An Efficient MLACA-SVD Solver for Superconducting Integrated Circuit Analysis

Ben A. P. Nel and Matthys M. Botha, *Member, IEEE*

***Abstract*—Inductance extraction for superconducting integrated circuits requires the accurate solution of structural current distributions. FastHenry is a well known, magnetoquasistatic solver suitable for this task. It is based upon the partial element equivalent circuit, integral equation method, with the structure discretized into hexahedral filaments. It employs the multilevel fast multipole algorithm (MLFMA) for compressed storage of the mutual inductance matrix, which accounts for most of the required memory. This MLFMA implementation is especially memory efficient, given certain approximations and algorithmic parameter choices. However, errors are introduced into the matrix representation. Here, a multilevel adaptive cross approximation solver with singular value decomposition recompression (MLACA-SVD) is presented as an alternative to FastHenry's existing MLFMA solver. MLACA-SVD compresses off-diagonal matrix blocks to a specified error tolerance, based on evaluating selected entries. Quadrature recipes are presented for guaranteed accuracy of matrix entry evaluation. Numerical results for examples of practical interest show that the MLACA-SVD memory scaling versus b (number of filaments) is practically identical to that of FastHenry's MLFMA, and is close to $\mathcal{O}(b \log b)$. The MLACA-SVD requires less memory for the same solution accuracy, and furthermore offers complete control over matrix approximation errors. For the examples considered, it is found to be a more efficient solver. It is well suited to parallelization.**

***Index Terms*—Hierarchical octree, low-rank factorization, numerical integration, superconductor, SuperTools, volume current density.**

I. Introduction

COMPLEX superconducting digital integrated circuits is of interest for beyond-CMOS digital applications, such as extremely energy efficient supercomputing systems [1]. The recent IARPA C3 program [2] provided strong impetus for developments in this area; and highlighted the need for sophisticated synthesis, layout, parameter extraction, and verification tools [3]. The current IARPA SuperTools program has the objective of developing tools for the design and verification of complex superconducting digital integrated circuits, such as 64-bit RISC processors [4]. The work presented in this paper forms part of the ColdFlux project [5], which falls under the SuperTools program.

To characterize the behavior of complex three-dimensional superconducting integrated circuits, e.g., by constructing equivalent circuit models [6], [7], it is necessary to accurately solve the structural current distributions [8]–[10]. FastHenry is a well known, efficient magnetoquasistatic solver which is suitable for superconducting, inductance extraction [11], [12]. It is based upon the partial element equivalent circuit (PEEC), integral equation method, with the structure discretized into right-angled hexahedral filaments. The mutual inductance matrix, representing inter-filament coupling, accounts for the bulk of the computer memory requirement. FastHenry employs the multilevel fast multipole algorithm (MLFMA) for compressed storage of this matrix. In this paper, a multilevel adaptive cross approximation solver with singular value decomposition recompression (MLACA-SVD) is presented as alternative to the existing MLFMA solver, with the aim of improving the efficiency of FastHenry.

The scalar, static Green's function features in FastHenry's mutual inductance matrix. The MLFMA is based upon analytical factorization of this function and hence, compression of the matrix. It is a very well-established approach [11], [13]. Its implementation in FastHenry is especially memory efficient, given the approximations and fixed algorithmic parameter choices introduced. Theoretically, the compressed matrix storage scaling is expected to be $\mathcal{O}(b)$ instead of the conventional $\mathcal{O}(b^2)$ [11], where b denotes the number of filaments. The adaptive cross approximation (ACA) is a purely algebraic approach to compression [14]–[16], which emerged after FastHenry was already established. ACA is applicable to asymptotically smooth kernels, such as the static Green's function of interest, for which the MLACA is expected to yield $\mathcal{O}(b \log b)$ memory scaling [14], [16], [17]. For an off-diagonal matrix block, the ACA determines an approximate rank and factorized form, by traversing rows and columns until convergence is reached, according to a specified error tolerance. The ACA algorithm has been extended with singular value decomposition (SVD) recompression [18], [19]. The SVD is not simply applied directly (without ACA), although it would yield the optimal approximant, because it would require knowledge of the whole block and would require much higher runtime. Finally, it should be noted that there are also FFT-based acceleration (compression) schemes [20], [21], but those require a regular mesh, while the present study is focused on providing an alternative which is directly applicable to FastHenry meshes.

Manuscript received December 31, 2018; accepted February 18, 2019. Date of publication March 27, 2019; date of current version May 24, 2019. This work was supported by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via the U.S. Army Research Office under Grant W911NF-17-1-0120, and in part by the National Research Foundation of South Africa under Grant 96222. This paper was recommended by Associate Editor H. Rogalla. *(Corresponding author: Ben A. P. Nel.)*

The authors are with the Department of Electrical and Electronic Engineering, Stellenbosch University, Stellenbosch 7600, South Africa (e-mail: 17762944@sun.ac.za; mmbotha@sun.ac.za).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TASC.2019.2906171

Section II reviews the existing FastHenry solver, with particular attention to aspects relevant to the study at hand. Section III explains how ACA-SVD is applied to compress a mutual inductance submatrix, representing the coupling between two disjoint groups of filaments. Given that the control of compression accuracy is an important feature of the ACA-SVD, it was found necessary to revisit the accuracy to which the direct calculation of matrix entries is performed, such that the benefits of accurate compression is not lost due to large quadrature errors. This leads to the development of quadrature recipes for guaranteed matrix entry accuracy, presented in Section IV. Section V describes the complete MLACA-SVD solver scheme, which is followed by numerical results in Section VI, to assess the performance of the new solver, for superconducting integrated circuit modeling. Section VII delivers the overall conclusions to this study.

