Efficient Incremental Analysis of On-Chip Power Grid via Sparse Approximation

Pei Sun and Xin Li
ECE Department, Carnegie Mellon University
5000 Forbes Avenue, Pittsburgh, PA 15213
{peis, xinli}@ece.cmu.edu

Ming-Yuan Ting
Mentor Graphics Corporation
46871 Bayside Parkway, Fremont, CA 94538
ming_ting@mentor.com

ABSTRACT

In this paper, a new sparse approximation technique is proposed for incremental power grid analysis. Our proposed method is motivated by the observation that when a power grid network is locally updated during circuit design, its response changes locally and, hence, the incremental "change" of the power grid voltage is almost zero at many internal nodes, resulting in a unique sparse pattern. An efficient Orthogonal Matching Pursuit (OMP) algorithm is adopted to solve the proposed sparse approximation problem. In addition, several numerical techniques are proposed to improve the numerical stability of the proposed solver, while simultaneously maintaining its high efficiency. Several industrial circuit examples demonstrate that when applied to incremental power grid analysis, our proposed approach achieves up to 130× runtime speed-up over the traditional Algebraic Multi-Grid (AMG) method, without surrendering any accuracy.

Categories and Subject Descriptors

B.7.2 [**Integrated Circuits**]: Design Aids – Verification

General Terms

Algorithms

Keywords

Integrated Circuit, Power Grid, Incremental Analysis

1. INTRODUCTION

An on-chip power grid provides the voltage supply for all integrated devices on a silicon chip. It is an important component that directly impacts signal integrity and, eventually, chip functionality of today's large-scale integrated circuits (ICs). As the power density of high-performance ICs (e.g., microprocessors) continuously increases and the power grid network becomes increasingly complex, designing and verifying on-chip power grid emerges as a challenging task. A typical power grid network consists of millions of internal nodes and, hence, power grid analysis and optimization can be extremely time-consuming.

For this reason, a large number of new CAD tools have been developed for power grid design and verification [1]-[15]. Efficient numerical algorithms are applied by these tools to explore the unique structure of power grid network in order to reduce the computational cost. These existing techniques can be classified into several broad categories: (1) Krylov-subspace method [1], (2) hierarchical analysis [2], (3) multi-grid solver [3]-[6], (4) randomized algorithm [7]-[8], and (5) vectorless analysis [9]-[12]. The aforementioned CAD tools have been successfully applied to a wide range of practical power grid problems.

In this paper, we focus on a unique set of power grid analysis problems where a power grid network is repeatedly updated during circuit design and we are interested in knowing the power grid response once a change is made. Such an *incremental* analysis is substantially different from the general-purpose power grid analysis that is solved by most existing tools. Namely, it is computationally inefficient, if not impossible, to apply a general-purpose power grid solver to repeatedly analyze the large-scale power grid circuit for which a sequence of small changes are made. The open question here is how to efficiently and incrementally update the power grid response so that we do not need to solve the entire power grid network for many times.

Towards this goal, we propose a new sparse approximation technique for incremental power grid analysis. Our work is motivated by the observation that a power grid network is often updated with local changes (e.g., increasing wire width and/or inserting extra vias in a local region) during circuit design. In these cases, the response of the power grid network also changes locally. In other words, the incremental "change" of the power grid voltage is almost zero at many internal nodes, resulting in a unique sparse pattern. An efficient numerical solver can be developed to find the underlying sparse solution with low computational cost.

In this paper, we adopt the Orthogonal Matching Pursuit (OMP) algorithm [16]-[18] from the statistics community to formulate the proposed numerical solver. The OMP algorithm is particularly tuned to fit the needs of our application for incremental power grid analysis. It applies a heuristic method to recursively identify the non-zero elements of the sparse solution and then solve their values. As such, the problem size and, hence, the computational complexity are significantly reduced, since we only need to solve the unknown values of the non-zero voltage changes, instead of all node voltages. In addition, several efficient techniques (e.g., pre-conditioning) are proposed to improve the numerical stability of the proposed incremental power grid solver, while simultaneously maintaining its high efficiency. As will be demonstrated by the numerical examples in Section 5, our proposed incremental solver achieves orders of magnitude more efficiency (up to 130× speed-up) compared to an Algebraic Multi-Grid (AMG) solver without incremental analysis capability.

The remainder of this paper is organized as follows. In Section 2, we derive the mathematical formulation of incremental power grid analysis, and then describe the proposed sparse approximation technique in Section 3. Several efficient numerical techniques are developed in Section 4 to further improve the numerical stability of the proposed incremental power grid solver. The efficacy of the proposed algorithm is demonstrated by several industrial power grid examples in Section 5. Finally, we conclude in Section 6.

DAC'11, June 5-10, 2011, San Diego, California, USA

2. MATHEMATICAL FORMULATION

Without loss of generality, we consider a power grid network that consists of resistors, capacitors, inductors and input (voltage and/or current) sources. The modified nodal analysis (MNA) equation of the power grid network is linear and it can be written in the standard form:

$$(G + sC)\cdot x = b \tag{1}$$

where $b \in R^N$ denotes the input, $x \in R^N$ represents the state variables, $G,C \in R^{N\times N}$ are the system matrices, and N is the size of the MNA equation.

