FastSies: A Fast Stochastic Integral Equation Solver for Modeling the Rough Surface Effect [*]

Zhenhai Zhu [1], Jacob White [2]

[1] Cadence Berkeley Labs, Berkeley, CA 94704
zhzhu@cadence.com
[2] Department of Electrical Engineering and Computer Science
Massachusetts Institute of Technology, Cambridge, MA 02139
white@mit.edu

ABSTRACT

In this paper we describe several novel sparsification techniques used in a Fast Stochastic Integral Equation Solver to compute the mean value and the variance of capacitance of 3D interconnects with random surface roughness. With the combination of these numerical techniques, the computational cost has been reduced from $O(N^4)$ to $O(Nlog^2(N))$, where N is the number of panels used for the discretization of nominal smooth surfaces. Numerical experiments show that the proposed numerical techniques are accurate and efficient.

1. INTRODUCTION

The surfaces of some interconnects, particularly the off-chip ones, are far from perfectly smooth. The topological features may have peak to valley distances larger than five microns [1, 2]. It has been shown by measurements that the surface roughness can result in an increase of resistance by as much as 50% at microwave frequencies [1]. Theoretical investigation in [3, 4] correlates very well with this measurement-based observation. It is shown in [5, 6] that the capacitance is also affected significantly by surface roughness.

A straightforward approach to model surface roughness is the Monte Carlo process. An ensemble of surface realizations are generated using a height probability distribution and the height spectral density. The governing equations are then solved for each realization. Due to the fine details in the random profile of rough surfaces, a rather refined discretization has to be used. In addition, the Monte Carlo approach typically needs many thousands of solves to get statistically accurate results.

A non-Monte-Carlo method, the Stochastic Integral Equation (SIE) Method, was proposed in [7] to address these difficulties. The SIE method can compute the mean and the variance of 3D capacitance with just one solve and it only discretizes the nominal smooth surfaces. A crucial assumption, the uncorrelatedness between charge density and Green's function, is first used in [7] to avoid the classical problem of how to express the average of the

*This work was supported by the MARCO Interconnect Focus Center, the DARPA Neocad program, the Semiconductor Research Corporation, and grants from Intel and the National Science Foundation.

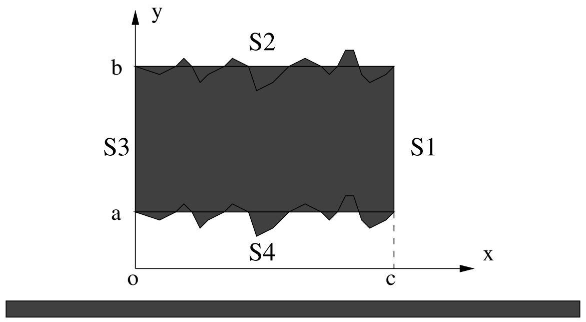

Figure 1: One conductor over a ground plane. The top and the bottom surfaces are rough and the side walls are smooth.

product of two random functions [8]. Then a second-order correction scheme based on the Taylor expansion is used to compensate for the error introduced by the uncorrelatedness assumption. However, this leads to a method with $O(N^4)$ overall computational complexity, where N is the number of panels on the nominal smooth surfaces.

In this paper, we propose several novel matrix sparsification techniques. The combination of these techniques substantially reduces the computational cost from $O(N^4)$ to $O(Nlog^2(N))$. For convenience of notation, we will use the name *Direct Stochastic Integral Equation Solver* for the method in [7], and *Fast Stochastic Integral Equation Solver* (FastSies) for the ideas presented in this paper.

2. DIRECT SIE SOLVER

In this section, we briefly review the direct Stochastic Integral Equation Solver proposed in [7]. For the sake of clarity, we use a simple 2D capacitance problem, a single conductor over a ground plane shown in figure 1, to explain the basic ideas in [7].

Taking the ensemble average on both sides of the governing equation for the capacitance extraction problem, and assuming that the charge density distribution is uncorrelated to Green's function, we obtain

$$\int_{\partial\tilde{D}_1} d\tilde{l}(x',y') \frac{<\tilde{\rho}(x',y')>}{\varepsilon_0} <\tilde{G}(x',y';x,y)>=1, \quad (x,y)\in\partial\tilde{D}_1, \tag{1}$$

where $<\tilde{G}(x',y';x,y)>$ is the ensemble average Green's function [7], and $\partial\tilde{D}_1$ is the nominal smooth surface of the conductor in figure 1. We can use a standard technique such as Galerkin to discretize (1) and obtain

$$[\bar{A}] <\tilde{\rho}^{(0)}>=\tilde{L}, \tag{2}$$

where

$$\bar{A}_{k,j} = \int_{\tilde{\Delta}_k} d\tilde{l}(x,y) \int_{\tilde{\Delta}_j} d\tilde{l}(x',y') <\tilde{G}(x',y';x,y)> \tag{3}$$

and the entries of vector $\tilde{L}$ in (2) are the size of panels, such as $\tilde{\Delta}_k$ and $\tilde{\Delta}_j$, on the nominal smooth surfaces. One of the main contributions in [7] is to demonstrate that the uncorrelatedness assumption only produces a zero-th order term in the Taylor expansion of the actual mean charge density. This is why we use $<\tilde{\rho}^{(0)}>$ instead of $<\tilde{\rho}>$ in (2). A second-order term is used in [7] to compensate for the error introduced by the uncorrelatedness assumption. Hence the mean capacitance is

$$\begin{aligned} <C> &= \tilde{L}^T <\tilde{\rho}> \simeq \tilde{L}^T(<\tilde{\rho}^{(0)}> + <\tilde{\rho}^{(2)}>) \\ &= <C^{(0)}> + <C^{(2)}>, \end{aligned} \quad (4)$$

where the zero-th order mean capacitance $<C^{(0)}>$ can be easily computed from the solution of (2). It is shown in [7] that the second-order term $<C^{(2)}>$ is

$$<C^{(2)}> \quad \simeq \quad trace(\bar{A}^{-T}B), \quad (5)$$

and the variance is

$$Var\{C\} \quad \simeq \quad <\tilde{\rho}^{(0)}> B <\tilde{\rho}^{(0)}>, \quad (6)$$

where

$$\begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_N \end{bmatrix} = \begin{bmatrix} F^{11} & \cdots & F^{1N} \\ \vdots & \ddots & \vdots \\ F^{N1} & \cdots & F^{NN} \end{bmatrix} \begin{bmatrix} <\tilde{\rho}_1^{(0)}><\tilde{\rho}^{(0)}> \\ <\tilde{\rho}_2^{(0)}><\tilde{\rho}^{(0)}> \\ \vdots \\ <\tilde{\rho}_N^{(0)}><\tilde{\rho}^{(0)}> \end{bmatrix} (7)$$

$$\begin{aligned} F_{nm}^{ij} &= <(A_{ij}-\bar{A}_{ij})(A_{mn}-\bar{A}_{mn})> \\ &= <A_{ij}A_{mn}> - \bar{A}_{ij}\bar{A}_{mn}, \quad i,j,m,n,=1,2,...,N, \end{aligned} \quad (8)$$

b_i is the i-th column of matrix B, A_{ij} and A_{mn} are the entries of the system matrix from the original governing integral equation before the ensemble average is taken in (1), and N is the number of panels used to discretize the nominal smooth surface.

