# EMPIRE: An Efficient and Compact Multiple-Parameterized Model Order Reduction Method *

Yiyu Shi and Lei He
Electrical Engineering Dept., University of California at Los Angeles
{yshi,lhe}@ee.ucla.edu

## ABSTRACT

*In physical design and optimization for VLSI/ULSI, parameterized model order reduction can be used to handle large design objectives. In this paper we propose an efficient yet accurate parameterized model order reduction method EMPIRE for physical design with multiple parameters. It is the first practical algorithm using implicit moment matching to handle high order moments of very large number of parameters. In addition, it can match the moments of different parameters with different accuracy according to their influence on the objective under study. Experiment results show that compared with the best existing algorithm CORE which uses explicit moment matching for the parameters, EMPIRE results in 47.8X improved accuracy at a similar runtime.*

**Categories and Subject Descriptors:** B.7.[Hardware]: – Integrated Circuits–Design Aids

**General Terms:** Algorithms, Design, Performance

**Keywords:** Reduction, Parameter, Sensitivity

## 1. INTRODUCTION

During the course of physical design and optimization for VLSI circuits and systems, it is highly useful to know how certain geometry parameters influence the circuit metrics of interest. For example, in decoupling capacitor (decap) budgeting, people are interested in the relationship between the voltage droop/bounce at the observation ports and the location/size of decaps [1]. Other examples include, but are not limited to, thermal via sizing [2], interconnection spacing in the parallel interconnect structure design [3], buffer and wire sizing in clock tree optimization, etc. In those problems, the sensitivities w.r.t the physical parameters are essential for the optimization.

The importance of the sensitivities should be credited to the strong nonlinearity between the optimization objective and the parameters. Those optimization problems are usually solved via sensitivity-based iterative linear or quadratic programming outlined as follows [1]- [2]: Once the sensitivities are known in each iteration, the nonlinear objective function $f(p_1, p_2, ..., p_n)$ can be linearized as $\sum_{i=1}^{n} \frac{\partial f}{\partial p_i} p_i$ or quadrated as $\sum_{i,j} \frac{\partial^2 f}{\partial p_i \partial p_j} p_i p_j + \sum_{i=1}^{n} \frac{\partial f}{\partial p_i} p_i$, where $\frac{\partial f}{\partial p_i}$ and $\frac{\partial^2 f}{\partial p_i \partial p_j}$ are the first order and second order sensitivities, respectively. However, if direct circuit simulation is used in each iteration to compute the sensitivities, it is inefficient and limits the capacity of handling large cases.

Two kinds of methods are usually applied for large cases. One is via hierarchical or multi-level methods [4], and the other is via parameterized model order reduction. Efficient though it is, the former method cannot guarantee the quality of the solution, especially in the presence of local optimas. Thus, parameterized model order reduction is ideal in the sense of both efficiency and quality. In fact, the moments of the parameters of design (referred to as POD moments in the rest of the paper) are exactly the sensitivities required for physical optimization. In this paper, we mainly focus on this method.

Unlike the traditional model order reduction [5,6], parameterized model order reduction not only reduces the circuits to a much smaller size, but also keeps the parameters of design and thus the sensitivity information [3, 7]. [3] extends the Arnoldi method used in [5] to match the POD moments. Nevertheless, the reduced circuit size is constrained by the number of parameters: the parameter number cannot be too large to ensure a reasonable model size after reduction, which makes it impractical for real applications to match more frequency domain moments. CORE [7] was proposed. It uses implicit moment matching for frequency domain moments and explicit moment matching for the POD moments.

However, problems still exist: First of all, when the parameter number is very large, even simply matching the first order of the POD moments can still result in a very large reduced system. Second, not all the PODs are of the same importance. The objective is usually more sensitive to some parameters than to others. This is discussed in more detail in Section 3. Accordingly, we need to more accurately match the moments of those more important PODs. However, CORE can only evenly match the moments of each POD. Finally, only matching the first or second order of POD moments are not accurate. For example, in Figure 1 we study a bus consists of 16 parallel wires (equivalent to an RC circuit with 5270 nodes), with the spacing between the first and second wire being treated as the parameter. At

*This paper is partially supported by NSF CAREER award CCR-0093273/0401682 and a UC MICRO grant sponsored by Analog Devices, Intel and Mindspeed. Address comments to lhe@ee.ucla.edu.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
*ISPD'07,* March 18–21, 2007, Austin, Texas, USA.
Copyright 2007 ACM 978-1-59593-613-4/07/0003 ...$5.00.

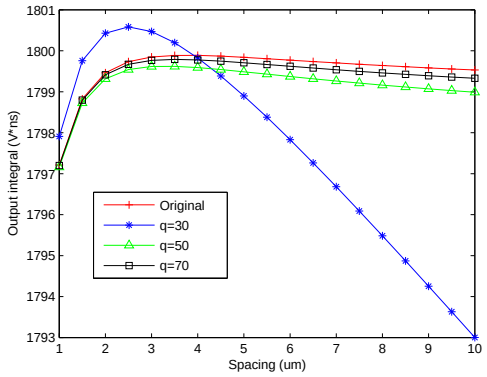


**Figure 1: The output integral w.r.t. the wire spacing at different reduced order $q$. The reduced model cannot matches the origin well when $q < 70$. This shows only matching the first or second order POD moments is far from enough.**

reduced orders q=30,50,70, a curve is presented showing the time domain output integral w.r.t. different spacing ranging from $1\mu m$ to $10\mu m$. As we can see, the reduced curve cannot match the original when $q < 70$. However, when we try to match high order POD moments, CORE becomes unstable due to the explicit moment matching method it uses.