II. MLFMA-Accelerated, Integral Equation-Based, PEEC Formulation of FastHenry

In FastHenry, the conducting structure of interest is discretized into a mesh with a total of b, right-angled hexahedral filaments, with a single, constant axial basis function upon each one. Let I_b denote the vector of complex-valued, phasor current coefficients, representing the currents flowing along all filaments, with reference directions according to the basis functions. The PEEC method, based on a volume integral equation with assumed magnetoquasistatic conditions, then yields

$$ZI_b = V_b \tag{1}$$

where V_b is the vector of filament potential differences, with $Z = R + j\omega L$ being the $b \times b$ branch impedance matrix and ω is the angular frequency. Galerkin testing is used. The resistance and mutual inductance matrices are defined as

$$R_{ii} = \frac{\ell_i}{\sigma(\boldsymbol{r})a_i} \tag{2}$$

$$\sigma(\boldsymbol{r}) = \sigma_{\mathrm{C}}(\boldsymbol{r}) + \frac{1}{j\omega\mu\lambda(\boldsymbol{r})^2} \tag{3}$$

$$L_{ij} = \frac{\mu}{4\pi a_i a_j}\int_{V_i}\int_{V_j}\frac{\hat{\ell}_i \cdot \hat{\ell}_j}{|\boldsymbol{r}-\boldsymbol{r}'|}dV'dV \tag{4}$$

with $i, j \in \{1, ..., b\}$; i and j refer to the testing and source filament numbers, respectively. Equation (3) defines the conductivity, with σ_{C} denoting the temperature-dependent, conventional conductivity; the second term accounts for superconductivity [22], with λ being the temperature-dependent London penetration depth. In (2) and (4), V_i, a_i, ℓ_i, and $\hat{\ell}_i$ denote the ith filament's domain, cross-sectional area, axial length, and axial unit-vector, respectively (similarly for j). The basis function associated with the ith filament is $\hat{\ell}_i/a_i$.

A mesh-current approach yields the final system of linear equations

$$MZM^T I_m = V_s \tag{5}$$

where m is the total number of linearly independent loops in the mesh and V_s is the (typically sparse) vector of source voltages in all loops, such that $MV_b = V_s$, with M following from Kirchhoff's voltage law applied to all loops. The loop currents relate to the filament currents as $M^T I_m = I_b$.

The resistance matrix is highly sparse (diagonal), but the mutual inductance matrix is dense and storage requirements are prohibitive for large b, which is typically the case for superconducting, integrated circuit structures. Therefore, the well-known MLFMA is used to represent L in a factorized format, theoretically requiring $\mathcal{O}(b)$ storage and $\mathcal{O}(b)$ runtime for a matrix-vector product [11]. An iterative solver using the generalized minimal residual (GMRES) method [23] combined with preconditioning is used to solve (5) [11].

To employ the MLFMA, start by defining the diagonal, vector-valued *basis matrix*, with diagonal entries equal to the basis functions evaluated within each element, multiplied with the elemental volume

$$\boldsymbol{\Lambda}_{ii} = \frac{V_i\hat{\ell}_i}{a_i} = \ell_i\hat{\ell}_i. \tag{6}$$

Furthermore, define the *potential matrix* Φ

$$\Phi_{ij} = \frac{\mu}{4\pi V_i V_j}\int_{V_i}\int_{V_j}\frac{1}{|\boldsymbol{r}-\boldsymbol{r}'|}dV'dV \tag{7}$$

with $i, j \in \{1, ..., b\}$. It follows that the mutual inductance matrix can be expressed as

$$L = \boldsymbol{\Lambda} \cdot \Phi\boldsymbol{\Lambda}. \tag{8}$$

An MLFMA-representation is established for Φ. The MLFMA-representation rests upon a hierarchical (multilevel), octree grouping of the mesh filaments. Nonself interactions between groups relating to off-diagonal blocks in Φ, are represented in factorized form by exploiting a truncated, series expansion representation of the Green's function $1/|\boldsymbol{r}-\boldsymbol{r}'|$ [13]. Factorization involves aggregation, translation, and disaggregation factors. Aggregation and disaggregation factors can be utilized at multiple levels and the objective is to treat interactions at the highest possible level, at which they qualify as far interactions, according to a near-interaction criterion. This criterion is necessary to ensure accuracy of the truncated series expansion. A cubic, second-nearest neighbor criterion is used, as illustrated in Fig. 6 (right). This means that interactions between groups at leaf level (the smallest group size), are stored directly in case both fall within a $3 \times 3 \times 3$ leaf-level group sized cube.

The accuracy with which this approach represents the true mutual inductance matrix will be evaluated later-on. The near-interaction entries are directly calculated using approximate analytical expressions or quadrature, as automatically determined to be most appropriate. Errors are introduced by the MLFMA into the far-interaction entries, due to the factorized Green's function representation itself, as well as due to the factorization being done for a midpoint-quadrature-based, approximate representation of Φ_{ij}, i.e.,

$$\Phi_{ij} \approx \frac{\mu}{4\pi}\frac{1}{|\boldsymbol{r}_i-\boldsymbol{r}_j|} \tag{9}$$

where $\boldsymbol{r}_i$ and $\boldsymbol{r}_j$ denote the centroids of volumes V_i and V_j, respectively. The reason for this is the MLFMA implementation is most efficient when dealing with interactions between point

sources (i.e., the midpoints), rather than interactions between distributed sources.

III. ACA-SVD Compression of a Mutual Inductance, Off-Diagonal Submatrix

Consider a submatrix denoted by L^{sub}, of the mutual inductance matrix (4), representing the interaction between two disjoint groups of filaments, denoted as groups P (consisting of p testing basis functions) and Q (consisting of q source basis functions). It follows that

$$L^{\text{sub}} = \mathbf{\Lambda}_P \cdot \Phi_{PQ} \mathbf{\Lambda}_Q \tag{10}$$

where Φ_{PQ} represents the appropriate submatrix of the global potential matrix (7); and $\mathbf{\Lambda}_P$ and $\mathbf{\Lambda}_Q$ are appropriate, diagonal submatrices of the global basis matrix (6).

Generally, the ACA is expected to perform well for inter-group matrices of which the entries are proportional to Green's functions associated with various physical boundary value problems [24]; particularly so for the asymptotically smooth, scalar Green's function of Poisson's equation, as featured in (4). Performance improves as the distance between groups grow, relative to the group diameters [14], [18], [25]. Scalar testing/source functions present no difficulties, as they effectively just scale matrix rows and columns, which can be normalized. However, irregular scaling of matrix entries within rows and columns, can cause catastrophic algorithm breakdown [26], [27]. Such irregular scaling is encountered in L^{sub}, due to the dot-product between the axial unit-vectors. e.g., suppose in the two groups each filament is oriented in one of two orthogonal directions, then with appropriate numbering of degrees of freedom, L^{sub} takes the form

$$L^{\text{sub}} = \begin{bmatrix} L_{11}^{\text{sub}} & 0 \\ 0 & L_{22}^{\text{sub}} \end{bmatrix}. \tag{11}$$

If the ACA algorithm starts by investigating a column intersecting with L_{11}^{sub}, then the algorithm will continue to place crosses centred inside L_{11}^{sub}, until convergence. The part L_{22}^{sub} will never be examined and will be falsely considered to be zero. Therefore, as in the case of the MLFMA compression, the ACA is not applied directly to L^{sub} in (10), but rather to Φ_{PQ}.