Equation (8) implies that the size of matrix F is $N^2 \times N^2$. This means that calculating all b_i in (7) needs $O(N^4)$ work. In addition, since both $\bar{A}$ and B are dense, calculation of $<C^{(2)}>$ in (5) needs $O(N^3)$ work. And finally, calculating $Var\{C\}$ in (6) needs $O(N^2)$ work. All these make it very time-consuming to use the Direct Stochastic Integral Equation Solver in [7] to analyze even a simple 3D structure.

3. OUTLINE OF FASTSIES

In order to make the Stochastic Integral Equation Method efficient enough to model rough surfaces of realistic 3D structures, we have proposed three steps to reduce the overall computational complexity from $O(N^4)$ to $O(Nlog^2(N))$. The outline of these steps are enumerated in this section. The details will be given in the sections that follow.

1. Sparsification of $\bar{A}$ and $\bar{A}^{-T}$
   If we use Galerkin method to discretize the integral operator in (1), then matrix $\bar{A}$ in (2) and its inverse $\bar{A}^{-1}$ in (5) are symmetric. Hence in the remaining part of this paper, we will make no distinction between $\bar{A}^{-T}$ and $\bar{A}^{-1}$.

   We directly apply the Hierarchical Matrix method in [9, 10] to find a sparse representation for $\bar{A}$ and $\bar{A}^{-1}$ in $O(Nlog(N))$ and $O(Nlog^2(N))$ time, respectively.

2. Sparsification of matrix B and F
   We first reduce the number of nonzeros in matrix F from $O(N^4)$ to $O(N^3)$ by exploiting the fact that the correlation length on rough surfaces is much shorter than typical feature sizes of interconnects. In view of (7), this implies that the cost of computing B has been reduced from $O(N^4)$ to $O(N^3)$.

   We then propose a novel combined sampling technique inspired by the Hierarchical Matrix method to reduce the work needed for B from $O(N^3)$ to $O(Nlog^2(N))$ and obtain a Hierarchical Matrix representation of B.

3. Compute $trace(\bar{A}^{-1}B)$
   Using the Hierarchical Matrix representation of $\bar{A}^{-1}$ and B, we can compute $<C^{(2)}>$ in (5) and $Var\{C\}$ in (6) in $O(Nlog(N))$ time.

4. SPARSIFICATION OF $\bar{A}$ AND $\bar{A}^{-1}$

Though the evaluation of the ensemble average Green's function $<\tilde{G}(x',y';x,y)>$ in (1) for each pair of source point and evaluation point involves a two-fold ensemble average integral [7], it turns out that $<\tilde{G}(x',y';x,y)>$ is still a rather smooth function of (x',y') and (x,y) on the nominal smooth surfaces, just like an ordinary Green's function. Hence one can directly use existing fast integral equation solvers such as Fast Multipole [11], Hierarchical SVD [12], Precorrected FFT [13] and Hierarchical Matrix (H-matrix) method [9, 10] to calculate the zero-th order mean charge density $<\tilde{\rho}^{(0)}>$ in (2).

However, when it comes to the sparse representation of $\bar{A}^{-1}$, the options are very limited. It is shown in [9, 10] that the H-matrix method can be used to construct an accurate sparse representation of the inverse of a discretized integral operator in $O(Nlog^2(N))$ time. The accuracy can be good enough to solve equations like (2) directly. A somewhat similar idea is also used in [14]. The sparse representation of matrix inverse only has $O(Nlog(N))$ nonzeros and is derived from the sparse representation of the original matrix. It is for this reason that we have decided to use the H-matrix method to sparsify both $\bar{A}$ and $\bar{A}^{-1}$ in this paper. In the following we review one of the key steps in the H-matrix method, the low-rank decomposition. This is also used in the Hierarchical SVD method [12].

A numerically low-rank matrix Q can be written as [10, 12]

$$Q \simeq W^T V, \;\; Q \in \mathcal{R}^{M\times M}, \;\; W,V \in \mathcal{R}^{r\times M}, \quad (9)$$

where r is the approximate numerical rank of Q. A number of heuristics to construct this WV decomposition have been proposed, such as the rank-revealing QR decomposition in [12] and the adaptive low-rank approximation in [15]. We do not intend to repeat these two algorithms here. But it suffices to point out that both methods need access to r columns and r rows of matrix Q in (9) and both methods have been claimed to access only $O(Nlog(N))$ entries in a hierarchically low-rank matrix of size $N \times N$ in order to construct its sparse representation.

5. SPARSIFICATION OF MATRIX B AND F

We have made two basic assumptions about the profile of rough surfaces.

1. There is no over-hang, i.e., the height fluctuation is a single-value function of the location on the nominal smooth surface. We believe this assumption is very reasonable given the surface roughness is mainly caused by electro-deposition [1].

2. The profile is described by the stationary Gaussian stochastic process. To our best knowledge, little data is available to determine the most appropriate mathematical models for rough surfaces of 3D interconnects. The stationary Gaussian stochastic process has been used extensively in rough surface scattering [16] perhaps for its mathematical simplicity.

5.1 Exploit the Short Correlation Length

We again use the simple 2D example shown in figure 1 to explain the basic idea. As it will become clear soon, this idea can be easily extended to 3D cases.

The joint p.d.f of stationary Gaussian stochastic processes is the joint normal, and its Gaussian correlation function is [17]

$$C(\xi) = exp(-\frac{\xi^2}{\eta^2}), \tag{10}$$

where η is correlation length. In this section we show how to sparsify matrix F by using the fact that the correlation function drops exponentially fast with the increase of distance between two points on the same rough surface.

There are four panels involved in the expression for F^{ij}_{mn} in (8). Here we focus on the case where these panels are all on the top side of the conductor in figure 1 and hence the height fluctuation of all four panels is along the y direction. Other cases can be treated similarly. Hence we have

$$\begin{aligned} < A_{ij}A_{mn} > \quad = \quad & \int_{\tilde{\Delta}_i} dx_i \int_{\tilde{\Delta}_j} dx_j \int_{\tilde{\Delta}_m} dx_m \int_{\tilde{\Delta}_n} dx_n \\ & \int_{-\infty}^{+\infty} dh_i \int_{-\infty}^{+\infty} dh_j \int_{-\infty}^{+\infty} dh_m \int_{-\infty}^{+\infty} dh_n \\ & P_4(h_i,h_j,h_m,h_n;x_i,x_j,x_m,x_n) \\ & G(x_j,b+h_j;x_i,b+h_i)G(x_n,b+h_n;x_m,b+h_m) \\ \simeq \quad & \int_{-\infty}^{+\infty} dh_i \int_{-\infty}^{+\infty} dh_j \int_{-\infty}^{+\infty} dh_m \int_{-\infty}^{+\infty} dh_n \\ & P_4(h_i,h_j,h_m,h_n;\bar{x}_i,\bar{x}_j,\bar{x}_m,\bar{x}_n)A_{ij}A_{mn}, \end{aligned} \tag{11}$$

where $(\bar{x}_i,b)$,$(\bar{x}_j,b)$,$(\bar{x}_m,b)$,$(\bar{x}_n,b)$ are the nominal position of the centroids of panels i,j,m and n, and h_i,h_j,h_m,h_n are the height fluctuations. The approximate equality is due to the change of order in panel integration and ensemble average integration. Function P_4 is the joint Gaussian distribution. To facilitate the following discussion, we define respectively set $near(i)$ and set $far(i)$ as