In this paper, we focus on DC analysis to calculate the IR drop of the power grid network. In this case, all capacitors are open, all inductors are short, and the MNA equation in (1) can be simplified as:

$$G \cdot x = b\,. \tag{2}$$

The goal of power grid analysis is to solve the linear equation in (2) to calculate the response x. In many practical applications, the MNA equation in (2) is extremely large (e.g., $N > 10^6$) and specific numerical algorithms are required to efficiently solve (2) with low computational cost, as is discussed in [1]-[15].

Next, we consider a change that is made to the power grid network (e.g., increasing wire width and/or inserting vias in a local region) during circuit design. We express the new MNA equation for the updated power grid network as:

$$\widetilde{G} \cdot \widetilde{x} = \widetilde{b}\,. \tag{3}$$

For incremental power grid analysis, we assume that Eq. (2) is already solved and its solution x is known. We are further interested in knowing the solution $\tilde{x}$ of (3) (namely, the response of the updated power grid network). Instead of solving (3) directly by a general-purpose power grid solver, we re-formulate (3) into an "incremental" form so that its solution $\tilde{x}$ can be determined by an efficient algorithm based on sparse approximation.

Combining (2) and (3) yields:

$$\widetilde{G} \cdot \delta = r \tag{4}$$

where

$$\delta = \widetilde{x} - x \tag{5}$$

$$r = \widetilde{b} - \widetilde{G} \cdot x\,. \tag{6}$$

Eq. (4) is the "incremental" MNA equation where the "change" δ of the power grid response is represented as the problem unknown. Once δ is found from (4), the updated response $\tilde{x}$ can be easily calculated from (5): $\tilde{x} = x + \delta$.

In most practical cases, a power grid network is often updated with local changes and, hence, the problem unknown δ in (4) is sparse. In other words, the incremental "change" of the power grid voltage is almost zero at many internal nodes, as will be demonstrated by the numerical examples in Section 5. Such a locality has been observed and reported in several previous works [14]-[15]. Solving the sparse solution δ from (4) is computationally cheaper than solving a general, non-sparse solution, since we only need to identify the non-zero elements in δ and then find their values. In the next section, an efficient numerical algorithm will be derived to approximate the solution δ by exploring the aforementioned sparsity.

3. SPARSE APPROXIMATION

In this section, an efficient Orthogonal Matching Pursuit (OMP) algorithm will be applied to solve the sparse solution δ of (4) with low computational cost. In what follows, we will describe the OMP algorithm in detail and highlight its novelties.

3.1 Orthogonal Matching Pursuit

The OMP algorithm was initially developed by the statistics community to solve the sparse solution of a linear equation [16]-[18]. Considering the MNA equation $\tilde{G}\cdot\delta = r$ in (4), OMP applies an iterative scheme to select a small subset of "important" elements in δ that are non-zero. At the end of the OMP iteration, all other elements that are not selected are forced to zero, thereby rendering a sparse solution δ. OMP has been recently applied to a number of CAD problems, e.g., large-scale performance modeling [19]-[20]. In this sub-section, we will first describe the detailed steps of the OMP algorithm and then explain why it has low computational complexity and is preferred over a general-purpose power grid solver.

Given the MNA equation $\tilde{G}\cdot\delta = r$ in (4), we conceptually consider each column of the matrix $\tilde{G}$ as a basis vector. These basis vectors are not necessarily orthogonal. The MNA equation $\tilde{G}\cdot\delta = r$ in (4) can be re-written as:

$$r = \delta_1\widetilde{G}_1 + \delta_2\widetilde{G}_2 + \cdots + \delta_N\widetilde{G}_N \tag{7}$$

where $\delta_i \in R$ is the i-th element of δ and $\tilde{G}_i \in R^N$ is the i-th column of $\tilde{G}$. Eq. (7) represents the right-hand-side vector r as the linear combination of all basis vectors $\{\tilde{G}_i;\ i = 1,2,...,N\}$.

The key idea of OMP is to iteratively select the important basis vectors based on the "normalized" inner product:

$$\rho_i = \left|\frac{r^T \cdot \widetilde{G}_i}{\widetilde{G}_i^T \cdot \widetilde{G}_i}\right| \quad (i = 1,2,\cdots,N)\,. \tag{8}$$

Mathematically, the normalized inner product ρ_i in (8) is the lest-squares solution of the over-determined linear equation: $\tilde{G}_i\cdot\rho_i = r$. If ρ_i is large, it implies that $\tilde{G}_i$ is an important basis vector as it contributes a significant portion of r. Hence, the corresponding coefficient δ_i in (7) should be non-zero.