Application of the standard ACA algorithm [14], [17], yields the following approximate factorization:

$$[\Phi_{PQ}]_{p\times q} \approx [U]_{p\times k} \left[V^T\right]_{k\times q}. \tag{12}$$

The algorithm requires evaluating selected rows and columns of Φ_{PQ}. In contrast to FastHenry's MLFMA, midpoint-integration is not used, but rather a quadrature recipe from Section IV, which ensures the rigorous evaluation of all matrix entries to a higher accuracy than a specified error tolerance.

The ACA-estimated rank $k \leq \min\{p, q\}$ (with the expectation that $k \ll \min\{p, q\}$ should hold) is determined by the factorization error tolerance setting. During the ACA factorization algorithm, the relative factorization error ε_k is approximately determined at each iteration k, by way of efficient recursive calculations [15]. It is defined and approximated as

$$\begin{aligned} \varepsilon_k &\equiv \frac{\|\Phi_{PQ} - U_k V_k^T\|_F}{\|\Phi_{PQ}\|_F} \\ &\approx \frac{\|U_k V_k^T - U_{k-1} V_{k-1}^T\|_F}{\|U_k V_k^T\|_F} \\ &= \frac{\|U_k(:,k)\|_F \|V_k(:,k)\|_F}{\|U_k V_k^T\|_F} \end{aligned} \tag{13}$$

where U_k and V_k denote the factors after k iterations; $\|\cdot\|_F$ represents the Frobenius norm. The algorithm terminates when a specified relative error tolerance level ε_{ACA}, is reached, i.e.,

$$\varepsilon_k \leq \varepsilon_{\text{ACA}}. \tag{14}$$

Generally, the columns of neither U nor V are orthogonal [19]. Therefore, SVD is further applied to eliminate any possible redundancy within U and V, as recommended in [18], [19]. First, QR decompositions of U and V are computed using the Householder algorithm [28], as

$$U = [Q_U]_{p\times k} [R_U]_{k\times k} \tag{15}$$

$$V = [Q_V]_{q\times k} [R_V]_{k\times k}. \tag{16}$$

Then apply SVD to $R_U R_V^T$

$$R_U R_V^T = \tilde{U} \Sigma \tilde{V}^T. \tag{17}$$

This yields the desired SVD of the product UV^T

$$UV^T = Q_U \tilde{U} \Sigma \left(Q_V \tilde{V}\right)^T. \tag{18}$$

The singular values are found in descending order on the diagonal of Σ, with the largest denoted by $\sigma_{\max}$. All values falling below a relative threshold are removed [18]. The threshold is defined as

$$\frac{\sigma_i}{\sigma_{\max}} \leq \varepsilon_{\text{SVD}} \qquad i \in \{1, ..., k\}. \tag{19}$$

This results in a new rank $\tilde{k} \leq k$ and recompression of the original U and V matrices. Since the SVD rank is precise, while the ACA rank is approximate, a $10\times$ buffer factor is employed to ensure reliable ACA-SVD accuracy

$$\varepsilon_{\text{SVD}} = 10\varepsilon_{\text{ACA}}. \tag{20}$$

The SVD tolerance is then considered the actual matrix compression tolerance.

The additional computational cost of adding SVD on top of ACA does not change the overall cost scaling of the ACA. The runtime for steps (15), (16), and (18) together, scales as $\mathcal{O}(k^2(p + q + k))$ [28].

IV. Accurate Evaluation of Mutual Inductance Matrix Entries

A major benefit of ACA-SVD compression is control over accuracy, by way of (19). For this to be of value, the errors in evaluating the mutual inductance matrix entries must also be controlled. Quadrature recipes have thus been developed which

guarantee specified bounds on the relative errors with which the matrix entries are evaluated. Relative error is defined as

$$\varepsilon_{\text{quad}} = \left| \frac{L_{ij}^{\text{quadrature}} - L_{ij}^{\text{reference}}}{L_{ij}^{\text{reference}}} \right|. \tag{21}$$

Throughout this section, the testing and source basis functions are assumed to be colinear, without loss of generality.

For the inner (source) integral in (4), either Gaussian (Gaussian–Legendre) quadrature [29] in product-rule format is used, or analytical evaluation [30]. The choice depends upon the distance to the testing domain, since Gaussian quadrature requires a smooth integrand. As the testing and source domains get closer, the near-singular (less smooth) behavior of the kernel increases.

For the outer (testing) integral, Gaussian product-rule quadrature is always used. This is an optimal choice with regards to error convergence, which follows from considering the most extreme behavior of the integrand, which occurs when V_i and V_j are equal (self-interaction) or share a face. It can be shown that the integrand's leading term behaves as $\mathcal{O}(\delta)$ when $\delta \to 0$, with δ denoting the orthogonal distance of the outer integration point from the coincident face. Since Gaussian quadrature is constructed for integrating polynomials, it is thus well-suited to this integrand.

The following normalized distance will be used in defining the recipes:

$$D_{\text{norm}} = \frac{D_{\min}}{d_{\max}} \tag{22}$$

with $D_{\min}$ denoting the minimum distance between the bounding boxes (in the global coordinate system) of V_i and V_j, and $d_{\max}$ denoting the maximum edge length among all six edge lengths in V_i and V_j. Table I shows the recipes for achieving relative error levels of $\varepsilon_{\text{quad}} < 10^{-6}$, $\varepsilon_{\text{quad}} < 10^{-4}$, and $\varepsilon_{\text{quad}} < 10^{-2}$. These error levels are guaranteed up to elemental maximum aspect ratios of 1:100. To demonstrate the use of the tables, consider two elemental domains with respective aspect ratios 1:5:10 and 1:1:20, and with $D_{\text{norm}} = 1.5$. Then, to achieve $\varepsilon_{\text{quad}} < 10^{-6}$ it follows from Table I that Gaussian product-rule quadrature should be used on both domains, with respective orders $4 \times 6 \times 6$ and $4 \times 4 \times 6$. Figs. 1, 3, 4, and 5 show quadrature error results using the proposed recipes, for two identical elemental domains under various relative positioning conditions. The elemental dimensions are chosen to be rather extreme, namely $1 \times 10 \times 100$. These results are invariant with regards to uniform scaling, thus no units are indicated. The results show that the recipes do indeed yield relative errors below the indicated thresholds.