$$near(i) = \{panel \;\; k \mid \; |\bar{x}_i - \bar{x}_k| \leq 3\eta\} \tag{12}$$

$$far(i) = \{panel \;\; k \mid \; |\bar{x}_i - \bar{x}_k| > 3\eta\}. \tag{13}$$

where $\bar{x}_i$ and $\bar{x}_k$ are respectively the centroids of panel i and k. It is straightforward to show that

$$j \in near(i) \Rightarrow i \in near(j) \tag{14}$$

$$j \in far(i) \Rightarrow i \in far(j). \tag{15}$$

In addition, it is also easy to show that

$$n \in near(j), i \in far(j) \Rightarrow i \in far(n) \tag{16}$$

is almost always true. In view of (10), we have

$$\begin{aligned} & j \in far\{i\} \Rightarrow |\bar{x}_i - \bar{x}_j| \geq 3\eta \\ \Rightarrow \quad & C(|\bar{x}_i - \bar{x}_j|) \leq 1.24 \times 10^{-4} \Rightarrow C(|\bar{x}_i - \bar{x}_j|) \simeq 0. \end{aligned} \tag{17}$$

Therefore,

$$\left\{ \begin{array}{l} m \in far\{i\} \\ n \in far\{i\} \\ m \in far\{j\} \\ n \in far\{j\} \end{array} \right. \Rightarrow \left\{ \begin{array}{l} C(|\bar{x}_i - \bar{x}_m|) \simeq 0 \\ C(|\bar{x}_i - \bar{x}_n|) \simeq 0 \\ C(|\bar{x}_j - \bar{x}_m|) \simeq 0 \\ C(|\bar{x}_j - \bar{x}_n|) \simeq 0 \end{array} \right. . \tag{18}$$

In other words, the height fluctuations h_i and h_j are approximately uncorrelated to h_m and h_n. Since their joint p.d.f P_4 is Gaussian, h_i and h_j are approximately independent of h_m and h_n. Hence A_{ij}, a function of h_i and h_j, is approximately independent of A_{mn}, a function of h_m and h_n [17]. Therefore, we have $< A_{ij}A_{mn} > \simeq < A_{ij} >< A_{mn} > = \bar{A}_{ij}\bar{A}_{mn}$ and $F^{ij}_{nm} \simeq 0$ immediately follows from (8).

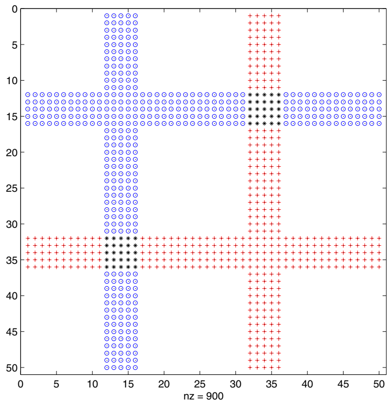

Figure 2: Typical sparsity pattern of the sparsified matrix F^{ij}. Here the total number of panels is 50, $i = 34$ and $j = 14$, and a rough surface segment of 3η long contains $p = 2$ panels. The nonzero entries are categorized into three regions marked by $+$ (region I), o (region II) and $*$ (region III), respectively.

It should be pointed out that (18) would be exact if panel pair i and j are on the top surface of the conductor in figure 1 and panel pair m and n are on the bottom surface, or if they are on different conductor surfaces in the case of multiple conductors. For 3D interconnects with multiple conductors, this is expected to be the majority case. Hence the error introduced by this scheme is quite insignificant.

We assume that a rough surface segment of 3η long contains at most p panels. Since η is independent of the total number of panels, so is p. The approximation in (18) implies that there are at most $4p+2$ non-zero rows and columns in each sparsified F^{ij}. A typical sparsity pattern of the sparsified matrix F^{ij} is shown in figure 2, where it is assumed that the indexes of the spatially close panels are also close to each other. The natural panel ordering for 2D structures always satisfies this. However, a panel ordering for 3D surfaces may not. But this does not change the fact that matrix F^{ij} can be approximated by a sparse matrix. Now it is clear that the sparsified matrix block F^{ij} has $O(N)$ non-zero entries and the total number of non-zero entries in the sparsified matrix F is $O(N^3)$.

5.2 Combined Sampling to Sparsify B

As will be shown in section 6, a sparse representation of matrix B is essential to efficiently compute $trace(\bar{A}^{-1}B)$. However, finding this representation is unconventional because matrix B is not from a discretized integral operator, as shown in (7) and (8). It is shown in the appendix that the matrix entry B_{im} scales approximately as $\frac{1}{r_{im}}$ when $m \in far(i)$, where r_{im} is the distance between panels i and m. This implies that it is possible to use either the H-matrix [10] or the hierarchical SVD [12] to sparsify matrix B. In view of (7), each column vector b_i is

$$\begin{aligned} b_i \quad &= \quad \sum_{j=1}^{N} F^{ij} < \rho^{(0)} >< \rho_j^{(0)} > \\ &= \quad \left[\begin{array}{cccc} f^{i1} & f^{i2} & \cdots & f^{iN} \end{array} \right] < \rho^{(0)} >, \\ &= \quad [M^{(i)}] < \rho^{(0)} > \end{aligned} \tag{19}$$

where

$$f^{ij} = F^{ij} < \rho^{(0)} >, \quad j = 1,2,\cdots,N. \tag{20}$$

As will be shown in the appendix, matrix entry $M^{(i)}_{mj}$ also scales approximately as $\frac{1}{r_{mj}}$ when $m \in far(j)$. So we can construct the H-matrix for $M^{(i)}$ and calculate $M^{(i)} < \rho^{(0)} >$ in $O(Nlog(N))$ time. But in constructing the H-matrix for B, usually only a small fraction of b_i is sampled. Unfortunately, calculation of even a small fraction of b_i still involves H-matrix construction for the whole matrix $M^{(i)}$, which takes $O(Nlog(N))$ time. [1] Hence the total time would be $O(N^2 log(N))$ if the H-matrix of the whole matrix $M^{(i)}$ is constructed for $i = 1, 2, ..., N$.

In this section, we propose a so-called combined sampling process to substantially reduce this cost. The key idea is to combine small segments of vector b_i for different i into a long vector and compute this long vector in one go. In order to focus our attention on the main ideas, we deliberately use a trivially simple example in this section to explain this combined sampling process. As it turns out, the algorithm directly extends to more general cases.

5.2.1 *Standard sampling process*

Assume that matrix B has a two-level H-matrix representation shown in figure 3, where $R_i (i = 1, 2, ..., 6)$ are low-rank matrices and $D_i (i = 1, 2, 3, 4)$ are full-rank matrices. Further more, assume that the size of matrix B is $N \times N = 32 \times 32$ and the rank of all R_i is the same $r = 2$. Consequently, the size of matrix D_i is 8×8. It should be pointed out that the actual values of N and r are unimportant, we use them here just for convenience of notation.