Motivated by this observation, OMP first calculates the normalized inner product between r and each $\tilde{G}_i$. It finds the set of important basis vectors $\{\tilde{G}_{s1},\tilde{G}_{s2},\ldots,\tilde{G}_{sm}\}$ for which the normalized inner product values are greater than or equal to a user-defined threshold ε:

$$\rho_{s1} \ge \varepsilon \quad \rho_{s2} \ge \varepsilon \quad \cdots \quad \rho_{sm} \ge \varepsilon\,. \tag{9}$$

Once the important basis vectors $\{\tilde{G}_{s1},\tilde{G}_{s2},\ldots,\tilde{G}_{sm}\}$ are identified, OMP finds the optimal approximation for r by the linear combination of $\{\tilde{G}_{s1},\tilde{G}_{s2},\ldots,\tilde{G}_{sm}\}$:

$$r \approx \delta_{s1}\widetilde{G}_{s1} + \delta_{s2}\widetilde{G}_{s2} + \cdots + \delta_{sm}\widetilde{G}_{sm} \tag{10}$$

where the coefficients $\{\delta_{s1},\delta_{s2},\ldots,\delta_{sm}\}$ are determined by the following least-squares fitting:

$$\underset{\delta_{s1},\delta_{s2},\cdots,\delta_{sm}}{\text{minimize}} \quad \left\|\delta_{s1}\cdot\widetilde{G}_{s1} + \delta_{s2}\cdot\widetilde{G}_{s2} + \cdots + \delta_{sm}\cdot\widetilde{G}_{sm} - r\right\|_2^2\,. \tag{11}$$

In (11), $\|\bullet\|_2$ denotes the L_2-norm, i.e., the square root of the sum of the squares of all elements in the vector.

Next, OMP removes the components $\{\delta_{s1}\tilde{G}_{s1},\delta_{s2}\tilde{G}_{s2},\ldots,\delta_{sm}\tilde{G}_{sm}\}$ from r and calculates the residual:

$$e = r - \delta_{s1}\cdot\widetilde{G}_{s1} - \delta_{s2}\cdot\widetilde{G}_{s2} - \cdots - \delta_{sm}\cdot\widetilde{G}_{sm}\,. \tag{12}$$

The residue e is orthogonal to the basis vectors $\{\tilde{G}_{s1},\tilde{G}_{s2},\ldots,\tilde{G}_{sm}\}$ due to the least-squares fitting in (11). Based on (12), OMP further identifies another set of important basis vectors $\{\tilde{G}_{t1},\tilde{G}_{t2},\ldots,\tilde{G}_{tn}\}$ by calculating the normalized inner product between e and each $\tilde{G}_i$:

$$\xi_i = \left|\frac{e^T \cdot \widetilde{G}_i}{\widetilde{G}_i^T \cdot \widetilde{G}_i}\right| \quad (i = 1,2,\cdots,N)\,. \tag{13}$$

Since e is orthogonal to $\{\tilde{G}_{s1},\tilde{G}_{s2},\ldots,\tilde{G}_{sm}\}$, the new basis vectors

$\{\tilde{G}_{t1},\tilde{G}_{t2},\ldots,\tilde{G}_{tn}\}$ selected by (13) do not overlap with the previous set $\{\tilde{G}_{s1},\tilde{G}_{s2},\ldots,\tilde{G}_{sm}\}$. All basis vectors $\{\tilde{G}_{s1},\tilde{G}_{s2},\ldots,\tilde{G}_{sm}\}$ and $\{\tilde{G}_{t1},\tilde{G}_{t2},\ldots,\tilde{G}_{tn}\}$ are then combined to approximate r by the following least-squares fitting:

$$\underset{\substack{\delta_{s1},\cdots,\delta_{sm}\\ \delta_{t1},\cdots,\delta_{tn}}}{\text{minimize}} \quad \left\| \delta_{s1}\cdot\tilde{G}_{s1}+\cdots+\delta_{sm}\cdot\tilde{G}_{sm}+\delta_{t1}\cdot\tilde{G}_{t1}+\cdots+\delta_{tn}\cdot\tilde{G}_{tn}-r \right\|_2^2 . \quad (14)$$

Note that even though the coefficients $\{\delta_{s1},\delta_{s2},\ldots,\delta_{sm}\}$ were previously solved from (11), their values are re-calculated in (14) where the extra basis vectors $\{\tilde{G}_{t1},\tilde{G}_{t2},\ldots,\tilde{G}_{tn}\}$ are added. The aforementioned iteration continues until the residual e is sufficiently small. Algorithm 1 summarizes the major steps of the OMP algorithm. More details on OMP (e.g., convergence analysis) can be found in [16]-[18].

Algorithm 1: Orthogonal Matching Pursuit (OMP)