In this paper we propose an efficient yet accurate model order reduction method EMPIRE for physical design with multiple parameters. Compared with CORE, with a small reduction size, it uses implicit moment matching to match high order POD moments, which is more accurate than the explicit moment matching used in CORE; in addition, it can match the moments of different PODs with different accuracy according to their influence on the objective. Experimental results show that compared with CORE and [3], EMPIRE results in $47.8X$ improved accuracy at a similar runtime.

The rest of the paper is organized as follows: In Section 2, we introduce the basic moment matching theorem for the parameterized model order reduction. In Section 3, EMPIRE algorithm are introduced. We present experimental results in Section 4 and give concluding remarks in Section 5. Proofs of theorems are included in a technical report [8].

## 2. PRELIMINARIES

Without loss of generality, a linear RLC circuit can be described by the following modified nodal analysis (MNA) equation:

$$\begin{aligned}(Gx + C\frac{dx}{dt}) &= Bu \\ y &= L^T x,\end{aligned} \tag{1}$$

where $G$ and $C$ ($\in R^{N\times N}$) are the state matrices, $B$ ($\in R^{N\times p_0}$) and $L$ ($\in R^{N\times q_0}$) are the I/O incidence matrices respectively, $x$ ($\in R^N$) is the state variable, $u$ ($\in R^{p_0}$) is the input current vector and $y$ ($\in R^{q_0}$) is the output vector.

In physical design and optimization such as the interconnect sizing and spacing, decap budgeting, the state matrices $G$ and $C$ are no longer constant. Instead, they can be expressed as functions, or more exactly, matrix functions of the parameters of interest. If we denote $p_1$, $p_2$, ..., $p_r$ as the parameters, then $G = G(p_1, p_2, ..., p_r)$ and $C = C(p_1, p_2, ..., p_r)$.

By Talyor expansion and variable renaming on $G$ and $C$ matrices [3], we can always cast the parameterized system into the following first order canonical form, where $s_i$ ($1 \le i \le t$) are the new variables, and $E_i$ are their corresponding constant coefficient matrices.

$$\begin{aligned}(E_0 + E_1 s_1 + E_2 s_2 + ... + E_t s_t)x &= Bu \\ y &= L^T x,\end{aligned} \tag{2}$$

In [3], V is computed such that

$$\begin{aligned}colspan(V) \;=\; span\{&\cup_{m=0}^{m_q} \cup_{k_2=0}^{m-(k_p+...+k_3)} ... \cup_{k_p=0}^{m} \\ &F^m_{k_2,...,k_p}(M_1, ..., M_p)B_M\},\end{aligned} \tag{3}$$

where $B_M = E_0^{-1}B$, and $F^m_{k_2,...,k_p}(M_1, ..., M_p)$ is calculated recursively as discussed in [3]. It is proved that projecting using the obtained $V$ can match the first $m_q$ moments of each POD.

Once the projection matrix $V$ is found, the reduced system can be obtained as

$$\begin{aligned}(\tilde{E}_0 + \tilde{E}_1 s_1 + \tilde{E}_2 s_2 + ... + \tilde{E}_t s_t)\tilde{x} &= \tilde{B}u \\ \tilde{y} &= \tilde{L}^T \tilde{x},\end{aligned} \tag{4}$$

where $\tilde{E}_i = V^T E_i V$, $\tilde{B} = V^T B$ and $\tilde{L} = V^T L$.

The size of the reduced circuit, i.e., the number of columns in the projection matrix $V$, is proportional to the parameter number $t$. When the parameter number is huge, which is usually the case in VLSI/ULSI design, the reduced system is still very large. In addition, numerical error is also a concern when the projection matrix is too large.

In the following section, we will propose the EMPIRE algorithm to reduce (3) to a projection matrix $\mathcal{V}$ with a much small column number.

## 3. ALGORITHM

To deal with large number of parameters, we propose our EMPIRE algorithm which is composed of three steps: *parameter number reduction*, *projection space collapse* and *frequency domain moment expansion*. The key idea of the first step is to pick the parameters with relative small perturbation on the original system according to the norms of their coefficient matrices. The key idea of the second step is to construct a projection matrix for the reduced parameters from step 1 and collapse it into a much smaller one with minimal error. This step has three sub-algorithms to select from according to the problem size. The key idea of the last step is to expand the collapsed projection matrix to match high order frequency domain moments. The main flow of the EMPIRE algorithm is shown in Figure 2.

Note that for the simplicity of presentation, the algorithm is described for the circuits with single port. It also works for circuits with multiple ports by minor changes.