Each low-rank block R_i in figure 3 is decomposed in the WV form shown in (9). It should be noted that this decomposition is not unique, one can scale W with an arbitrary factor α and scale V with $\frac{1}{\alpha}$. For simplicity, we assume here that both W and V have been normalized such that the decomposition is unique. It is easy to check that

$$\begin{cases} Q_1 = Q_2^T \\ Q_1 = W_1^T V_1 \\ Q_2 = W_2^T V_2 \end{cases} \Rightarrow \begin{cases} W_1 = V_2 \\ V_1 = W_2. \end{cases} \tag{21}$$

In view of (7) and (8), it is relatively straightforward to check that matrix B is symmetric. Therefore, the low-rank blocks in figure 3 satisfy $R_1 = R_2^T$, $R_3 = R_4^T$ and $R_5 = R_6^T$. Let the WV decomposition of $R_i (i = 1, 2, ..., 6)$ be

$$R_i = W_i^T V_i,\ R_i \in \mathcal{R}^{8\times 8},\ W_i, V_i \in \mathcal{R}^{2\times 8},\ \ i = 1, 2, 3, 4. \tag{22}$$

$$R_i = W_i^T V_i,\ \ R_i \in \mathcal{R}^{16\times 16},\ \ W_i, V_i \in \mathcal{R}^{2\times 16},\ \ i = 5, 6. \tag{23}$$

In view of (21), we have

$$V_1 = W_2, V_2 = W_1 \tag{24}$$

$$V_3 = W_4, V_4 = W_3 \tag{25}$$

$$V_5 = W_6, V_6 = W_5. \tag{26}$$

Hence we only need matrices $W_i (i = 1, 2, ..., 6)$. Matrices $V_i (i = 1, 2, ..., 6)$ are redundant.

[1]Suppose we want to compute $y = Ax$, where A corresponds to a discretized integral operator with $\frac{1}{r}$ kernel. It is interesting that all well-known fast solver algorithms can not do better than $O(N)$ when only one entry of y is needed, despite the fact that all these algorithms can compute the whole vector y with N entries in $O(N)$ or $O(Nlog(N))$ time. The multipole expansion in Fast Multipole Method, sampling in hierarchical SVD, panel clustering in H-matrix method, and direct matrix setup in Pre-corrected FFT method involve all N source panels, regardless of how many entries in y are to be computed. This initialization cost is amortized over all N entries of y. But if only a small number of entries in y is needed, then the overhead of initial setup is as expensive as direct calculation of those few entries in y.

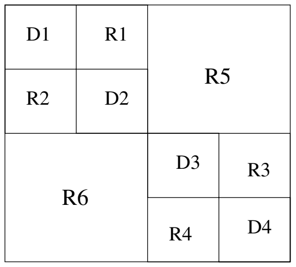

Figure 3: Two-level H-matrix representation of matrix B. R_i is the low-rank matrix and D_i is the full-rank matrix.

5.2.2 *Combined sampling process*

In order to obtain each D_i, we need to compute 8 column vectors in B. Hence to get all $D_i (i = 1, 2, 3, 4)$, we effectively have to compute the whole matrix B. Consider putting D_i into a big matrix as follow

$$T_1 = \begin{bmatrix} D_1 \\ D_2 \\ D_3 \\ D_4 \end{bmatrix} = \begin{bmatrix} \tilde{b}_1 & \tilde{b}_2 & \cdots & \tilde{b}_8 \\ \tilde{b}_9 & \tilde{b}_{10} & \cdots & \tilde{b}_{16} \\ \tilde{b}_{17} & \tilde{b}_{18} & \cdots & \tilde{b}_{24} \\ \tilde{b}_{25} & \tilde{b}_{26} & \cdots & \tilde{b}_{32} \end{bmatrix}, \tag{27}$$

where $\tilde{b}_i$ is one quarter of b_i with suitably selected entries. For example, $\tilde{b}_1 = b_1(1:8)$, $\tilde{b}_9 = b_9(9:16)$, $\tilde{b}_{17} = b_{17}(17:24)$ and $\tilde{b}_{25} = b_{25}(25:32)$. Here, we have used Matlab matrix notation. Clearly, the size of matrix T_1 is 32×8. Therefore, instead of sampling the whole matrix B, we only need to compute 8 column vectors in T_1.

As explained before, to obtain the low-rank representation of R_i, we need to sample two column vectors in B. Consider putting $R_i (i = 1, 2, 3, 4)$ into a big matrix as

$$T_2 = \begin{bmatrix} R_1 \\ R_2 \\ R_3 \\ R_4 \end{bmatrix} = \begin{bmatrix} W_1^T V_1 \\ W_2^T V_2 \\ W_3^T V_3 \\ W_4^T V_4 \end{bmatrix}. \tag{28}$$

To obtain $W_i (i = 1, 2, 3, 4)$, we just need to compute

$$S_1 = \begin{bmatrix} \tilde{b}_9 & \tilde{b}_{10} \\ \tilde{b}_1 & \tilde{b}_2 \\ \tilde{b}_{25} & \tilde{b}_{26} \\ \tilde{b}_{17} & \tilde{b}_{18} \end{bmatrix}, \tag{29}$$

where $\tilde{b}_i$ is again one quarter of b_i with suitably selected entries. The sampling column indexes such as 9,10,1,2 and so on should be carefully picked using the algorithms in [12, 15]. These specific numbers are used here just for illustration purpose. Therefore, instead of sampling a total of 8 column vectors in matrix B, we only need to compute 2 column vectors in S_1.

Finally, to obtain the low-rank representation of $R_i (i = 5, 6)$, we need to sample two column vectors in B. Consider putting $R_i (i = 5, 6)$ into a big matrix as

$$T_3 = \begin{bmatrix} R_5 \\ R_6 \end{bmatrix} = \begin{bmatrix} W_5^T V_5 \\ W_6^T V_6 \end{bmatrix}. \tag{30}$$

To obtain $W_i (i = 5, 6)$, we just need to compute

$$S_2 = \begin{bmatrix} \tilde{b}_{17} & \tilde{b}_{18} \\ \tilde{b}_1 & \tilde{b}_2 \end{bmatrix}, \tag{31}$$

where $\tilde{b}_i$ is one half of b_i with suitably selected entries. Therefore, instead of sampling a total of 4 column vectors in matrix B, we only need to compute 2 column vectors in S_2.

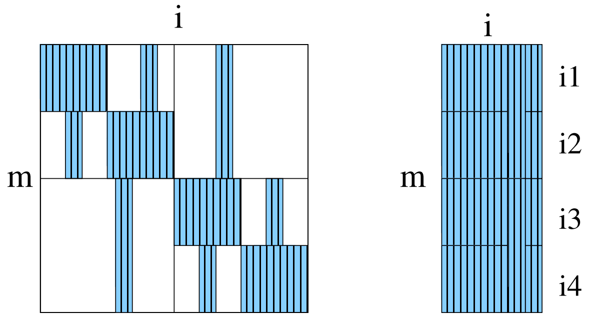

Figure 4: Location of sampled columns in matrix B and their compressed-row format. Due to symmetry of B, only columns need to be sampled.

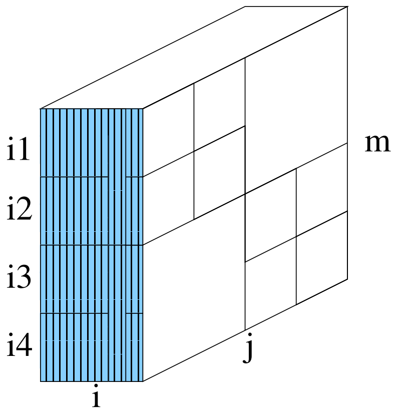

Figure 5: Relation between column sampling for B (on $m-i$ plane) and the hierarchical structure of $\tilde{M}$ (on $m-j$ plane). Each slice on $m-j$ plane represents one matrix $\tilde{M}$ and is to be multiplied with $<\rho^{(0)}>$ to obtain the corresponding sampled column on $m-i$ plane . The number of slices is equal to the number of sampled columns in figure 4.