1. Start from the MNA equation $\tilde{G}\cdot\delta = r$ in (4).
2. Let the residual $e = r$ and the set $\Omega = \{\}$.
3. Based on (13), calculate the normalized inner product ξ_i between e and each $\tilde{G}_i$ where $i = 1,2,...,N$.
4. Select the set of basis vectors $\{\tilde{G}_{s1},\tilde{G}_{s2},\ldots,\tilde{G}_{sm}\}$ for which the normalized inner product values are greater than or equal to a user-defined threshold ε:
$$\xi_{s1} \ge \varepsilon \quad \xi_{s2} \ge \varepsilon \quad \cdots \quad \xi_{sm} \ge \varepsilon . \quad (15)$$
5. Update Ω by $\Omega = \Omega \cup \{s_1,s_2,\ldots,s_m\}$.
6. Approximate r by the linear combination of $\{\tilde{G}_i;\ i \in \Omega\}$:
$$r \approx \sum_{i\in\Omega} \delta_i \cdot \tilde{G}_i \quad (16)$$
where the coefficients are determined by least-squares fitting:
$$\underset{\delta_i;i\in\Omega}{\text{minimize}} \quad \left\| \sum_{i\in\Omega} \delta_i \cdot \tilde{G}_i - r \right\|_2^2 . \quad (17)$$
7. Calculate the residual:
$$e = r - \sum_{i\in\Omega} \delta_i \cdot \tilde{G}_i . \quad (18)$$
8. If the residual e is sufficiently small, stop iteration and go to Step 9. Otherwise, go to Step 3.
9. For any $\tilde{G}_i$ that is not selected (i.e., $i \notin \Omega$), the corresponding coefficient δ_i is set to 0.

Algorithm 1 requires a number of iterations to find all important basis vectors $\{\tilde{G}_i;\ i \in \Omega\}$ and their corresponding coefficients $\{\delta_i;\ i \in \Omega\}$. Each of these iterations consists of two major operations: (1) normalized inner product calculation (Step 3 of Algorithm 1), and (2) least-squares fitting (Step 6 of Algorithm 1). Since the MNA matrix $\tilde{G}$ in (4) is sparse, calculating the normalized inner product by (13) involves sparse matrix-vector operations and it can be performed with low computational cost.

On the other hand, to study the computational cost of least-squires fitting in Step 6 of Algorithm 1, we assume that the solution $\delta \in R^N$ of the MNA equation $\tilde{G}\cdot\delta = r$ is sparse. It contains K non-zeros where $K << N$. The least-squares fitting in (17) needs to find very few (i.e., up to K) unknown coefficients and, hence, also has low computational cost. The detailed algorithm for least-squares fitting will be discussed in Section 4.

Based on these observations, we can conclude that Algorithm 1 is much more efficient than a general-purpose power grid solver, since it aims to solve K non-zero elements for the sparse solution δ and the general-purpose solver needs to solve all N ($N >> K$) unknowns in δ. As will be demonstrated by the numerical examples in Section 5, our proposed incremental solver achieves orders of magnitude more efficiency (up to 130× speed-up) compared to an Algebraic Multi-Grid (AMG) solver without incremental analysis capability.

3.2 Comparison with Traditional Techniques

It is worth emphasizing that the proposed OMP solver is substantially different from other existing techniques for power grid analysis. For instance, the authors of [13] propose an incremental power grid analysis algorithm based on domain decomposition. The key idea is to re-use the Cholesky decomposition result of the original power grid network to solve the MNA equation of the updated system. While the domain decomposition method has been successfully demonstrated with high efficiency for several practical applications, it is only applicable to a limited set of power grid analysis problems where a direct solver (i.e., Cholesky decomposition) is used. In other words, the domain decomposition method proposed in [13] becomes inefficient, or even inapplicable, if the multi-grid algorithm [3]-[6] or the randomized algorithm [7]-[8] is used as the core numerical engine for large-scale power grid analysis. The proposed OMP technique, however, is generally applicable to all cases where the original power grid network can be solved by any numerical solver that a user selects.

On the other hand, the locality of power grid networks has been explored in [14]-[15] to speed-up power grid analysis. The method proposed in [14] aims to partition the power grid network based on its geometrical structure in order to reduce computational cost. In this paper, we further extend the locality concept to incremental power grid analysis. In addition, unlike the algorithm in [14] that needs to know the geometrical structure of the power grid, the proposed OMP method takes the MNA equation as the only input. It automatically determines the internal nodes for which the incremental change of the node voltage should be zero (or non-zero). From this point of view, the proposed incremental analysis engine is not constrained to any specific geometrical structure and it can be generally applied to a broad range of practical power grid circuits.

4. IMPLEMENTATION DETAILS

To make the OMP algorithm of practical utility, a number of implementation issues must be carefully considered. In this section, we will outline these implementation issues and then develop efficient numerical techniques to address them.

4.1 Least-Squares Fitting

The least-squares fitting in (17) is the most expensive step within the OMP iteration loop and it often dominates the overall computational cost. Hence, an efficient algorithm must be developed to solve (17) so that the computational cost of Algorithm 1 is minimized. Without loss of generality, we re-write (17) into the matrix form:

$$\underset{v}{\text{minimize}} \quad \left\| A\cdot v - r \right\|_2^2 \quad (19)$$

where $A \in R^{N\times K}$ contains the basis vectors $\{\tilde{G}_i;\ i \in \Omega\}$ that are selected by OMP, $v \in R^K$ contains the corresponding coefficients $\{\delta_i;\ i \in \Omega\}$, and K is the total number of these unknown coefficients. The optimization in (19) aims to solve the least-squares solution v of the following over-determined linear equation:

$$A\cdot v = r . \quad (20)$$

In most practical cases, QR decomposition is used to solve an

over-determined linear equation [21]. Given $A \cdot v = r$ in (20), we first decompose A as:

$$A = Q \cdot W \tag{21}$$

where $Q \in R^{N \times K}$ contains K orthonormal vectors (i.e., $Q^T Q = I$ where I is an identify matrix), and $W \in R^{K \times K}$ is an upper triangular matrix. Substituting (21) into (20), the least-squares solution v can be represented as [21]:

$$v = W^{-1} \cdot \left(Q^T r\right). \tag{22}$$

Note that it is not necessary to explicitly calculate the matrix inverse W^{-1} in (22). Since the matrix W is upper triangular, the linear equation in (22) can be solved by a sequence of backward substitutions.

