michal Rewienski and Jacob White. A Trajectory Piecewise-Linear Approach to Model Order Reduction and Fast Simulation of Nonlinear Circuits and Micromachined Devices. IEEE Transactions on Computer-Aided Design, 22(2), February 2003.
[11] Ning Dong and Jaijeet Roychowdhury. Piecewise Polynomial Nonlinear Model Reduction. Proceedings of the IEEE Design Automation Conference, 2003.
[12] Saurabh K. Tiwary and Rob A. Rutenbar. Scalable Trajectory Methods for On-Demand Analog Macromodel Extraction. Proceedings of the IEEE Design Automation Conference, 2005.
[13] S.K. Tiwary and R.A. Rutenbar. Faster, parametric trajectory-based macromodels via localized linear reductions. In Computer-Aided Design, 2006. ICCAD '06. IEEE/ACM International Conference on, pages 876–883, Nov. 2006.
[14] J. Phillips. Analog Macromodeling Using Kernel Methods. Proceedings of the IEEE International Conference on Computer Aided Design, 2003.
[15] J. Roychowdhury and R. Melville. Delivering global DC convergence for large mixed-signal circuits via homotopy/continuation methods. IEEE Transactions on Computer-Aided Design, 25:66–78, 2006.
[16] Michal Rewienski and Jacob White. A Trajectory Piecewise-Linear Approach to Model Order Reduction and Fast Simulation of Nonlinear Circuits and Micromachined Devices. In Proceedings of the IEEE International Conference on Computer Aided Design, November 2001.
[17] Dmitry Vasilyev, Michal Rewienski, and Jacob White. A TBR-based Trajectory Piecewise-Linear Algorithm for Generating Accurate Low-order Models for Nonlinear Analog Circuits and MEMS. Proceedings of the IEEE Design Automation Conference, 2003.
[18] J. Savoj and B. Razavi. A 10-Gb/s CMOS clock and data recovery circuit with a half-rate linear phase detector. IEEE Journal of Solid-State Circuits, 36:761–768, May 2001.
[19] BSIM3 model, http://www-device.eecs.berkeley.edu/bsim3/.
[20] Y. Wang and H.G. Dohlman. Pheromone Signaling Mechanisms in Yeast: A Prototypical Sex Machine. Science, 306:1508–1509, Nov 2004.
[21] B. Kofahl and E. Klipp. Modelling the dynamics of the yeast pheromone pathway. Yeast, 21:831–850, 2004.
[22] T.M. Thomson and D. Endy. Rapid Characterization of Cellular Pathways Using Time-Varying Signals. In Sixth International Conference on Systems Biology, Oct 2005.