In view of (19), the combined sampling in (27), (29) and (31) boils down to computing a combined vector in the form of

$$\tilde{b}=\begin{bmatrix}\tilde{b}_\alpha\\ \tilde{b}_\beta\\ \vdots\\ \tilde{b}_\gamma\end{bmatrix}=\begin{bmatrix}\tilde{M}^{(\alpha)}\\ \tilde{M}^{(\beta)}\\ \vdots\\ \tilde{M}^{(\gamma)}\end{bmatrix}<\rho^{(0)}>=\tilde{M}<\tilde{\rho}^{(0)}>, \qquad (32)$$

where α, β and γ represent the column indexes in (27), (29) and (31), and $\tilde{M}^{(\alpha)}$, $\tilde{M}^{(\beta)}$ and $\tilde{M}^{(\gamma)}$ are respectively a part of $M^{(\alpha)}$, $M^{(\beta)}$ and $M^{(\gamma)}$ defined in (7) with suitably selected rows. As will be shown in the appendix, matrix entry $\tilde{M}_{mj}$ scales approximately as $\frac{1}{r_{mj}}$ when $m\in far(j)$, just like matrix entry $M^{(i)}_{mj}$. Hence the H-matrix method can be used to calculate $\tilde{b}$ in $O(Nlog(N))$ time. This immediately implies that the low-rank representation for R_i in (22) and (23) and the small full-rank matrices D_i in (27) can be computed efficiently.

5.2.3 A graphical interpretation of the combined sampling process

In summary, in order to construct the two-level H-matrix of B in figure 3, we need to compute T_1 in (27), S_1 in (29) and S_2 in (31), a total of 12 column vectors. Each one of these 12 column vectors involves the H-matrix construction of a matrix $\tilde{M}$. It should be noted

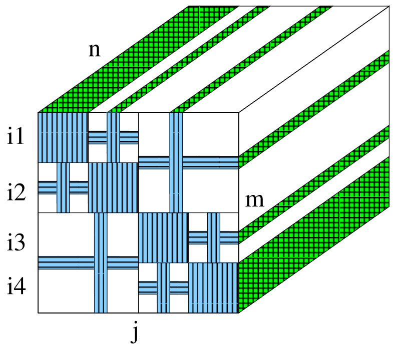

Figure 6: Location of sampled columns and rows in matrix $\tilde{M}$ (front $m-j$ plane) and their relation to entries in matrix F^{ij}. Each "thread" along index n direction represents one row in matrix F^{ij}. The entry $\tilde{M}_{mj}$ in the shaded region on the $m-j$ plane is the result of inner product between the "thread" and $<\rho^{(0)}>$. Notice that most of the "threads" only have a small number of nonzeros, as implied by figure 2.

that matrix $\tilde{M}$ is asymmetric. Hence unlike treatment of matrix B in section *5.2.1*, we have to access 12 columns and 12 rows in $\tilde{M}$ to construct its H-matrix.

The column sampling of matrix B is shown in figure 4. Figure 5 shows the relation between column sampling for B and the H-matrix of $\tilde{M}$. And figure 6 shows the sampled entries in matrix $\tilde{M}$ and their relation to the entries of matrix F^{ij}. These are three main components in the combined sampling process.

By definition, the number of non-zeros in figure 4 is equal to the number of sampling entries in the Hierarchical SVD or H-matrix. It has been shown that this number is $O(Nlog(N))$ [12, 10]. Since there is no empty row in figure 4 and each row has roughly the same number of non-zeros, the average number of nonzeros per row or the number of columns in the compressed-row format in figure 4 is $O(log(N))$. This is the total number of matrices $\tilde{M}$ in figure 6. Construction of the H-matrix representation for each $\tilde{M}$ as well as carrying out the matrix-vector product $M^{(i)}<\rho^{(0)}>$ in (19) needs $O(Nlog(N))$ work. Hence the total work is $O(Nlog^2N)$.

6. COMPUTING $TRACE(\bar{A}^{-1}B)$

Having found the H-matrix representation for matrix $\bar{A}^{-1}$ and B, we are ready to present an efficient algorithm to compute $trace(\bar{A}^{-1}B)$. As explained in [10], the clustering tree for $\bar{A}^{-1}$ is derived from that for $\bar{A}$. Because both $\bar{A}$ and B are derived from the same set of panels on the same surfaces and the off-diagonal entries of both matrices scale as $\frac{1}{r}$, it is safe to assume that the panel clustering trees for them are the same. Due to block matrix manipulation such as add, multiplication and rank-k truncation, the clustering tree for $\bar{A}^{-1}$ can be different from that of $\bar{A}$ at leaf level, though most of the two trees should be the same. Hence matrix $\bar{A}^{-1}$ and B may or may not have the same leaf blocks. We will cover both cases in this section.

Block partition We will first present a generic approach. Suppose we want to compute $trace(PQ)$ where $P,Q\in\mathcal{R}^{N\times N}$ and their entries are arbitrary. Using a binary block partition, we have

$$\begin{aligned}trace(PQ) &= trace(\begin{bmatrix}P_{11} & P_{12}\\ P_{21} & P_{22}\end{bmatrix}\begin{bmatrix}Q_{11} & Q_{12}\\ Q_{21} & Q_{22}\end{bmatrix})\\ &= trace(\begin{bmatrix}P_{11}Q_{11}+P_{12}Q_{21} & *\\ * & P_{21}Q_{12}+P_{22}Q_{22}\end{bmatrix})\\ &= trace(P_{11}Q_{11})+trace(P_{12}Q_{21})\\ &\quad +trace(P_{21}Q_{12})+trace(P_{22}Q_{22}). \qquad (33)\end{aligned}$$

If we keep on recursively partitioning in this fashion, we can compute the result in $O(N^2)$ time. This, of course, is the same as when we compute the *trace* directly. But we can do much better this way if both P and Q have low-rank representations. Let P_s and Q_s be one of the sub-blocks at a certain level down to the partition tree, just like P_{11} and Q_{11} in (33).

Low-rank matrix block times low-rank matrix block Assume we have a low-rank representation for both P_s and Q_s

$$P_s = U_P V_P^T, \ P_s \in \mathcal{R}^{M\times M}, \ U_P, V_P \in \mathcal{R}^{M\times k}$$
$$Q_s = U_Q V_Q^T, \ Q_s \in \mathcal{R}^{M\times M}, \ U_Q, V_Q \in \mathcal{R}^{M\times k}, \quad (34)$$

where M is the size of blocks P_s and Q_s and k is the approximate rank of P_s and Q_s, then we have

$$trace(P_sQ_s) = trace([U_P(V_P^T U_Q)]V_Q^T). \quad (35)$$

If we compute the *trace* in the order enforced by the parenthesis in (35), then it only takes $2Mk^2 + Mk$ work. This is comparable to the cost of a matrix-vector product, $2Mk$.