Finally, it should be noted that more efficient recipes yielding less over-accuracy (as displayed in the example results), in exchange for fewer quadrature points, are possible. This could be done by introducing more D_{norm}-range brackets, more edge-length range brackets, and by determining the rule on each element based on $D_{\min}$ relative to its own maximum edge length, rather than $d_{\max}$. Such further refinements are beyond the scope of this paper and can be introduced as part of code-optimization, as desired.

TABLE I
QUADRATURE RECIPIES TO GUARANTEED RELATIVE ERROR VALUES $\varepsilon_{\text{quad}}$. IN CASE OF GAUSSIAN QUADRATURE FOR BOTH INNER AND OUTER INTEGRALS, NOTE THAT THE ORDERS ARE DETERMINED FOR THE INNER AND OUTER ELEMENTS INDIVIDUALLY AND IN THE SAME WAY. I.E. FOR A GIVEN ELEMENT, FIND ITS MINIMUM EDGE LENGTH (DENOTED BY $\ell_{\min}$) AND ASSIGN AN ORDER TO EACH DIMENSION OF THAT ELEMENT, ACCORDING TO EDGE LENGTH ℓ, ALONG THAT DIMENSION

Quadrature recipe for $\varepsilon_{\text{quad}} < 10^{-6}$.

Range of D_{norm}	Outer scheme	Inner scheme	Gaussian product-rule order along each elemental dimension, according to the edge length in that dimension. $\ell = \ell_{\min}$	$\ell_{\min} < \ell \leq 10\ell_{\min}$	$10\ell_{\min} < \ell \leq 100\ell_{\min}$
Self	Gaussian	Analytic	7	12	27
< 0.1	Gaussian	Analytic	7	13	31
< 0.5	Gaussian	Analytic	5	8	10
< 2.5	Gaussian	Gaussian	4	6	6
< 12	Gaussian	Gaussian	3	4	4
< 450	Gaussian	Gaussian	2	2	2
≥ 450	Gaussian	Gaussian	1	1	1

Quadrature recipe for $\varepsilon_{\text{quad}} < 10^{-4}$.

Range of D_{norm}	Outer scheme	Inner scheme	Gaussian product-rule order along each elemental dimension, according to the edge length in that dimension. $\ell = \ell_{\min}$	$\ell_{\min} < \ell \leq 10\ell_{\min}$	$10\ell_{\min} < \ell \leq 100\ell_{\min}$
Self	Gaussian	Analytic	4	7	14
< 0.15	Gaussian	Analytic	4	8	18
< 0.5	Gaussian	Analytic	3	5	5
< 3	Gaussian	Gaussian	3	4	4
< 4	Gaussian	Gaussian	2	2	2
< 41	Gaussian	Gaussian	1	2	2
≥ 41	Gaussian	Gaussian	1	1	1

Quadrature recipe for $\varepsilon_{\text{quad}} < 10^{-2}$.

Range of D_{norm}	Outer scheme	Inner scheme	Gaussian product-rule order along each elemental dimension, according to the edge length in that dimension. $\ell = \ell_{\min}$	$\ell_{\min} < \ell \leq 10\ell_{\min}$	$10\ell_{\min} < \ell \leq 100\ell_{\min}$
Self	Gaussian	Analytic	2	3	3
< 0.25	Gaussian	Analytic	1	3	6
< 3.5	Gaussian	Gaussian	1	3	4
> 3.5	Gaussian	Gaussian	1	1	1

V. MLACA-SVD SOLVER

In Section III, the compression of an off-diagonal block of Φ is presented, which is the core function employed in the MLACA-SVD solver. The solver employs the existing hierarchical (multilevel), octree mesh grouping scheme from FastHenry's MLFMA. Similarly to the MLFMA, intergroup interactions are classified as far or near interactions. Also similarly, the MLACA-SVD solver compresses all far interactions, with the objective to treat interactions at the highest possible level. What is different

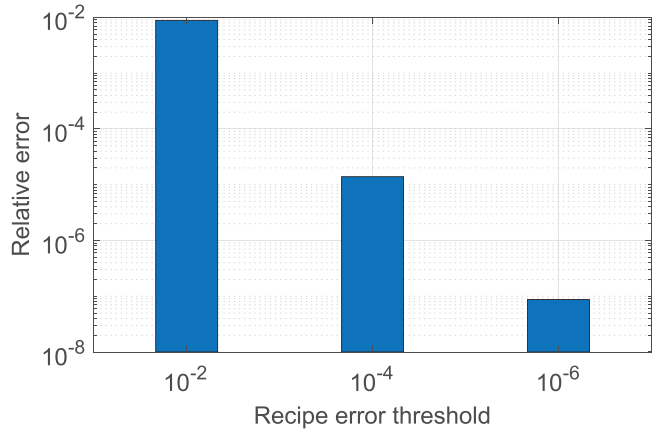

Fig. 1. Self-interaction (L_{ii}) quadrature errors using the three recipes, for a $1 \times 10 \times 100$ element.

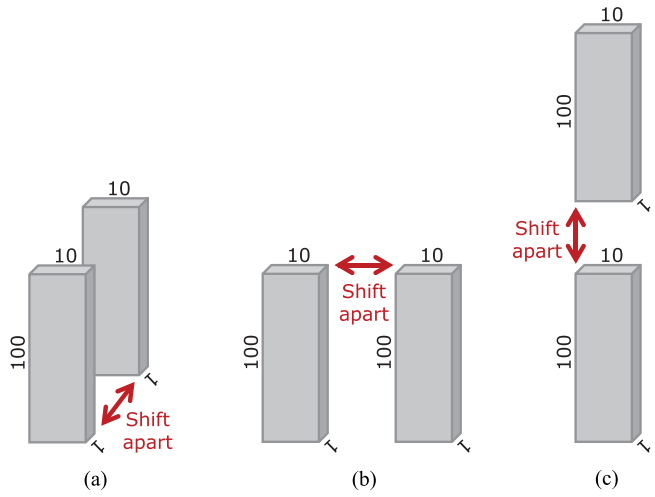

Fig. 2. Figurative illustration of the experimental setups for the results in Figs. 3–5. (a) Shifting apart in direction of the elemental 1-dimensions. (b) Shifting apart in direction of the elemental 10-dimensions. (c) Shifting apart in direction of the elemental 100-dimensions.

from FastHenry's MLFMA, apart from the compression algorithm, is the near-interaction criterion, the symmetric storage scheme, and the calculation of near-interaction matrix entries.