### 3.1 Parameter Number Reduction

To start with, we have the following definition:

DEFINITION 1. *We define the significance of a parameter $s_i$ w.r.t. its coefficient matrix $E_i$ as*

$$SIG(s_i) = ||E_i||_2 \widetilde{s}_i, \tag{5}$$

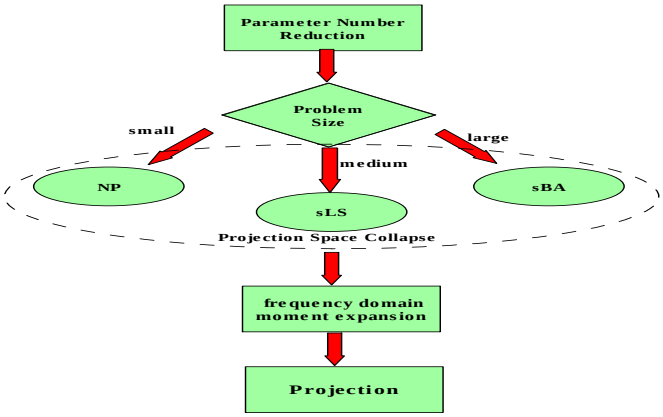


**Figure 2: The overall flow for EMPIRE algorithm.**

*where $||E_i||_2$ is the square-norm of $E_i$* [1]*, and $\widetilde{s}_i$ is an estimation of $s_i$. $\widetilde{s}_i$ does not need to be very accurate, for example either the expectation or the maximum value of $s_i$ can be used. Practically, they are known prior to optimization.*

The algorithm for *Parameter Number Reduction* is outlined in Algorithm 1. It computes *Aver*, the average of the significance for all the parameters. Those parameters that have coefficient matrices with norm smaller than $\epsilon \times Aver$ are removed when constructing the projection matrix. Here $\epsilon$ is user-specified: smaller $\epsilon$ results in smaller parameter number with less accuracy, while larger $\epsilon$ results in larger parameter number with higher accuracy.

**Algorithm 1** Parameter Number Reduction

**INPUT**: Total parameter number $t$ and the coefficient matrices $E_i$ $(1 \leq i \leq t)$ for each parameter;
**OUTPUT**: Total parameter number $w$ after parameter number reduction and the parameters $\hat{s}_1$,...,$\hat{s}_w$ ;
**INITIALIZATION**: Sum=0, j=0;;
**for** i=1; $i \leq t$; i++ **do**
  Sum=Sum+$SIG(s_i)$;
**end for**
Aver=Sum/t;
**for** i=1; $i \leq t$; i++ **do**
  **if** $||E_i||_2 \geq \epsilon \times Aver$ **then**
    $\hat{s}_j = s_i$;
    $j++$;
  **end if**
**end for**
$w = j$;

The correctness of this algorithm is guaranteed by the following theorem:

THEOREM 1. *Let $\mu$ be any eigenvalue of $E_0+E_is_i(E_0, E_i \in R^{N\times N})$ and $\lambda$ be the eigenvalues of $E_0$. We have*

$$\min_{\lambda\in\lambda(E_0)} |\lambda - \mu| \leq k||E_i||_2 s_i \propto SIG(s_i), \tag{6}$$

*where $k$ is a constant.*

Theorem 1 indicates that the distance between any eigenvalue of $E_0$ and $E_0 + E_is_i$ is proportional to SIG($s_i$). Since eigenvalues explicitly decide the behavior (such as time constant) of the circuit, it is natural that parameters with smaller significance have smaller impact on the circuit performance.

[1] The 2-norm of a vector is defined as the square root of the elements' square sum, while the 2-norm of a matrix $A$ is defined as $\sup_{x\neq 0} \frac{||Ax||_2}{||x||_2}$

Theorem 1 also indicates that the perturbation on the eigenvalues caused by the variation of a parameter is proportional to the norm of that parameter's coefficient matrix. With proof for Theorem 1 in a technical report [8], we validate it on 20 RC meshes, each with 10000 nodes and 5000 parameters (pitch width). We perturb the 2-norm of the coefficient matrix by multiplying it with a constant number and measure the perturbation of the output by integrating the normalized absolute difference $|y_n - y_o|/max|y_n - y_o|$, where $y_n$ is the original output and $y_o$ is the output after perturbation. We finally average the normalized perturbation over all the meshes. The relationship between the 2-norm of the coefficient matrices of the parameters and the perturbation of the output is shown in Figure 3. It is clear that the perturbation increases significantly with the 2-norm of the coefficient matrix, which experimentally validates the Theorem 1.

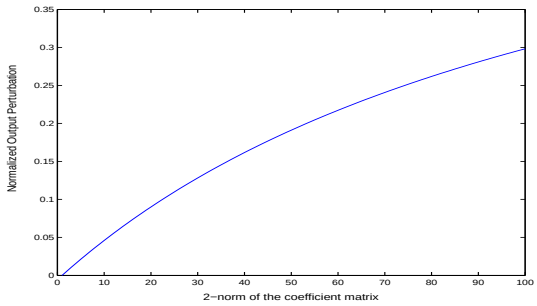


**Figure 3: The relationship between the 2-norm of the coefficient matrix and the output perturbation.**

## 3.2 Projection Space Collapse

After the parameters with the large significance are selected, we can construct a projection matrix $V$ based upon the reduced parameters. Similar method as in [3] is used to construct this projection matrix. Usually $V$ still has a large column number. By *Projection Space Collapse*, we reduce the column number of $V$ $(\in R^{N\times p})$ by finding a matrix $\hat{V}$ $(\in R^{N\times q_0})$ which has much smaller column number than $V$ (i.e.,$q_0 \ll p$), while they are as "close" as possible.