Low-rank matrix block times full-rank matrix block Assume P_s is in low-rank representation as in (34) and Q_s has full rank, then we have

$$trace(P_sQ_s) = trace(U_P(V_P^T Q)). \quad (36)$$

It is easy to check that computing $trace(P_sQ_s)$ takes $M^2k + Mk$ work. This is comparable to the cost of full-rank matrix-vector product, M^2. The same is true when the roles of P_s and Q_s are switched.

Full-rank matrix block times full-rank matrix block If both P_s and Q_s have full rank, the calculation of $trace(P_sQ_s)$ takes M^2 work, same as the cost of full-rank matrix-vector product, M^2.

Total cost It is shown in [9, 10] that the matrix vector product takes $O(Nlog(N))$ work if the matrix is in H-Matrix form. As shown above, the cost of computing *trace* is different from that of matrix vector product by about a factor k. Since $k << M$ in H-matrix form and k is a constant independent of M, computing the *trace* also takes $O(Nlog(N))$ work.

7. NUMERICAL RESULTS

In this section we use numerical experiments to separately test the validity of the approximations made in each key step outlined in section 3. All examples are run on a Pentium IV desktop PC with 2GB memory.

A) Sparsification of $\bar{A}$ and $\bar{A}^{-1}$ We can compute the zero-th order mean capacitance by either solving (2) iteratively using the H-matrix of $\bar{A}$ or by computing $<\tilde{\rho}^{(0)}>=\bar{A}^{-1}\tilde{L}$ using the H-matrix of $\bar{A}^{-1}$. To test the accuracy, we first analyze a small $1\times 1mm$ 3D plate over a ground plane and compare the results in table 1 to those in [7]. The good agreement confirms the accuracy of the H-matrix construction for both $\bar{A}$ and $\bar{A}^{-1}$. For larger 3D plates of size $5\times 5mm$, $10\times 10mm$, $15\times 15mm$ and $20\times 20mm$, the direct SIE method in [7] becomes infeasible because it consumes more than 2GB memory. So we only compare the results obtained from solving (2) iteratively and computing $<\tilde{\rho}^{(0)}>=\bar{A}^{-1}\tilde{L}$. This is shown in table 2. The good agreement further confirms the accuracy of the H-matrix for $\bar{A}^{-1}$ and hence it can be reliably used to compute *trace* in (5).

The CPU time used for the sparsification of $\bar{A}$ and $\bar{A}^{-1}$ are plotted against $Nlog(N)$ and $Nlog^2(N)$ curves respectively in figure 7. Clearly the CPU time grows as expected. It is worth noting that the extra CPU time for constructing the H-matrix of $\bar{A}^{-1}$ is only a small fraction of the CPU time for constructing the H-matrix of $\bar{A}$.

We can certainly use Monte Carlo method to compute the mean capacitance of the large 3D plates used in table 2. But due to fine details on the rough surfaces, we have to use at least 8 panels per

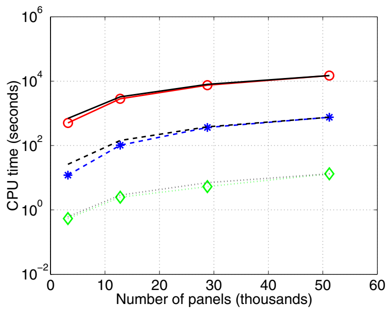

Figure 7: CPU time for sparsification of $\bar{A}$ (solid line with o) and $\bar{A}^{-1}$ (dashed line with *), and for computing trace($\bar{A}\bar{A}^{-1}$) (dotted line with diamond). The solid line and dotted line without any symbol are the $Nlog(N)$ curves normalized differently. The dashed line without any symbol is the $Nlog^2(N)$ curve.

Table 1: Zero-th order mean capacitance of a $1\times 1mm$ 3D plate over a ground plane. Unit:pF

η	σ	SIE I [7]	solve (2) iteratively	$<\tilde{\rho}^{(0)}>=\bar{A}^{-1}\tilde{L}$
0.1	0.1	63.850	63.899	63.860

correlation length. Assuming four thousand Monte Carlo simulations are necessary to obtain statistically accurate results, none of the large examples in table 2 can be completed in less than one week, even if the H-matrix method is used as its core engine. It is clear that *FastSies* is much more efficient than Monte Carlo method.

B) Exploit short correlation length We have analyzed the same 3D plate example in table 1 using the $O(N^3)$ algorithm covered in section 5.1. This gives the more accurate mean capacitance with the second-order term $< C^{(2)} >$ in (4). The results are compared in table 3 to those obtained using the $O(N^4)$ algorithm presented in [7] . It is clear that the approximation in 5.1 section introduces very little error.

C) B is hierarchically low rank We use a circular wire over a ground plane example directly from [7]. The singular values of the 4 sub-blocks of the matrix B are shown in figure 8. Furthermore, the singular values of the 4 sub-blocks of the matrix block B_{11}, a sub-block of B, are shown in figure 9. It is clear that matrix B is hierarchically low rank.

D) $\tilde{M}$ is hierarchically low rank We use the same circular wire over a ground plane example as the previous example. As shown in (32), the combined matrix $\tilde{M}$ consists of rows from different $M^{(i)}$. We have filled three different combined matrices

$$\tilde{M}_1 = \begin{bmatrix} \tilde{M}^{(1)} \\ \tilde{M}^{(17)} \\ \tilde{M}^{(33)} \\ \tilde{M}^{(49)} \end{bmatrix} \quad \tilde{M}_2 = \begin{bmatrix} \tilde{M}^{(2)} \\ \tilde{M}^{(18)} \\ \tilde{M}^{(34)} \\ \tilde{M}^{(50)} \end{bmatrix} \quad \tilde{M}_3 = \begin{bmatrix} \tilde{M}^{(1)} \\ \tilde{M}^{(2)} \\ \vdots \\ \tilde{M}^{(64)} \end{bmatrix}, \quad (37)$$

where $\tilde{M}_3$ is an extreme case in the sense that each of its rows is

Table 2: Zero-th order mean capacitance of a few large 3D plates over a ground plane. Unit:pF. $\eta = 0.1mm$ and $\sigma = 0.1mm$.

size	#panel	solve (2) iteratively	$<\tilde{\rho}^{(0)}>=\bar{A}^{-1}\tilde{L}$
5×5	3200	749.604	749.603
10×10	12800	2508.42	2508.44
15×15	28800	5246.21	5246.42
20×20	51200	8953.56	8954.01

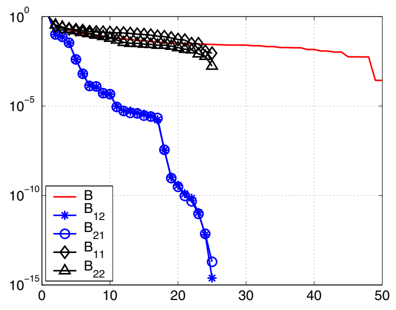

Figure 8: Distribution of the singular values of the four sub-blocks of the matrix B, circular wire over a ground plane.

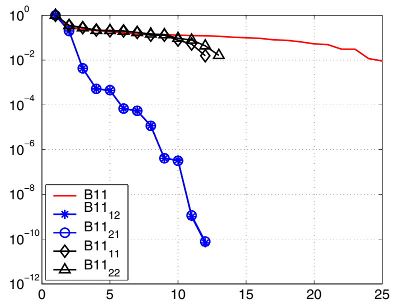

Figure 9: Distribution of the singular values of the four sub-blocks of the matrix B_{11}, circular wire over a ground plane.

taken from a different $\tilde{M}^{(i)}, i = 1, 2, ..., 64$. The singular values of the 4 sub-blocks of matrix M_3 are shown in figure 10. The singular values of the 4 sub-blocks of M_3^{11}, one of the diagonal blocks of M_3, are shown in figure 11. Clearly, matrix $\tilde{M}_3$ is hierarchically low-rank. Though not shown here, the same is true for matrices $\tilde{M}_1$ and $\tilde{M}_2$.