For the ACA on a given level of the octree, the η-admissibility condition, which predicts exponential decay of singular values for asymptotically smooth kernels [14], [25], can be stated as follows:

$$\operatorname{diam}(D_Q) \leq \eta \operatorname{dist}(D_P, D_Q), \qquad \{0 < \eta\} \tag{23}$$

where D_P and D_Q are taken as the octree cubes (identical in size) defining the testing and source filament groups, respectively. Increasing the distance between D_P and D_Q results in more rapid singular value decay, therefore reducing η is regarded as strengthening the admissibility condition. Fig. 6 illustrates two near-interaction criteria, depicted in two dimensions for simplicity. FastHenry's MLFMA uses the second-nearest neighbor criterion exclusively. The MLACA-SVD is tested with this same criterion. It corresponds to $\eta < \sqrt{3}/2$, which is a strong admissibility condition. The MLACA-SVD is also tested with the nearest-neighbor criterion. It is a weaker admissibility condition, with $\eta < \sqrt{3}$, which should theoretically result in a less accurate ACA approximation for the same rank [18]. The benefit of the nearest-neighbor criterion is that valid far interactions generally occur at a higher level, allowing ACA-SVD compression of larger submatrices. In Section VI, the relative efficiencies of these criteria are evaluated.

Since the MLACA-SVD compresses individual matrix blocks independently, the symmetric nature of L can be exploited. Only the upper-triangular part is factorized and stored. This information is then used both directly and in transposed form, when calculating a matrix-vector product. Symmetry is also exploited in the storage of near-interaction matrix entries.

To determine the near-interaction matrix entries resulting from leaf-level near-interactions, the MLACA-SVD solver does not use the direct-calculation routines of FastHenry. Rather, the quadrature recipes presented in Section IV are used; and with the same quadrature error threshold as that for constructing the ACA-SVD factorizations, such that accuracy is fully controlled.

Finally, with regards to the matrix equation solver: Exactly the same preconditioner and GMRES iteration scheme as for the MLFMA solver (noted in Section II) are used for the MLACA-SVD.

VI. Results

Fig. 7 shows the two superconducting, integrated circuit test models used to evaluate the performance of the MLACA-SVD solver; these are an adiabatic quantum-flux-parametron (AQFP) gate [31] and an energy-efficient single flux quantum (eSFQ) circuit [32]. Performance with regards to memory requirement and accuracy with which the true matrix L is approximated by the compressed versions, is considered.

All MLACA-SVD results are obtained using the $\varepsilon_{\text{quad}} < 10^{-6}$ quadrature recipe from Section IV, both for obtaining the factorization and the near-interaction matrix entries. The label "MLFMA" refers to results obtained with FastHenry's MLFMA solver with default settings, which employs the standard FastHenry routines to evaluate the near-interaction matrix entries.

A. Memory

The total memory required to store the mutual inductance matrix L (4) in compressed form, is measured as the mesh density is varied. Meshing is done with the InductEx package [7]. For the MLACA-SVD solver the near-interaction criteria of Fig. 6 are both considered, as well as four factorization tolerance settings, $\varepsilon_{\text{SVD}} \in \{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}\}$. Figs. 8 and 9 show the memory scaling results for the two models.

First, consider the observed scaling orders. Dashed trend lines of $\mathcal{O}(b \log b)$ show that the MLACA-SVD versions scale at this rate or slightly above it. This is expected for static/low-frequency solutions with asymptomatically smooth kernels [17]. According to [11], the expected scaling of the MLFMA is $\mathcal{O}(b)$ and here it does seem to scale on average, a little better than the MLACA-SVD versions. However, note that the MLFMA does

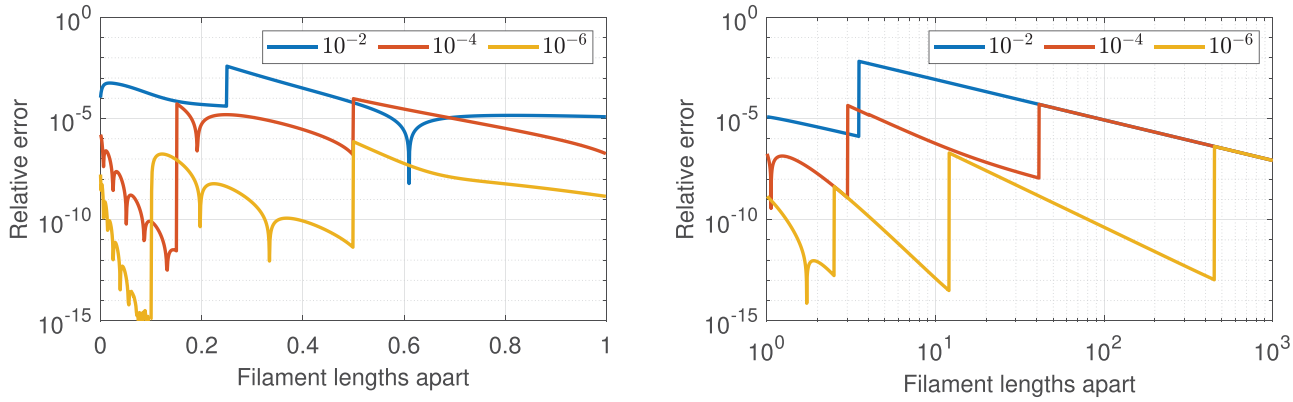

Fig. 3. Mutual-interaction (L_{ij}) quadrature errors using the three recipes, for two $1 \times 10 \times 100$ elements, with separation distance varied from 0 to 1000. The elements start off by sharing their 10×100 faces, with separation introduced orthogonally to these faces, in the 1-dimensions direction [see Fig. 2(a)].