To quantitatively measure how "close" two matrices are, we have the following definitions:

DEFINITION 2. *The distance from a vector $r$ to the space spanned by the column vectors of $\hat{V}$, colspan($\hat{V}$), is defined as*

$$d(r, \hat{V}) = \min_{\forall\hat{v}\in colspan(\hat{V})} ||r - \hat{v}||_2, \tag{7}$$

*where $\hat{v}$ is any vector in colspan($\hat{V}$)*

DEFINITION 3. *The weighted distance between colspan($V$) and colspan($\hat{V}$), colspan($\hat{V}$), is defined as*

$$D(V, \hat{V}) = \sum_{i=1}^{p} W_i d(V(i,:), \hat{V}), \tag{8}$$

*where $V(i,:)$ is the $i$-th column in $V$, and $W_i$ are the weights.*

Note that the weights in Definition 3 are used to reflect the significance difference between PODs as well as the dif-

| Algorithm | NP | sLS | sBA |
|---|---|---|---|
| Runtime | Slow | Medium | Fast |
| Accuracy | High | Medium | Low |

**Table 1: Runtime and accuracy comparison between nonlinear programming, sequential Least Square and sequential Barycenter Allocation.**

ference between different moment orders, i.e.,

$$W(\hat{s}_1, \hat{s}_2, ..., \hat{s}_k, i) = (\sum_{j=1}^{k} SIG(s_j)/(k * i), \quad (9)$$

where $W(s_1, s_2, ..., s_k, i)$ is the weight for the $i$-th joint moment of any k parameters $s_1, s_2, ..., s_k$. For example, the weight for the 1st moment of $p_1$, $M_1B_m$, is simply $SIG(s_1)$, and the weight for the 2nd joint moment of $(p_1, p_2)$, $(M_1M_2 + M_2M_1)B_m$, is $(SIG(s_1) + SIG(s_2))/4$. We do not use the arithmetic average here because the lower order moments are more critical in terms of accuracy, and thus should have heavier weights. The weights can also be user-defined, reflecting the importance of the corresponding parameter.

Along with the definitions, in order to minimize the total error caused by the *Projection Space Collapse*, the following optimization problem needs to be solved:

FORMULATION 1. *Find an orthonormalized matrix* $\hat{V}$ $(\in N \times q_0)$*, such that* $D(V, \hat{V})$ *is minimized.*

In the following, we propose three methods to solve the problem: an exact algorithm via nonlinear programming (NP); a greedy algorithm via sequential least square (sLS); and a greedy algorithm via sequential Barycenter Allocation (sBA). The three methods offer a spectrum of tradeoffs between runtime and accuracy, as shown in Table 1.

### 3.2.1 Exact Algorithm via Nonlinear Programming

If we denote $\hat{V} = [\hat{v}_1, \hat{v}_2, \hat{v}_3, ..., \hat{v}_{q_0}]$ with $\hat{v}_i$ indicating the $i$-th column in $\hat{V}$, they constitute an orthonormalized basis of colspan($\hat{V}$) since $\hat{V}$ is orthonormalized. We also denote $\hat{v}_{q_0+1}, \hat{v}_{q_0+2}, ..., \hat{v}_N$ as the basis of the subspace orthogonal to colspan($\hat{V}$). To cast the problem formulation into a mathematical problem, we have the following theorem:

THEOREM 2. *The distance defined in (7) can be expressed as:*

$$(\sum_{i=q_0+1}^{N} (\hat{v}_i^T r)^2)^{1/2}. \quad (10)$$

By Theorem 2, the original problem can be casted into the following nonlinear optimization problem:

$$\min_{\hat{v}_i \in R^N} \sum_{i=1}^{p} W_i (\sum_{j=q_0+1}^{N} (\hat{v}_j^T v_i)^2)^{1/2}$$
$$s.t. \quad \hat{v}_i^T \hat{v}_j = \delta_{ij}, \quad q_0 + 1 \le i, j \le N, \quad (11)$$

where $W_i$ is the weight for the moment presented by the $i$-th column in $V$ from (9), $N$ is the circuit size, $q_0$ is the reduced size, $\hat{v}_i$ and $v_i$ are the column vectors of the orthogonormalized projection matrices $\hat{V}$ and $V$, respectively, and $\delta_{i,j}$ is the Kroneck delta function, i.e.,

$$\delta_{i,j} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases}. \quad (12)$$

By solving this optimization problem, we can obtain a set of basis $\hat{v}_{q_0+1}, \hat{v}_{q_0+2}, ..., \hat{v}_N$. Then we can find $q_0$ vectors orthogonal to them by solving the under-determined linear equation:

$$[\hat{v}_{q_0+1}, \hat{v}_{q_0+2}, ...\hat{v}_{q_0+n}]^T x = \mathbf{0}, \quad (13)$$

This equation should have $q_0$ non-trivial solutions. Then apply Gramm-Schmidt orthogonal (GSO) transformation technique, we can obtain the $q_0$ basis $\hat{v}_1, \hat{v}_2, ...\hat{v}_{q_0}$ we want.

Note that if we minimize the norm-square instead of norm, then (11) becomes a QCQP (Quadratic Constrained Quadratic Programming) problem with a total number of $(N - q_0)N$ variables after we insert $\hat{v}_i = [x_{i,1}, x_{i,2}, ..., x_{i,N}]^T$ $(q_0 + 1 \le i \le N)$.