E) Compute $trace$ **in** $O(Nlog(N))$ **time** A reliable way to check the algorithm in section 6 is to use it to compute $trace(\bar{A}\bar{A}^{-1})$. The expected result is obviously the size of matrix $\bar{A}$. Table 4 shows the computational results and the CPU time is plotted in figure 7 against $Nlog(N)$ curve. Clearly the algorithm is very accurate and its CPU time grows as $O(Nlog(N))$.

8. CONCLUSIONS AND FUTURE WORK

We have proposed three key steps to reduce the computational complexity of a previously proposed direct stochastic integral equation solver from $O(N^4)$ to $O(Nlog^2(N))$. The accuracy and CPU time of the key elements in each step have been individually validated by carefully designed numerical experiments. These experiments clearly show that the ideas presented in this paper can enable

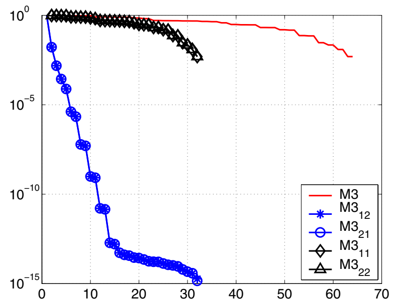

Figure 10: Distribution of the singular values of the four sub-blocks of the matrix $\tilde{M}_3$.

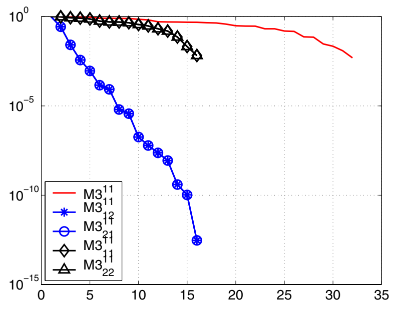

Figure 11: Distribution of the singular values of the four sub-blocks of the matrix block $\tilde{M}_3^{11}$.

Table 3: Up to second-order mean capacitance of 3D plate calculated with or without the approximation in section 5.1. Unit:pF.

η	σ	Monte Carlo	with	without [7]
0.2	0.1	62.656(4000run)	62.87	62.706
0.1	0.1	66.237(4000run)	65.95	65.471

the new fast stochastic integral equation solver (FastSies) to attack large and realistic problems.

Due to the close connection between the numerical techniques for capacitance extraction and impedance extraction, as demonstrated in [18, 12], we believe that the FastSies will be a very useful tool in addressing the more complicated impedance extraction problems in the presence of conductor surface roughness.

9. REFERENCES

[1] H. Tanaka and F. Okada, "Precise measurements of dissipation factor in microwave printed circuit boards," *IEEE Transactions on Instruments and Measurement*, vol. 38, no. 2, pp. 509–514, 1989.

[2] S. Chi and A. Agrawal, "Fine line thin dielectric circuit board characterization," *Proceedings of* 44th *Electronic Components and Technology Conference*, pp. 564–569, 1-4 May 1994.

[3] S. Morgan, "Effect of surface roughness on Eddy current losses at microwave frequencies," *Journal of Applied Physics*, vol. 20, pp. 352–362, 1949.

[4] C. Holloway and E. Kuester, "Power loss associated with conducting and supperconducting rough surfaces," *IEEE Transactions on Microwave Theory and Techniques*, vol. 48, no. 10, pp. 1601–1610, 2000.

[5] L. Young, *Advances in Microwaves*. New York: Academic Press, 1971.

[6] R. Patrikar, C. Dong, and W. Zhuang, "Modelling interconnects with surface roughness," *MICROELECTRONICS JOURNAL*, vol. 33, no. 11, pp. 929–934, 2002.

[7] Z. Zhu, A. Demir, and J. K. White, "A stochastic integral equation method for modeling the rough surface effect on interconnect capacitance," *International Conference on Computer Aided-Design*, pp. 887–891, 2004.

[8] G. Brown, "A stochastic Fourier transform approach to scattering from perfectly conducting randomly rough surfaces," *IEEE Transactions on Antennas and Propagation*, vol. 30, no. 6, pp. 1135–1143, November 1982.

[9] W. Hackbusch, "A sparse matrix arithmetic based on H-matrices. part

Table 4: Calculated $trace(\bar{A}\bar{A}^{-1})$ for different matrix sizes

Matrix size	$trace(\bar{A}\bar{A}^{-1})$
3200	3200.000065
12800	12800.000012
28800	28800.00045
51200	51199.99985

II: Application to multi-dimensional problems," *Computing*, vol. 64, pp. 21–47, 2000.

[10] S. Borm, L. Grasedyck, and W. Hackbusch, "Hierarchical matrices," *Lecture note available at www.hmatrix.org/literature.html*, 2003.

[11] L. Greengard, *The Rapid Evaluation of Potential Fields in Particle Systems*. Cambridge, Massachusetts: M.I.T. Press, 1988.

[12] S. Kapur and D. Long, "IES3: A fast integral equation solver for efficient 3-dimensional extraction," *International Conference on Computer Aided-Design*, pp. 448–455, 1997.

[13] J. R. Phillips and J. K. White, "A precorrected-FFT method for electrostatic analysis of complicated 3D structures," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 16, pp. 1059–1072, 1997.

[14] D. Gope and V. Jandhyala, "An iteration-free fast multilevel solver for dense method of moment systems," *10th Topical Meeting on Electrical Performance of Electronic Packaging*, Oct. 2001, Boston, MA.

[15] M. Bebendorf and S. Rjasanow, "Adaptive low-rank approximation of collocation matrices," *Computing*, vol. 70, pp. 1–24, 2003.

[16] L. Tsang, J. Kong, K. Ding, and C. Ao, *Scattering of electromagnetic waves: numerical simulations*. New York: John Wiley and Sons, Inc., 2001.

[17] A. Papoulis, *Probability, random variables, and stochastic processes*. New York: McGraw-Hill Inc., 1965.

[18] M. Kamon, M. J. Tsuk, and J. White, "FastHenry: A multipole-accelerated 3-D inductance extraction program," *IEEE Transactions on Microwave Theory and Techniques*, vol. 42, no. 9, pp. 1750–1758, September 1994.

APPENDIX

In this appendix, we show that the matrix entry B_{im}, or $b_i(m)$ in (19), scales as $\frac{1}{r_{mi}}$ when $m \in far(i)$, and $\tilde{M}_{mj}$ in (32) scales as $\frac{1}{r_{mj}}$ when $m \in far(j)$.

