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

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.

Parameterized model order reduction not only reduces the circuits to a much smaller size, but also keeps the parameters of design (POD) and thus the sensitivity information. [3] extended the Arnoldi method used in [4] 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. For better accuracy, CORE [5] was proposed to use 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 2. Accordingly, we need to obtain a higher accuracy for those moments associated with more important PODs. However, CORE can only evenly match the moments of each POD. Finally, matching only the first or second order of POD moments are not necessarily 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 as the design parameter. At reduced orders q=30,50,70, we plot 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 model 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.

*This project was partially supported by SRC project 1110 and 1116, and NSF CAREER award 0401682. Address comments to lhe@ee.ucla.edu.

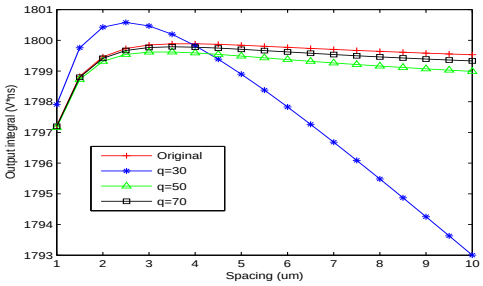

Figure 1: The output integral w.r.t. the wire spacing at different reduced order q. The reduced model cannot matches the origin well when $q < 70$. This shows only matching the first or second order POD moments is far from enough.

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

2. ALGORITHM

2.1 Preliminaries and Overview

We start with the following first order canonical form of a parameterized system, where s_i $(1 \le i \le t)$ are the variables, and E_i are their corresponding constant coefficient matrices.

$$(E_0 + E_1 s_1 + E_2 s_2 + ... + E_t s_t)x = Bu$$
$$y = L^T x, \quad (1)$$

In [3], V is computed such that

$$colspan(V) = span\{\cup_{m=0}^{m_q} \cup_{k_2=0}^{m-(k_p+...+k_3)} ... \cup_{k_p=0}^{m} F_{k_2,...,k_p}^{m}(M_1, ..., M_p)B_M\}, \quad (2)$$

where $B_M = E_0^{-1}B$, and $F_{k_2,...,k_p}^{m}(M_1, ..., M_p)$ 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.

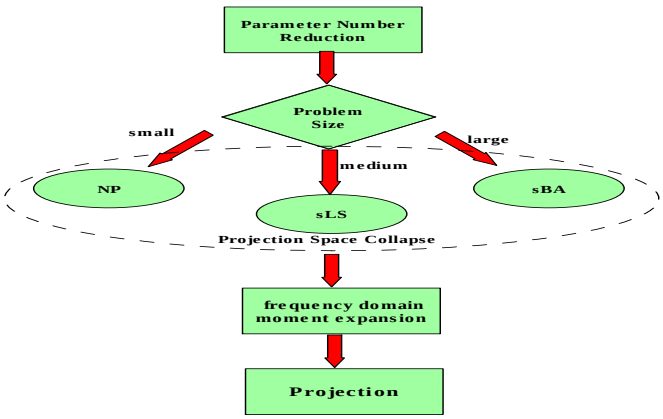

Figure 2: The overall flow for EMPIRE algorithm.

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

$$(\tilde{E}_0 + \tilde{E}_1 s_1 + \tilde{E}_2 s_2 + ... + \tilde{E}_t s_t)\tilde{x} = \tilde{B}u$$
$$\tilde{y} = \tilde{L}^T \tilde{x}, \quad (3)$$

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, we will propose the EMPIRE algorithm to reduce (2) to a projection matrix $\mathcal{V}$ with a much small column number. To deal with large number of parameters, EMPIRE 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.

2.2 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*

$$SIG(s_i) = ||E_i||_2 \widetilde{s}_i, \quad (4)$$

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.*

[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 $\sup_{x \neq 0} \frac{||Ax||_2}{||x||_2}$

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 ϵ is user-specified: a smaller ϵ results in a smaller parameter number with less 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 μ be any eigenvalue of $E_0 + E_i s_i$ ($E_0, E_i \in R^{N \times N}$) and λ be the eigenvalues of E_0. We have*

$$\min_{\lambda \in \lambda(E_0)} |\lambda - \mu| \leq k||E_i||_2 s_i \propto SIG(s_i), \quad (5)$$

where k is a constant.

Theorem 1 indicates that the distance between any eigenvalue of E_0 and $E_0 + E_i s_i$ 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. 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.

2.3 Projection Space Collapse

After the parameters with the large significance are selected, we can construct a projection matrix V based upon the reduced parameters. A method similar to that 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*

$$d(r, \hat{V}) = \min_{\forall \hat{v} \in colspan(\hat{V})} ||r - \hat{v}||_2, \quad (6)$$

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

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.