While the aforementioned QR method has been widely applied to many practical applications, it is not the most efficient way to solve the over-determined linear equation in (20) for power grid analysis. Remember that the matrix A in (20) contains a large number of rows (i.e., N is large). In many practical cases, N is in the order of 10^6~10^8. Even though the matrix A is sparse, the matrix Q in (21) is not necessarily sparse, since a large number of non-zero fill-ins can be generated by QR decomposition. For this reason, the computational cost of forming the matrices Q and W in (21) can be prohibitive for large-scale problems.

An alternative approach to solve the over-determined linear equation in (20) is based on pseudo-inverse [21]:

$$v = \left(A^T A\right)^{-1} \cdot \left(A^T r\right). \tag{23}$$

In (23), since $A^T A$ is positive-definite, we can calculate its Cholesky decomposition:

$$A^T A = P \cdot \left(L^T L\right) \cdot P^T \tag{24}$$

where $L \in R^{K \times K}$ is a lower triangular matrix with positive diagonal elements, and $P \in R^{K \times K}$ is a permutation matrix that is used to maximize the sparsity of L [21]. Note that the matrix $A^T A \in R^{K \times K}$ is much smaller than the matrix $A \in R^{N \times K}$ where $K << N$ in our application. In other words, unlike the QR decomposition that is performed on a large-size matrix A, the Cholesky decomposition in (24) is applied to a small-size matrix $A^T A$ and, hence, can be efficiently computed with low computational cost.

Substituting (24) into (23) yields:

$$v = P \cdot L^{-1} \cdot L^{-T} \cdot \left(P^T A^T r\right). \tag{25}$$

Once the lower triangular matrix L is found by Cholesky decomposition, the linear equation in (25) can be easily solved by a sequence of forward and backward substitutions. Hence, the aforementioned Cholesky method is much more computationally efficient than the QR approach.

However, the low computational cost of the Cholesky method comes with a penalty. It is well-known that the solution v solved by pseudo-inverse is not numerically stable [21]. To understand this numerical issue, we note that the condition number of the matrix $A^T A$ in (23) is equal to the square of the condition number of the matrix A. In other words, the pseudo-inverse in (23) substantially increases the condition number and, hence, can result in an inaccurate solution. It, in turn, motivates us to propose several important techniques to mitigate this numerical problem, as will be discussed in detail in the next sub-section.

4.2 Improving Numerical Stability

To address the aforementioned numerical issues, two efficient techniques will be developed in this sub-section: (1) pre-conditioning, and (2) adaptive algorithm selection. In what follows, we will describe the mathematical formulations and detailed implementations of these two techniques.

1) Pre-conditioning: The key idea of pre-conditioning is to scale each column of the matrix A in (20) so that its condition number is reduced. While there are many possible options to perform pre-conditioning, we apply the following simple-yet-efficient scaling to the matrix A in (20):

$$\widetilde{A} = A \cdot D^{-1} \tag{26}$$

where $D \in R^{K \times K}$ is a diagonal matrix:

$$D = \begin{bmatrix} d_1 & & & 0 \\ & d_2 & & \\ & & \ddots & \\ 0 & & & d_K \end{bmatrix}. \tag{27}$$

In (27), the diagonal elements $\{d_i;\ i = 1,2,\ldots,K\}$ are equal to:

$$d_i = \left\| A_i \right\|_2 \quad \left(i = 1,2,\cdots,K\right) \tag{28}$$

where $A_i \in R^N$ stands for the i-th column of the matrix A. In other words, the pre-conditioning scheme in (26)-(28) normalizes each column of A. Substituting (26) into (20), the linear equation $A \cdot v = r$ can be re-written as:

$$\widetilde{A} \cdot \widetilde{v} = r \tag{29}$$

$$v = D^{-1} \cdot \widetilde{v}\ . \tag{30}$$

After pre-conditioning, we first solve the solution $\tilde{v}$ from (29) and then find the solution v from (30). As will be demonstrated by the numerical examples in Section 5, the proposed pre-conditioning is able to reduce the condition number of $A^T A$ by orders of magnitude (up to 10^4×).

2) Adaptive algorithm selection: Even though the aforementioned pre-conditioning can effectively improve the numerical stability, it is possible that the matrix $\tilde{A}$ in (29) remains ill-conditioned. This can happen if the power grid network is ill-conditioned (e.g., containing a number of extremely small resistors). In these cases, we want to apply the QR method to solve the over-determined linear equation in (29), since it is more numerically stable than the Cholesky approach. On the other hand, if the matrix $\tilde{A}$ is well-conditioned, the Cholesky approach is preferred, since it has low computational cost. For this reason, we need an efficient algorithm to estimate the condition number of $\tilde{A}$ so that the appropriate solver (i.e., either Cholesky decomposition or QR decomposition) can be selectively applied.