We can solve the QCQP and obtain the projection matrix. Exact and Accurate as it is, when the original circuit size $N$ is large, this method becomes impractical in terms of runtime. In the following, we present two alternative heuristic methods for speedup.

### 3.2.2 Greedy Algorithm via Sequential Least Square

The heuristic methods in this section and Section 3.2.3 exploit the other interpretations of a column in a matrix: it can be viewed either as a function where the first N-1 elements indicate the coordinates, and the the last element indicate the value, or it can be viewed as a point in the N-dimension space. In this section, we use the first interpretation.

Instead of deciding the projection matrix $\hat{V}$ directly, we use a sequence of N-dimension least square problems (sLS) to solve it, with a total iteration number of $q_0$. In each iteration, we look for a line to fit the data set given by all the column vectors in the original projection matrix $V$. We treat the first $N - 1$ dimensions as independent variables, and the last dimension as dependent variable, we try to find a linear relationship to fit the data set:

$$v_n = a_1 v_1 + a_2 v_2 + ... + a_{N-1} v_{N-1}, \quad (14)$$

such that the least square error is achieved, i.e.,

$$\Phi = \sum_{i=1}^{p} W_i (v_{N,i} - (a_1 v_{1,i} + a_2 v_{2,i} + ... + a_{N-1} v_{N-1,i}))^2 = min, \quad (15)$$

where $v_{k,j}$ denotes the element at the $k$-th row and $j$-th column of the original projection matrix $V$.

Note that $a_1$,$a_2$,... and $a_{N-1}$ are unknown coefficients while all the other variables are given. To obtain the least square, the unknown coefficients $a_1$,$a_2$,..., and $a_{N-1}$ must yield zero first order derivatives, i.e.,

$$\begin{cases} \frac{\partial \Phi}{\partial v_1} &= 2 \sum_{i=1}^{p} W_i v_{1,i} (v_{N,i} - \sum_{j=1}^{p} u_j) = 0 \\ \frac{\partial \Phi}{\partial v_2} &= 2 \sum_{i=1}^{p} W_i v_{2,i} (v_{N,i} - \sum_{j=1}^{p} u_j) = 0 \\ \cdots & \\ \frac{\partial \Phi}{\partial v_{N-1}} &= 2 \sum_{i=1}^{p} W_i v_{N-1,i} (v_{N,i} - \sum_{j=1}^{p} u_j) = 0 \end{cases}, \quad (16)$$

where

$$u_j = v_{N,j} - (a_1 v_{1,j} + a_2 v_{2,j} + ... + a_{N-1} v_{N-1,j}) \quad (17)$$

Expanding the above equations, we have

$$Ax = B, \quad (18)$$

where

$$A = \begin{pmatrix} \sum_{i=1}^{p} W_i v_{1,i}^2 & \cdots & \sum_{i=1}^{p} W_i v_{1,i} v_{N-1,i} \\ \sum_{i=1}^{p} W_i v_{1,i} v_{2,i} & \cdots & \sum_{i=1}^{p} W_i v_{2,i} v_{N-1,i} \\ \vdots & \ddots & \vdots \\ \sum_{i=1}^{p} W_i v_{N-1,i} W_i v_{1,i} & \cdots & \sum_{i=1}^{p} W_i v_{N-1,i}^2 \end{pmatrix} \quad (19)$$

,

$$x = \begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_{N-1} \end{pmatrix}, \quad (20)$$

and

$$B = \begin{pmatrix} \sum_{i=1}^{p} W_i v_{1,i} v_{N,i} \\ \sum_{i=1}^{p} W_i v_{2,i} v_{N,i} \\ \vdots \\ \sum_{i=1}^{p} W_i v_{N-1,i} v_{N,i} \end{pmatrix} \quad (21)$$

After the direction of the line $[a_1, a_2, ..., a_{N-1}, -1]$ is found, those column vectors are transformed to be orthogonal to the optimal vector by GSO. The physical explanation of this step is to remove the projections on the newly generated vector. By doing so, the vectors in $V$ and in $\hat{V}$ are always orthogonal and thus cannot be represented by the linear combination of the column vectors in current $\hat{V}$. The algorithm is outlined in Algorithm 2.

**Algorithm 2** Sequential Quadratic Programming

**INPUT**: Original Projection Matrix $V = [v_1, ..., v_m] \in R^{N \times p}$, reduced order $q_0$;
**OUTPUT**: Projection Matrix $\hat{V}$ $(\in R^{N \times q_0})$;
**INITIALIZATION**: $\hat{V} = \Phi$;
**for** i=1; $i \leq q_0$; i++ **do**
  **SOLVE**: LS problem (18);
  **for** j=1;$j < i$;j++ **do**
    $\hat{v} = \hat{v} - (\hat{v}_j^T \hat{v})\hat{v}_j$ (1);
  **end for**
  **if** $||\hat{v}||_2 \leq \epsilon$ **then**
    Continue;
  **end if**
  **for** j=1; $j < i$; j++ **do**
    $\hat{v}_j = \hat{v}_j - (\hat{v}_j^T \hat{v})\hat{v}$ (2);
  **end for**
  $\hat{v} = \frac{\hat{v}}{||\hat{v}||_2}$;
  $\hat{V} = [\hat{V}, \hat{v}]$;
**end for**

### 3.2.3 Greedy Algorithm via Sequential Barycenter Allocation

The method via sLS still requires solving a sequence of $N$ dimension linear equation, the complexity of which is $O(N^3)$. This is enough for medium-scale problems. However, when dealing with circuits with millions of nodes, it is still unacceptable. To this end, by exploiting the second interpretation mentioned at the beginning of Section 3.2.2, we propose a barycenter allocation based approach (sBA), which is has a linear complexity.