The nonzero entries in figure 2 are categorized into three regions marked by $+$ (region I), o (region II) and $*$ (region III), respectively. These regions are

1. Region I: $\{$ Horizontal sub-region $|$ $m \in near(i)$ and $n \in far(j)$ $\}$ or $\{$ Vertical sub-region $|$ $m \in far(j)$ and $n \in near(i)$ $\}$
2. Region II: $\{$ Horizontal sub-region $|$ $m \in near(j)$ and $n \in far(i)$ $\}$ or $\{$ Vertical sub-region $|$ $m \in far(i)$ and $n \in near(j)$ $\}$
3. Region III: $\{$ $m \in near(j)$ and $n \in near(i)$ $\}$ or $\{$ $m \in near(i)$ and $n \in near(j)$ $\}$

For a fixed panel i, it is obvious that most other panels belong to the set $far(i)$. Hence we only focus on matrix blocks F^{ij} with $i \in far(j)$. Using properties in (14)-(16), one can show that the regions I and II in these matrix blocks satisfy

1. Region I: $j \in far(i)$, $j \in far(m)$ and $j \in far(n)$
2. Region II: $i \in far(j)$, $i \in far(m)$ and $i \in far(n)$

Following the same reasoning in (17)-(18), we immediately obtain

$$P_4(h_i,h_j,h_m,h_n;\bar{x}_i,\bar{x}_j,\bar{x}_m,\bar{x}_n) \simeq P_3(h_i,h_m,h_n;\bar{x}_i,\bar{x}_m,\bar{x}_n)P_1(h_j) \tag{38}$$

in region I, and

$$P_4(h_i,h_j,h_m,h_n;\bar{x}_i,\bar{x}_j,\bar{x}_m,\bar{x}_n) \simeq P_3(h_j,h_m,h_n;\bar{x}_j,\bar{x}_m,\bar{x}_n)P_1(h_i) \tag{39}$$

in region II, where function P_1 and P_3 are Gaussian and joint Gaussian, respectively . Let

$$F^{ij} = H_1^{ij} + H_2^{ij} + H_3^{ij} \tag{40}$$

where the entries in sparse matrices H_1^{ij}, H_2^{ij} and H_3^{ij} respectively belong to region I, II and III in figure 2. Since matrix H_3^{ij} is very sparse and its entries are usually one to two orders of magnitude smaller than those of H_1^{ij} and H_2^{ij}, we neglect its contribution. In view of (20), we have

$$f^{ij} \simeq (H_1^{ij} + H_2^{ij}) < \rho^{(0)} >= f_1^{ij} + f_2^{ij}. \tag{41}$$

We first focus on the vertical sub-region of region I. To conduct the asymptotic analysis, we only need to look at the off-diagonal entries i.e., $m \in far(n)$. By definition, we have $n \in near(i)$ in the vertical sub-region of region I. From property (16) it is easy to check that this implies $m \in far(i)$. Similar to (38), we immediately obtain

Table 5: Asymptotic behavior of terms in (41)

	$f_1^{ij}(m)$	$f_2^{ij}(m)$
$r_{mi},\ m \in far(i)$	$\frac{1}{r_{mi}}$	$\frac{1}{r_{mi}}$
$r_{mj},\ m \in far(j)$	$\frac{1}{r_{mj}}$	$\frac{1}{r_{mj}}$

$$P_3(h_i,h_m,h_n;\bar{x}_i,\bar{x}_m,\bar{x}_n) \simeq P_2(h_i,h_n;\bar{x}_i,\bar{x}_n)P_1(h_m). \tag{42}$$

Substituting (42), (38) and (11) into (8) and in view of (40), we obtain

$$\begin{aligned} H_1^{ij}(m,n) \quad \simeq \quad & \int_{-\infty}^{+\infty} dh_j P_1(h_j) \int_{-\infty}^{+\infty} dh_m P_1(h_m) \\ & \int_{-\infty}^{+\infty} dh_i \int_{-\infty}^{+\infty} dh_n \\ & P_2(h_i,h_n;\bar{x}_i,\bar{x}_n) A_{ij} A_{mn} - \bar{A}_{ij}\bar{A}_{mn} \end{aligned} \tag{43}$$

for the vertical sub-region. Substituting (43) into (41), we obtain

$$\begin{aligned} f_1^{ij}(m) \quad \simeq \quad & \int_{-\infty}^{+\infty} dh_j P_1(h_j) \int_{-\infty}^{+\infty} dh_m P_1(h_m) \\ & \int_{-\infty}^{+\infty} dh_i A_{ij} V_m^i - \bar{A}_{ij}\bar{V}_m^i, \end{aligned} \tag{44}$$

where

$$V_m^i = \sum_{n \in near\{i\}} \int_{-\infty}^{+\infty} dh_n P_2(h_i,h_n;\bar{x}_i,\bar{x}_n) A_{mn} < \rho_n^{(0)} > \tag{45}$$

and

$$\bar{V}_m^i = \sum_{n \in near\{i\}} \bar{A}_{mn} < \rho_n^{(0)} > . \tag{46}$$

We know the off-diagonal entries A_{mn} and $\bar{A}_{mn}$ scales as $\frac{1}{r_{mn}}$, where r_{mn} is the distance between panels m and n. Since $n \in near\{i\}$ in (45), r_{ni} is very small, as shown in (12). For the purpose of asymptotic analysis, panels n and i can be considered as overlapping. Therefore, V_m^i and $\bar{V}_m^i$ all scale as $\frac{1}{r_{mi}}$. [2] This implies that $f_1^{ij}(m)$ scales as $\frac{1}{r_{mi}}$. The degeneration in (38) and (42) makes $f_1^{ij}(m)$ independent of r_{mj}, as can be easily checked from (44).

Using the same analysis outlined above for the horizontal sub-region of region I, one can also show that the corresponding entry $f_1^{ij}(m)$ scales as $\frac{1}{r_{mj}}$. By definition, $m \in near(i)$ in the horizontal sub-region. So the asymptotic behavior of $f_1^{ij}(m)$ in term of r_{mi} is irrelevant here.

The actual asymptotic behavior of $f_1^{ij}(m)$ is the total sum of the contributions from both horizontal and vertical sub-regions. Hence $f_1^{ij}(m)$ scales as $\frac{1}{r_{mi}}$ (when $m \in far(i)$) and $\frac{1}{r_{mj}}$ (when $m \in far(j)$), respectively.

We have applied the same procedure to $f_2^{ij}(m)$. The asymptotic behavior of both $f_1^{ij}(m)$ and $f_2^{ij}(m)$ are listed in table 5.

In view of (19), the asymptotic behavior of B_{im} or $b_i(m)$ in terms of r_{mi} is the same as that of $f_1^{ij}(m) + f_2^{ij}(m)$. From table 5 it is clear B_{im} should scale as $\frac{1}{r_{mi}}$, except for the case where $m \in near\{i\}$. Hence matrix B is hierarchically low rank, similar to a discretized integral operator with $\frac{1}{r}$ kernel.

In view of (19) and (41), it is clear that the asymptotic behavior of $M_{mj}^{(i)}$ in terms of r_{mj} is the same as that of $f_1^{ij}(m) + f_2^{ij}(m)$. From table 5, $M_{mj}^{(i)}$ should scale as $\frac{1}{r_{mj}}$, except for the case where $m \in near\{j\}$. Notice this is independent of the index i, hence the combined matrix entry $\tilde{M}_{mj}$ should also scale as $\frac{1}{r_{mj}}$. In other words, matrix $\tilde{M}$ is hierarchically low rank, similar to a discretized integral operator with $\frac{1}{r}$ kernel.

[2] The rapid variation in $P_2(h_i,h_n;\bar{x}_i,\bar{x}_n)$ has been absorbed into the summation in (45) and becomes irrelevant.