Remember that if Eq. (29) is solved by pseudo-inverse, we need to calculate the following Cholesky decomposition:

$$\widetilde{A}^T \widetilde{A} = \widetilde{P} \cdot \left(\widetilde{L}^T \widetilde{L}\right) \cdot \widetilde{P}^T \tag{31}$$

where $\tilde{L} \in R^{K \times K}$ is a lower triangular matrix with positive diagonal elements, and $\tilde{P} \in R^{K \times K}$ is a permutation matrix. Since the permutation matrix $\tilde{P}$ does not change the condition number [21], we have:

$$\kappa\left(\widetilde{A}^T \widetilde{A}\right) = \kappa\left(\widetilde{L}^T \widetilde{L}\right) = \left[\kappa\left(\widetilde{L}\right)\right]^2 \tag{32}$$

where $\kappa(\bullet)$ denotes the condition number of a matrix. Because the matrix $\tilde{L}$ is lower triangular with positive diagonal elements, its condition number is bounded by [21]:

$$\kappa\left(\widetilde{L}\right) \geq \frac{\sigma_{MAX}}{\sigma_{MIN}} \tag{33}$$

where σ_{MAX} and σ_{MIN} stand for the maximal and minimal diagonal elements of $\tilde{L}$, respectively. Combining (32)-(33), we have:

$$\kappa\left(\widetilde{A}\right) = \sqrt{\kappa\left(\widetilde{A}^T \widetilde{A}\right)} \geq \frac{\sigma_{MAX}}{\sigma_{MIN}}\ . \tag{34}$$

Eq. (34) reveals an important fact that once the Cholesky decomposition in (31) is calculated to solve the over-determined

linear equation $\tilde{A}\cdot\tilde{v} = r$, it can be re-used to estimate the condition number $\kappa(\tilde{A})$. We only need to check the diagonal elements of $\tilde{L}$ to compute the lower bound of the condition number in (34). Therefore, the additional computational cost for condition number estimation is negligible. If the condition number $\kappa(\tilde{A})$ is less than or equal to a user-defined threshold (say, μ), the Cholesky method will be used to solve the least-squares solution $\tilde{v}$ in (29). Otherwise, the QR method should be used in order to improve numerical stability. Such a strategy of adaptive algorithm selection allows us to achieve minimal computational time while simultaneously making the proposed solver numerically stable.

4.3 Summary

Algorithm 2: Adaptive Least-Squares Fitting

1. Start from the over-determined linear equation $A\cdot v = r$ in (20).
2. Apply the pre-conditioning scheme in (26) to calculate $\tilde{A}$.
3. Compute the Cholesky decomposition in (31) for $\tilde{A}^T\tilde{A}$.
4. Estimate the condition number $\kappa(\tilde{A})$ using (34).
5. If $\kappa(\tilde{A}) \le \mu$ (where μ is a user-defined threshold), solve the least-squares solution $\tilde{v}$ for the over-determined linear equation $\tilde{A}\cdot\tilde{v} = r$ by pseudo-inverse:
$$\tilde{v} = \tilde{P}\cdot\tilde{L}^{-1}\cdot\tilde{L}^{-T}\cdot\left(\tilde{P}^T\tilde{A}^T r\right). \quad (35)$$
6. Otherwise, if $\kappa(\tilde{A}) > \mu$, apply QR decomposition to solve the over-determined linear equation $\tilde{A}\cdot\tilde{v} = r$:
$$\tilde{A} = \tilde{Q}\cdot\tilde{W} \quad (36)$$
$$\tilde{v} = \tilde{W}^{-1}\cdot\left(\tilde{Q}^T r\right) \quad (37)$$
where $\tilde{Q} \in R^{N\times K}$ contains K orthonormal vectors and $W \in R^{K\times K}$ is an upper triangular matrix.
7. Substitute $\tilde{v}$ into (30) to solve the least-squares solution v of the over-determined linear equation $A\cdot v = r$ in (20).

Algorithm 2 summarizes the key steps for the proposed adaptive least-squares fitting that is required during each OMP iteration. Given an over-determined linear equation $A\cdot v = r$ in (20), we first normalize each column of A for pre-conditioning. Next, Cholesky decomposition is performed for the normalized matrix $\tilde{A}^T\tilde{A}$, and the condition number $\kappa(\tilde{A})$ is estimated. If $\kappa(\tilde{A})$ is sufficiently small, the least-squares solution is found by pseudo-inverse. Otherwise, QR decomposition is used to solve the least-squares fitting problem. As will be demonstrated by the numerical examples in Section 5, Algorithm 2 (with pre-conditioning and adaptive algorithm selection) achieves up to 4× speed-up over a simple solver that uses QR decomposition only.