The main idea for sBA is that instead of looking for the line with least square method, we try to locate a point in the N-dimension space that has the smallest sum of distance sum to the points represented by the column vectors in $V$. We have the following theorem:

THEOREM 3. *The barycenter of a set of points $x_1, x_2, ..., x_k$ with weights $W_1, W_2, ..., W_k$, i.e., the point*

$$x = \sum_{i=1}^{k} W_i x_i \quad (22)$$

*minimizes the weighted sum of the square-norm distance to all the points.*

Theorem 3 indicates that in each step we only need to find out the barycenter of the points represented by the column vectors in $V$ with weights computed from (9). The algorithm is outlined in Algorithm 3. Compared with Algorithm 2, the only difference is that the quadratic programming is replaced by barycenter computation.

**Algorithm 3** Sequential Barycenter Allocation

**INPUT**: Original Projection Matrix $V = [v_1, ..., v_m] \in R^{N \times p}$, reduced order $q_0$;
**OUTPUT**: Projection Matrix $\hat{V}$ $(\in R^{N \times q_0})$;
**INITIALIZATION**: $\hat{V} = \Phi$;
**for** i=1; $i \leq q_0$; i++ **do**
  **COMPUTE**: $\hat{v} = \frac{1}{p} \sum_{i=1}^{p} W_i v_i$;
  **for** j=1;$j < i$;j++ **do**
    $\hat{v} = \hat{v} - (\hat{v}_j^T \hat{v})\hat{v}_j$ (1);
  **end for**
  **if** $||\hat{v}||_2 \leq \epsilon$ **then**
    Continue;
  **end if**
  **for** j=1; $j < i$; j++ **do**
    $\hat{v}_j = \hat{v}_j - (\hat{v}_j^T \hat{v})\hat{v}$ (2);
  **end for**
  $\hat{v} = \frac{\hat{v}}{||\hat{v}||_2}$;
  $\hat{V} = [\hat{V}, \hat{v}]$;
**end for**

## 3.3 Frequency Domain Moment Expansion and Projection

After *Projection Space Collapse*, a projection matrix $\hat{V}$ $(\in R^{N \times q_0})$ is obtained with small column number. In this step, we try to improve the accuracy by matching more moments in frequency domain.

The algorithm is outlined in Algorithm 4. The key step in it is from the following theorem:

THEOREM 4. *The projection matrix $\mathcal{V}$ obtained in Algorithm 4 can match up to the q-th order of the frequency domain moments.*

Furthermore, we choose to match only the $q$-th order frequency domain moments, instead of other $q$-th order joint frequency-POD moments due to the following theorem:

THEOREM 5. *Among all the q-th order moments $s_1^{i_1} s_2^{i_2} .... (i_1 + i_2 + , ... = q)$, the frequency domain moments $s_j^q$ ($s_j$ is the frequency variable) has the maximum influence on the output accuracy.*

After the final projection matrix $\mathcal{V}$ is obtained, we can use (4) to project the original system into a much smaller one.

## 3.4 Summary

In summary, in the first step *Parameter Number Reduction*, we reduce the number of total parameters by specifying $\epsilon$. Then in the second step *Projection Space Collapse*, we obtain a small projection matrix $\hat{V}$ $(\in R^{N \times q_0})$ which has the minimum distance to the original one by specifying the column number $p$ of the projection matrix to compute directly,

**Algorithm 4** Frequency domain moment expansion

**INPUT**: Projection matrix $\hat{V} \in R^{N \times q_0}$, reduced order $q$, and $E_s$, the coefficient matrix for the frequency variable $s$;
**OUTPUT**: Projection matrix $\mathcal{V}$ ($\in R^{N \times q}$);
**INITIALIZATION**: $\mathcal{V} = \hat{\mathcal{V}}$, $R = B_M$;
**for** i=0;i¡q;i++ **do**
  $R = E_0^{-1} E_s R$;
  $R_0$=orthogonormalize R w.r.t. $\mathcal{V}$;
  $\mathcal{V} = [\mathcal{V} \quad \mathcal{R}_l]$;
**end for**

and the column number $q_0$ of the projection matrix after collapse. We have three algorithms to choose from. In the last step *Frequency Domain Moment Expansion*, we match high order frequency domain moments by specifying the order $q$ of the frequency domain moments to match.

# 4. EXPERIMENTAL RESULTS

In this section, we present numerical experiments to demonstrate the efficiency and accuracy of the EMPIRE algorithm. All the algorithms are implemented in MATLAB, and run on a Linux workstation (P4 2.66G CPU and 2G RAM). We use different sizes of extracted RC meshes from industrial applications. We first compare the three algorithms proposed in this paper, and then compare our hybrid algorithm with the method in [3] and CORE [7].