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

$$D(V,\hat{V}) = \sum_{i=1}^{p} W_i d(V(i,:),\hat{V}), \tag{7}$$

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 difference between different moment orders, i.e.,

$$W(\hat{s}_1,\hat{s}_2,...,\hat{s}_k,i) = (\sum_{j=1}^{k} SIG(s_j)/(k*i), \tag{8}$$

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.*

We proposed 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. For detailed algorithms, please see [6].

2.4 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 2. The key step in it is from the following theorem:

THEOREM 2. *The projection matrix $\mathcal{V}$ obtained in Algorithm 2 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 3. *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.*

Algorithm 2 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**
  $R=E_0^{-1}E_sR$;
  R_0=orthogonormalize R w.r.t. $\mathcal{V}$;
  $\mathcal{V}=[\mathcal{V} \quad \mathcal{R}_\ell]$;
end for

After the final projection matrix $\mathcal{V}$ is obtained, we can use (3) to project the original system into a much smaller one.

3. 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 and compare our hybrid algorithm with the method in [3] and CORE [5].

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.

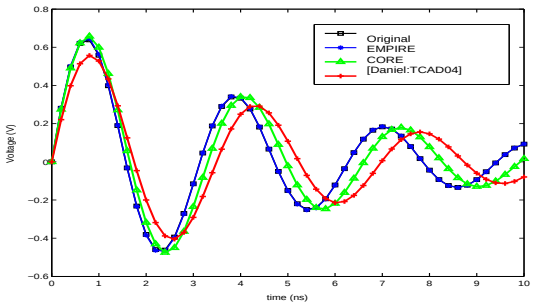

Figure 3: The time domain waveform comparison between EMPIRE, CORE and [3]. EMPIRE is identical to the original.

Figure 3 and Figure 4 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 5 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.

Table 2 compares the runtime between the three methods on RC meshes of different scales. EMPIRE has a similar

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.

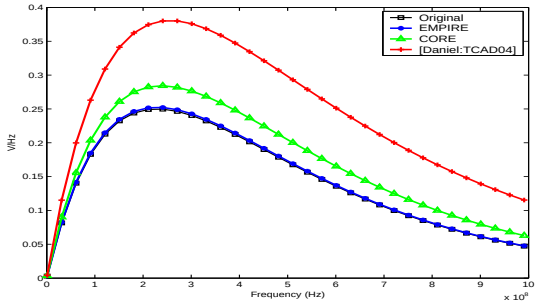

Figure 4: The frequency domain response comparison between EMPIRE, CORE and [3]. EMPIRE is identical to the original.

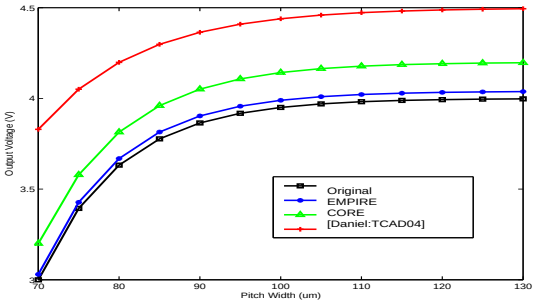

Figure 5: The output response v.s. a randomly selected pitch width. EMPIRE is close to the original.

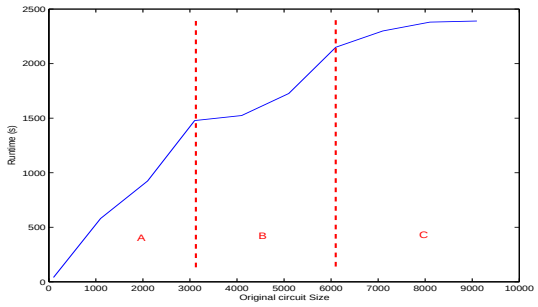

Figure 6: The runtime for EMPIRE w.r.t. different original circuit size.

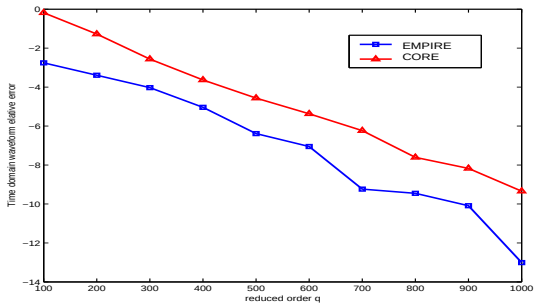

Figure 7: The accuracy comparison between EMPIRE and CORE on different reduction size.

runtime compared with CORE, and is 18.3X faster than [3] for model reduction time and 61.2X faster for simulation time. In addition, [3] cannot finish large examples.

Figure 6 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.

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 7 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 47.8X 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.

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

[5] 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.

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