5. NUMERICAL EXAMPLES

In this section, the efficacy of the proposed incremental power grid solver is demonstrated by several industrial circuit examples where the input supply voltage is 1.8 V for all test cases. For testing and comparison purpose, three different power grid solvers are studied: (1) algebraic multi-grid solver (AMG) [4], [6], (2) OMP with QR decomposition for least-squares fitting (OMP-QR), and (3) OMP with adaptive least-squares fitting (OMP-Adaptive). All these solvers are implemented with C++ using the sparse matrix package from University of Florida (www.cise.ufl.edu/research/sparse/). The numerical experiments are performed on a 2.8 GHz Linux server with 8 GB memory.

Table 1 summarizes the problem size of all power grid examples and their corresponding condition numbers. In these examples, the proposed pre-conditioning scheme in (26)-(28) reduces the condition number by up to 10^4×. Note that all power grid examples shown in Table 1 are well-conditioned, after pre-conditioning is applied. It is expected that the advantage of pre-conditioning would be more pronounced for the power grid circuits that are more ill-conditioned.

Table 2 compares the incremental analysis accuracy for AMG and OMP. Here, a direct solver based on LU decomposition is used to calculate the exact power grid response and estimate the error for AMG and OMP. In these examples, both OMP-QR and OMP-Adaptive result in the same accuracy and, hence, we do not distinguish these two methods in Table 2. Studying the error values in Table 2, we would find that the proposed OMP solver offers significantly improved accuracy over AMG for these incremental power grid analysis examples.

Table 1. Power grid network size and condition number

CKT	# of Nodes	Condition Number of A^TA	
		w/o Pre-conditioning	w/ Pre-conditioning
PG1	127,026	2.3×10^6	6.1×10^3
PG2	285,839	7.0×10^6	5.6×10^3
PG3	540,321	1.4×10^7	1.1×10^4
PG4	834,384	7.8×10^6	1.8×10^3
PG5	952,931	8.8×10^7	4.0×10^3

Table 2. Comparison of power grid analysis accuracy

CKT	AMG		OMP (Proposed)	
	E_{AVG} (mV)	E_{MAX} (mV)	E_{AVG} (mV)	E_{MAX} (mV)
PG1	4.40	17.70	0.120	10.10
PG2	2.10	9.01	0.026	6.75
PG3	1.60	5.00	0.002	1.16
PG4	3.80	12.4	0.003	1.10
PG5	0.07	0.44	0.001	0.07

Table 3. Comparison of power grid analysis time (Sec.)

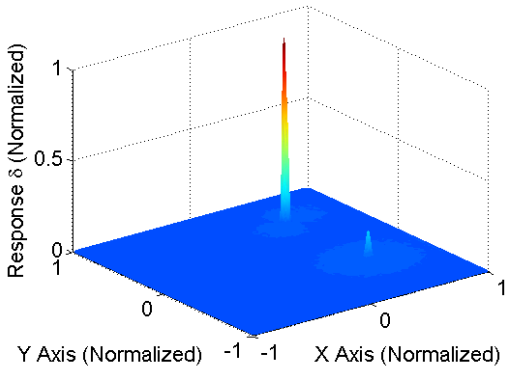
CKT	AMG	OMP-QR	OMP-Adaptive
PG1	1.69	0.056	0.037
PG2	3.55	0.200	0.048
PG3	6.63	0.262	0.155
PG4	8.34	0.108	0.086
PG5	12.9	0.144	0.099

Table 3 further compares the runtime for all three power grid solvers: AMG, OMP-QR, and OMP-Adaptive. When calculating the runtime in Table 3, we assume that the response of the original power grid system in (2) is already known and our goal is to compute the response of the updated power grid network in (3). In other words, the runtime of solving the original power grid system (2) is not counted in Table 3.

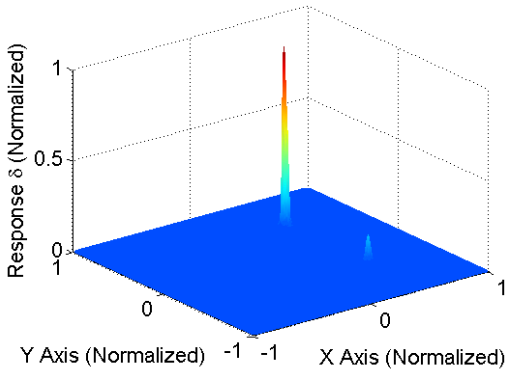
Two important observations can be made from the data in Table 3. First, the proposed OMP-Adaptive algorithm achieves up to 130× speed-up over the traditional AMG solver. Unlike AMG that considers the updated power grid network as a completely new system, OMP-Adaptive incrementally updates the power grid solution by identifying a small set of internal nodes where the response is changed. It, in turn, results in substantial runtime speed-up. Second, since OMP-Adaptive optimally applies the most efficient solver for least-squares fitting, it offers up to 4× speed-up over OMP-QR, as shown in Table 3.

Finally, Figure 1 plots the solution δ (normalized) of the incremental MNA equation in (4). The exact solution of δ is found by a direct solver based on LU decomposition and is shown in Figure 1(a). Figure 1(b) plots the solution δ computed by the proposed OMP algorithm. Note that the results in Figure 1(a) and

Figure 1(b) accurately match each other. In addition, the "incremental" response δ is extremely sparse. Namely, its value is close to zero at a large number of spatial locations. Such a sparse structure is the necessary condition to make the proposed OMP algorithm efficient for incremental power grid analysis.