For EMPIRE, we set the threshold $\epsilon = 0.5$ and compute the projection matrix $V$ to match the parameters to their 6-th order moments. Then we collapse it to $q_0 = w$, where $w$ is the reduced parameter number. The column number $q$ of the final projection matrix $\mathcal{V}$ varies in the experiments.

## 4.1 Comparison between sBA, sLA and NP

Figure 4 demonstrates how the distance between the original projection matrix and the reduced one generated by EMPIRE changes with the column number of the reduced one (i.e., the reduced circuit size). The experiments are run on 20 RC meshes, each with 10000 nodes and 5000 parameters (pitch width). For each of the three projection space collapse methods, the results are averaged over the 20 cases. As we expect, the distance drops down with the increase of the column number in the reduced projection matrix. When the reduced size is large enough, the distance converges to 0. This validates the algorithms for projection space collapse. In addition, note that the nonlinear programming method converges the fastest, and the sequential barrier allocation method converges the slowest.

## 4.2 Comparison with Existing Work

Figure 5 and Figure 6 show the time and frequency domain responses for a power grid with 20,000 nodes and 5000 parameters. The pitch widths are the parameters of design and the attenuated sine waveforms are applied at the input ports. For [3], we match all the moments up to the second order. For CORE and EMPIRE, we match the parameter moments up to 4th order, and the frequency domain moments up to $q = 200$. As we can see from the figures, EMPIRE matches the original well, better than both CORE and [3].

Figure 7 shows the output response v.s. a randomly selected pitch width with 30% variation around the nominal value $100\mu m$. As we can see from the figure, EMPIRE exactly matches the original, better than CORE and [3] do.

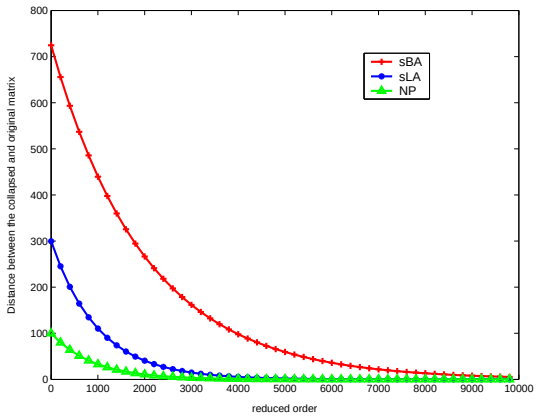


**Figure 4: The distance between the collapsed matrix and the original matrix versus the reduced order.**

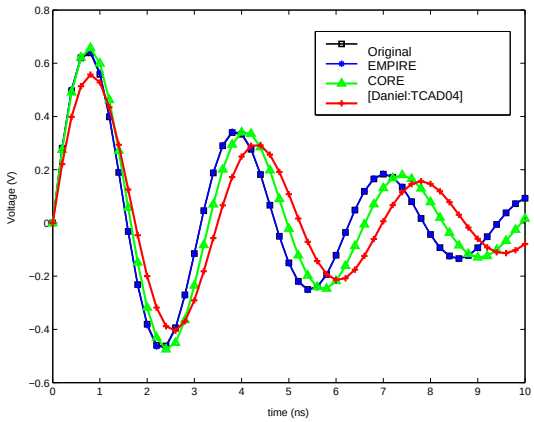


**Figure 5: The time domain waveform comparison between EMPIRE, CORE and [3]. EMPIRE is identical to the original.**

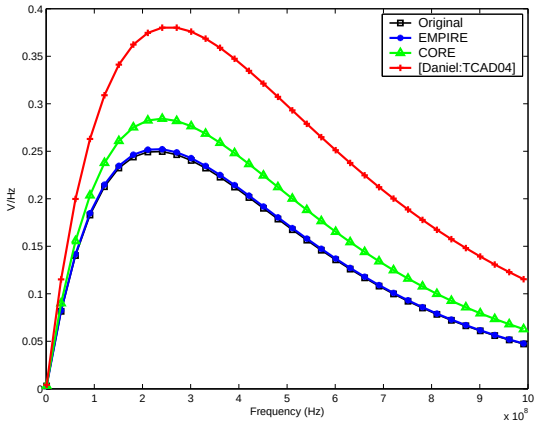


**Figure 6: The frequency domain response comparison between EMPIRE, CORE and [3]. EMPIRE is identical to the original.**

| Node # | P # | Reduced Size | | | model reduction time (s) | | | simulation time (s) | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | [3] | CORE | EMPIRE | [3] | CORE | EMPIRE | [3] | CORE | EMPIRE |
| 1400 | 600 | 940 | 40 | 40 | 1262 | 196 | 202 | 982.4 | 64 | 64 |
| 2450 | 1000 | 1880 | 60 | 60 | 9766 | 424 | 518 | 7832.1 | 129 | 128 |
| 5800 | 2600 | N/A | 80 | 80 | N/A | 1426 | 1577 | N/A | 224 | 226 |
| 7930 | 4800 | N/A | 100 | 100 | N/A | 2292 | 2784 | N/A | 312 | 311 |
| 12500 | 8400 | N/A | 120 | 120 | N/A | 4213 | 4910 | N/A | 471 | 471 |

**Table 2: Runtime comparison between [3], CORE and EMPIRE.**

Table 2 compares the runtime between the three methods on RC meshes of different scales. EMPIRE has a similar runtime compared with CORE, and is 18.3$X$ faster than [3] for model reduction time and 61.2$X$ faster for simulation time. In addition, [3] cannot finish large examples.