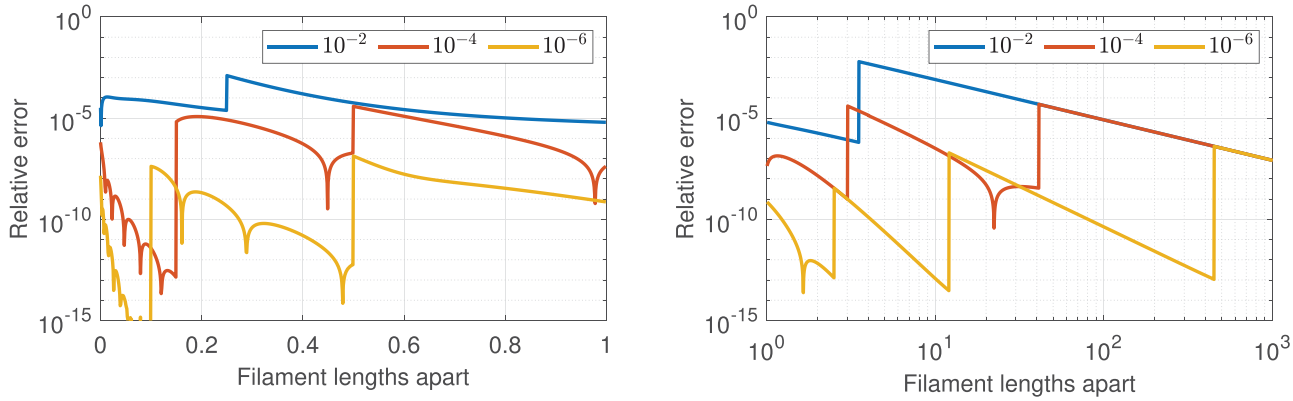

Fig. 4. Mutual-interaction (L_{ij}) quadrature errors using the three recipes, for two $1 \times 10 \times 100$ elements, with separation distance varied from 0 to 1000. The elements start off by sharing their 1×100 faces, with separation introduced orthogonally to these faces, in the 10-dimensions direction [see Fig. 2(b)].

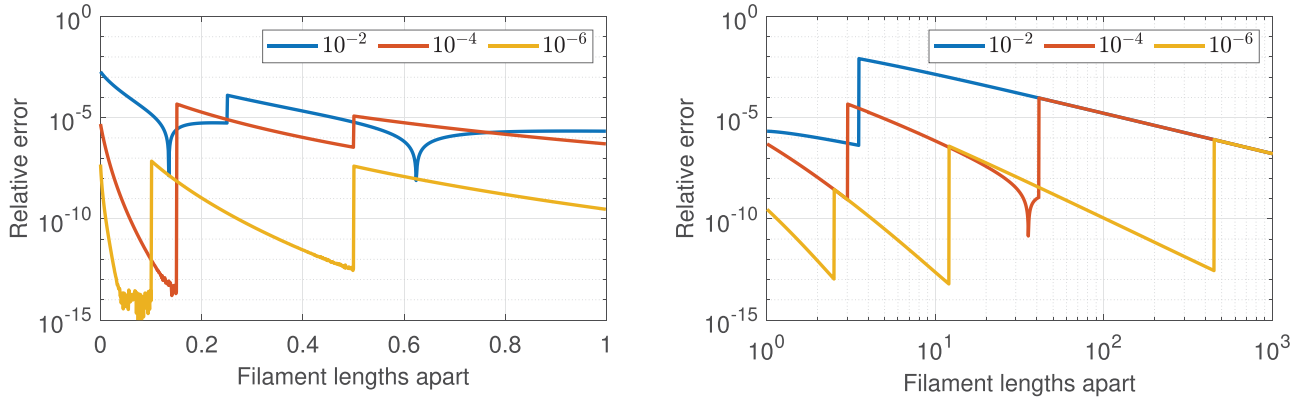

Fig. 5. Mutual-interaction (L_{ij}) quadrature errors using the three recipes, for two $1 \times 10 \times 100$ elements, with separation distance varied from 0 to 1000. The elements start off by sharing their 1×10 faces, with separation introduced orthogonally to these faces, in the 100-dimensions direction [see Fig. 2(c)].

not exploit symmetry in the storage of its directly-calculated, near-interaction entries. At low mesh sizes these tend to dominate the MLFMA memory requirement. To remove this influence, the MLFMA is compared with the second-nearest neighbor criterion MLACA-SVD, with near-interaction storage excluded for both, i.e., purely the compressed storage of exactly the same sets of matrix entries are compared. Fig. 10 shows the results, which indicate that for the considered test cases, the MLFMA in fact scales the same or even slightly worse than the MLACA-SVD.

Second, consider the comparative memory requirements of Figs. 8 and 9 in absolute terms. Clearly, applying the nearest-neighbor criterion MLACA-SVD always yields a lower memory requirement than with the second-nearest criterion version, for the same mesh and SVD tolerance. Also clearly, the MLACA-SVD memory requirement is strongly and uniformly influenced

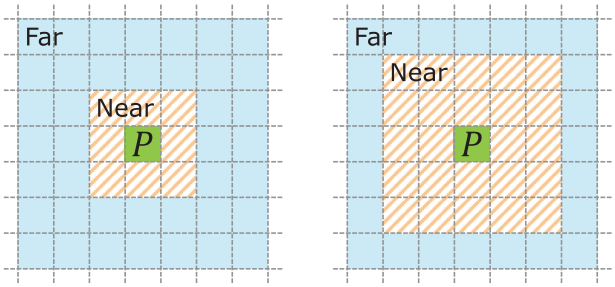

Fig. 6. Two near-interaction criteria, with respect to group P. Left: nearest neighbors. Right: second-nearest neighbors.

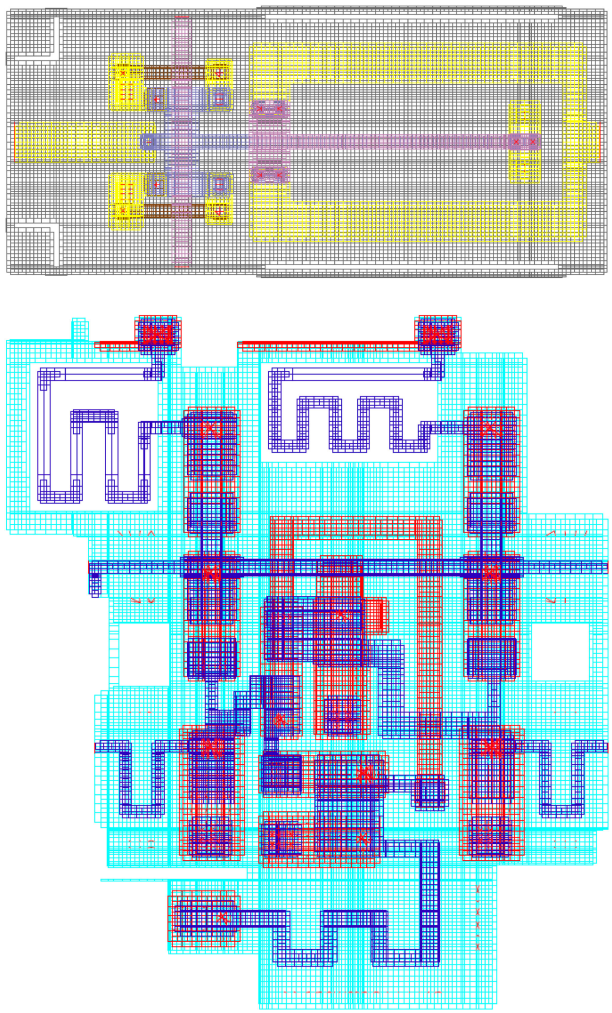

Fig. 7. Example meshes of the two test models, representing superconducting, integrated circuit structures. Top: an AQFP gate. Bottom: an eSFQ circuit.

by the choice of SVD tolerance. Arguably the most important observation, is that the nearest-neighbor criterion MLACA-SVD out-performs the MLFMA. However, this should be considered in conjunction with the relative accuracies of these compressed representations. Accuracy is considered next.