(a)



(b)

Figure 1. The solution δ (normalized) of the incremental MNA equation in (4) is sparse for the largest power grid example PG5: (a) exact solution calculated by a direct solver based on LU decomposition, and (b) approximate solution calculated by the proposed OMP algorithm.

6. CONCLUSIONS

In this paper, we proposed a new incremental power grid analysis technique where an efficient Orthogonal Matching Pursuit (OMP) algorithm was adopted to solve the sparse incremental power grid response with low computational cost. In addition, several numerical techniques (i.e., pre-conditioning and adaptive algorithm selection) were developed to improve the numerical stability and reduce the computational cost of the proposed power grid solver. As was demonstrated by a number of industrial circuit examples, the proposed OMP algorithm achieves up to 130× runtime speed-up over the traditional Algebraic Multi-Grid (AMG) method without incremental analysis capability. The proposed incremental power grid solver can be further incorporated into a power grid optimization engine to facilitate efficient on-chip power grid design for nanoscale integrated circuits.

7. ACKNOWLEDGEMENTS

This work is supported in part by Mentor Graphics Corporation and the National Science Foundation under contract CCF–0811023.

8. REFERENCES

[1] T. Chen and C. Chen, "Efficient large-scale power grid analysis based on preconditioned Krylov-subspace iterative methods," *IEEE DAC*, pp. 559-562, 2001.

[2] M. Zhao, R. Panda, S. Sapatnekar and D. Blaauw, "Hierarchical analysis of power distribution networks," *IEEE Trans. CAD*, vol. 21, no. 2, pp. 159-168, Feb. 2002.

[3] J. Kozhaya, S. Nassif and F. Najm, "A multigrid-like technique for power grid analysis," *IEEE Trans. CAD*, vol. 21, no. 10, pp. 1148-1160, Oct. 2002.

[4] H. Su, E. Acar and S. Nassif, "Power grid reduction based on algebraic multigrid principles," *IEEE DAC*, pp. 109-112 2003.

[5] Z. Feng and P. Li, "Multigrid on GPU: tackling power grid analysis on parallel SIMT platforms," *IEEE ICCAD*, pp. 647-654, 2008.

[6] C. Zhuo, J. Hu, M. Zhao, and K. Chen, "Power grid analysis and optimization using algebraic multigrid," *IEEE Trans. CAD*, vol. 27, no. 4, pp. 738-751, Apr. 2008.

[7] H. Qian, S. Nassif and S. Sapatnekar, "Power grid analysis using random walks," *IEEE Trans. CAD*, vol. 24, no. 8, pp. 1204-1224, Aug. 2005.

[8] P. Li, "Statistical sampling-based parametric analysis of power grids," *IEEE Trans. CAD*, vol. 25, no. 12, pp. 2852-2867, Dec. 2006.

[9] D. Kouroussis and F. Najim, "A static pattern-independent technique for power grid voltage integrity verification," *IEEE DAC*, pp. 99-104, 2003.

[10] D. Kouroussis, I. Ferzli and F. Najm, "Incremental partitioning-based Vectorless power grid verification," *IEEE ICCAD*, pp. 358-364, 2005.

[11] H. Qian, S. Nassif and S. Sapatnekar, "Early-stage power grid analysis for uncertain working modes," *IEEE Trans. CAD*, vol. 24, no. 5, pp. 676-682, May 2005.

[12] N. Ghani and F. Najm, "Fast vectorless power grid verification using an approximation inverse technique," *IEEE DAC*, pp. 184-189, 2009.

[13] Y. Fu, R. Panda, B. Reschke, S. Sundareswarn and M. Zhao, "A novel technique for incremental analysis of on-chip power distribution networks," *IEEE ICCAD*, pp. 817-823, 2007.

[14] E. Chiprout, "Fast flip-chip power grid analysis via locality and grid shells," *IEEE ICCAD*, pp. 485-488, 2004.

[15] S. Pant and E. Chiprout, "Power grid physics and implications for CAD," *IEEE DAC*, pp. 199-204, 2006.

[16] E. Candes, "Compressive sampling," *International Congress of Mathematicians*, 2006.

[17] D. Donoho, "Compressed sensing," *IEEE Trans. Information Theory*, vol. 52, no. 4, pp. 1289-1306, 2006.

[18] J. Tropp and A. Gilbert, "Signal recovery from random measurements via orthogonal matching pursuit," *IEEE Trans. Information Theory*, vol. 53, no. 12, pp. 4655-4666, 2007.

[19] X. Li and H. Liu, "Statistical regression for efficient high-dimensional modeling of analog and mixed-signal performance variations," *IEEE DAC*, pp. 38-43, 2008.

[20] X. Li, "Finding deterministic solution from underdetermined equation: large-scale performance modeling of analog/RF circuits," *IEEE Trans. CAD*, vol. 29, no. 11, pp. 1661-1668, Nov. 2010.

[21] G. Golub and C. Loan, *Matrix Computations*, The Johns Hopkins Univ. Press, 1996.