Figure 8 plots the runtime w.r.t. original circuit size for EMPIRE. The parameter number for each circuit is equal to 60% of the node number. All the circuits are reduced to the same size $q = 60$. Clearly the runtime for EMPIRE can be divided into three regimes, corresponding to the three different projection space collapse methods NP (A), sLS (B) and sBA (C). In regime A, the runtime increases rapidly with the original circuit size, with the gain of high accuracy; and in regime C the runtime increases slowly, at the cost of low accuracy.

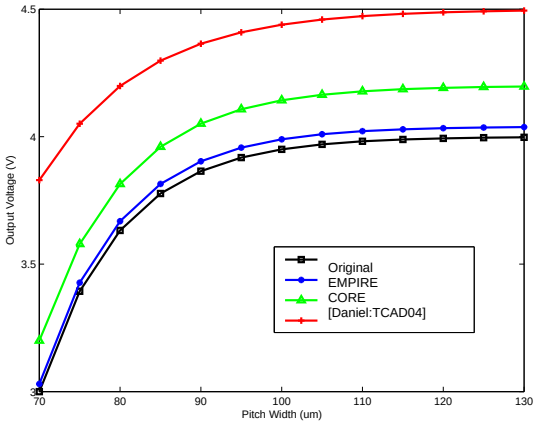


**Figure 7: The output response v.s. a randomly selected pitch width. EMPIRE is close to the original.**

Finally, we study the scalability for the EMPIRE algorithm. We use an RC mesh with 10000 nodes and 5000 parameters, and reduce it to different sizes from 100 to 1000 by EMPIRE and CORE. [3] is not included here because the reduced size is not controllable. Figure 9 shows the time domain waveform relative error for EMPIRE and CORE, respectively. EMPIRE always has the minimum waveform error. At order $q = 600$, EMPIRE is 47.8$X$ more accurate than CORE. This should be fully credited to the higher moment matching accuracy as well as the numerical stability from the implicit moment matching.

## 5. CONCLUSIONS

In this paper we propose an efficient yet accurate parameterized model order reduction method EMPIRE for physical design and optimization with large circuits and multiple parameters. We compare EMPIRE with the best existing work CORE. Experimental results show that EMPIRE results in 47.8$X$ improved accuracy at a similar runtime.

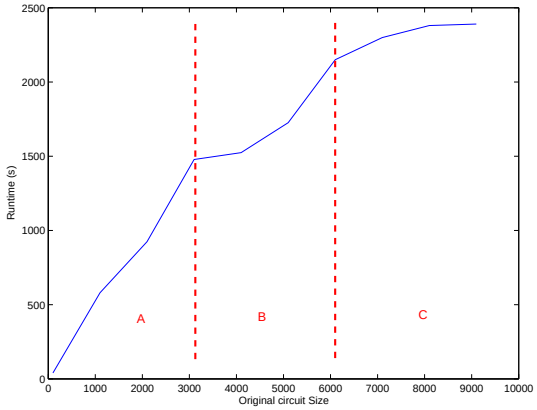


**Figure 8: The runtime for EMPIRE w.r.t. different original circuit size.**

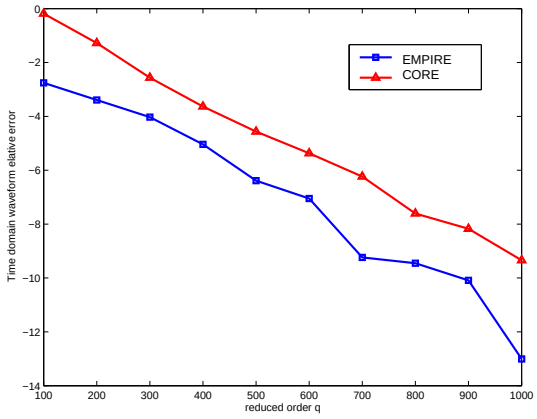


**Figure 9: The accuracy comparison between EMPIRE and CORE on different reduction size.**

# 6. REFERENCES


[1] H. Su and et al, "An Algorithm for Optimal Decoupling Capacitor Sizing and Placement for Standard Cell Layouts," in *ISPD*, 2002.

[2] H. Yu and et al, "Thermal Via Allocation for 3D ICs Considering Temporally and Spatially Variant Thermal Power," 2006.

[3] L. Daniel and et al, "A Multiparameter Moment Matching Model Reduction Approach for Generating Geometrically Parameterized Interconnect Performance Models," *TCAD*, 5 2004.

[4] Y. Cai and et al, "Relaxed Hierarchical Power/Ground Grid Analysis," in *ASPDAC*, 2005.

[5] A. Odabasioglu and et al, "PRIMA: Passive reduced-order interconnect macromodeling algorithm," *TCAD*, pp. 645–654, 1998.

[6] Y. Su and et al, "SAPOR: Second-Order Arnoldi Method for Passive Order Reduction of RCS Circuits," in *ICCAD*, 2004.

[7] X. Li and et al, "Parameterized interconnect order reduction with explicit-and-implicit multi-parameter moment matching for inter/intra-die variations," in *ICCAD*, 2005.

[8] http://eda.ee.ucla.edu/publications.html