B. Accuracy

Both test models have a number of defined ports. At each port in turn, a unit voltage source is connected with all others

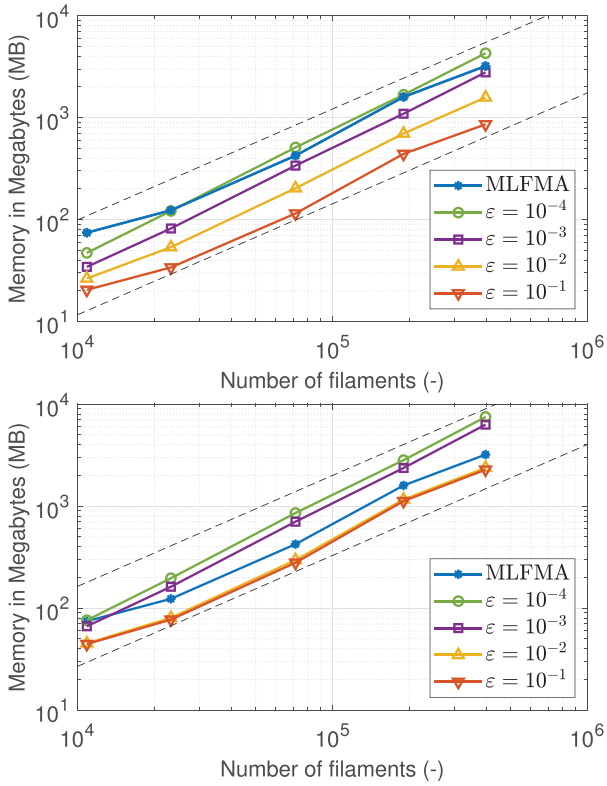

Fig. 8. Compressed matrix memory requirement versus number of filaments, for the AQFP gate. The total number of filaments is varied from 10 874 to 398 318; tolerance values refer to $\varepsilon_{\mathrm{SVD}}$. Top: MLACA-SVD with nearest-neighbor criterion. Bottom: MLACA-SVD with second-nearest neighbor criterion.

shorted out. After solving the global current distribution, a vector of the port currents is obtained. These vectors form the columns of a port current matrix I_{port}, which is the key input to circuit parameter extraction tools such as InductEx [7].

Accuracy of the compressed representations is evaluated by way of relative port current matrix errors, defined as

$$\text{Relative error} = \frac{\left\| I_{\mathrm{port}}^{\mathrm{approximate}} - I_{\mathrm{port}}^{\mathrm{reference}} \right\|_F}{\left\| I_{\mathrm{port}}^{\mathrm{reference}} \right\|_F}. \tag{24}$$

Reference port current matrices $I_{\mathrm{port}}^{\mathrm{reference}}$ are obtained by calculating all mutual inductance matrix entries with the $\varepsilon_{\mathrm{quad}} < 10^{-6}$ recipe, for a direct solution in the case of the AQFP gate (23 226 filaments) and for a second-nearest neigbour criterion, MLACA solution ($\varepsilon_{\mathrm{ACA}} = 10^{-7}$, no SVD recompression) in the case of the eSFQ circuit (38 895 filaments). The latter solution together with the approximate solutions $I_{\mathrm{port}}^{\mathrm{approximate}}$ (with MLACA-SVD or MLFMA) are obtained with the iterative solver convergence criterion set to an extremely small value, such that the error is only determined by the accuracy of the compressed matrix representation (i.e., these are essentially direct solutions with the reconstituted matrices).

Fig. 11 shows the results. Relative errors of $\sim 10^{-2}$ are introduced by the MLFMA. This is due to errors in calculating the near-interaction entries, as well as the coarse, midpoint quadrature upon which the compression is based [see (9)] and

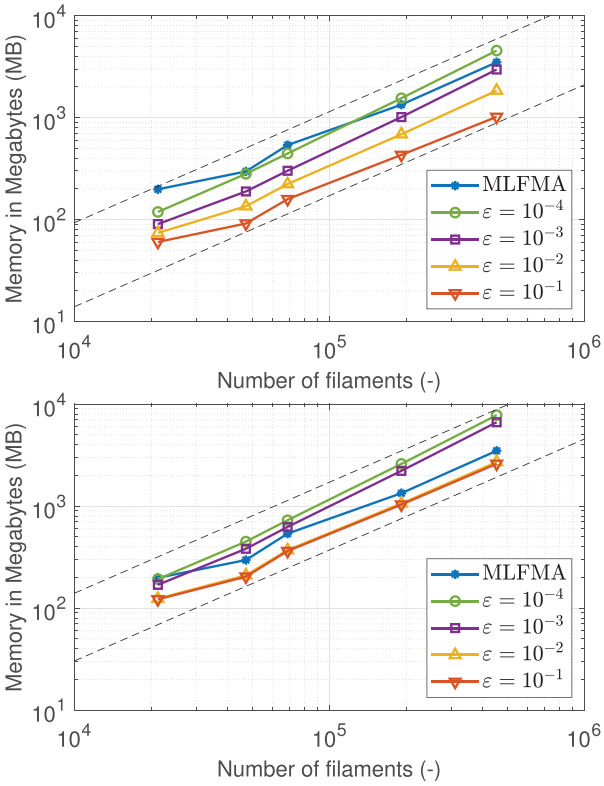

Fig. 9. Compressed matrix memory requirement versus number of filaments, for the eSFQ circuit. The total number of filaments is varied from 21 251 to 452 874; tolerance values refer to $\varepsilon_{\mathrm{SVD}}$. Top: MLACA-SVD with nearest-neighbor criterion. Bottom: MLACA-SVD with second-nearest neighbor criterion.

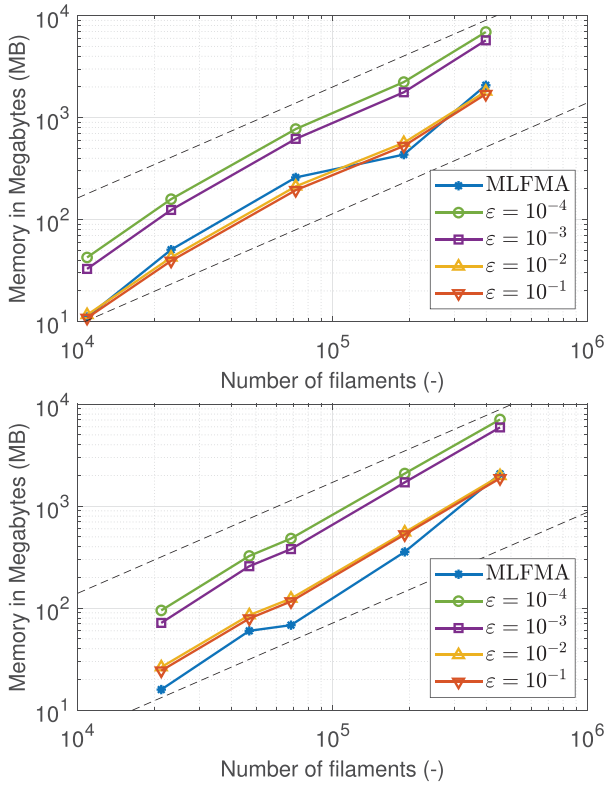

Fig. 10. Compressed matrix memory requirement with near-interaction storage excluded, for MLACA-SVD and MLFMA with the same, second-nearest neighbor criterion. Tolerance values refer to $\varepsilon_{\mathrm{SVD}}$ and $\mathcal{O}(b \log b)$ trend lines are shown. Top: AQFP gate. Bottom: eSFQ circuit.

the approximate nature of the compression itself. On the other hand, the accuracy with which the MLACA-SVD represents the true matrix is fully controllable through parameter $\varepsilon_{\mathrm{SVD}}$ (19). An important implication is that when using a mesh which is sufficiently fine to yield a very accurate current distribution solution (while rigorously evaluating the matrix entries), the accuracy can be limited by the MLFMA-introduced error. This is not an issue for the MLACA-SVD, which can in principle solve the electromagnetic field problem to any desired accuracy.

The results of Fig. 11 together with those in Figs. 8 and 9, indicate that the MLACA-SVD is more efficient than FastHenry's MLFMA. Particularly, the nearest-neighbor criterion together with either $\varepsilon_{\mathrm{SVD}} = 10^{-3}$ or $\varepsilon_{\mathrm{SVD}} = 10^{-2}$ is both more accurate than the MLFMA and require less memory than the MLFMA. For the MLACA-SVD, the nearest-neighbor criterion is clearly more efficient than the second-nearest neighbor criterion, when considering accuracy versus memory. Nevertheless, the nominal, improved accuracy obtained with the larger near-interaction criterion for the same mesh, does warrant discussion. It is partly due to the nearer interaction criterion simply resulting in a larger percentage of the matrix being approximated [33] (assuming that near interactions are calculated more accurately than compressed ones). However, it is mainly due to the relationship between the η-admissibility condition (23) and accuracy, as explained in Section V.

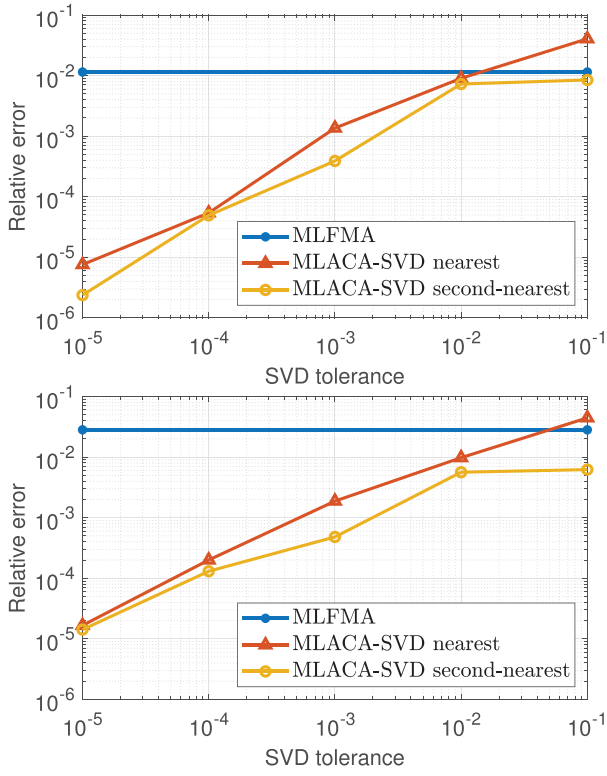

Fig. 11. Relative port current matrix errors versus SVD tolerance. Top: AQFP gate model with 23 226 filaments. Bottom: eSFQ circuit model with 38 895 filaments.

VII. Conclusion

In this paper, an MLACA-SVD solver is presented, which improves the accuracy and efficiency of magnetoquasistatic analysis of superconducting structures with FastHenry, e.g., for superconducting integrated circuit inductance extraction purposes. The MLACA-SVD solver is an alternative to the existing MLFMA solver in FastHenry. The same underlying set of linear equations is solved, but with different approaches for calculating and storing the near-interaction matrix entries and the compressed, far-interaction matrix entries. Near-interactions are calculated with rigorously determined semi-analytical procedures of which the accuracy can be controlled, rather than using FastHenry's existing, approximate calculations of fixed accuracy. Far interactions are compressed with the MLACA-SVD scheme, which employs the new near-interaction procedures, to evaluate the selected matrix entries needed for compression. For the compression itself, the SVD error tolerance can be set, which controls its accuracy at the cost of increased memory consumption. Two near-interaction distance criteria are tested for the MLACA-SVD, one being exactly the same as that of FastHenry's MLFMA (second-nearest neighbors) and the other being the nearest-neighbor criterion. The latter is found to yield more efficient results.

The MLACA-SVD memory requirement for the compressed far interactions versus number of filaments, is shown to scale practically identical to that of the MLFMA, for the complex and realistic superconducting integrated circuit models considered. The total memory requirement for the MLACA-SVD solver is lower than that of FastHenry's MLFMA, for the same relative error in the final output (port currents). Beyond this important property, the MLACA-SVD provides full user control over approximation errors in the matrix representation, such that solutions can be obtained to any desired accuracy.

Finally, runtimes are not presented as these scale the same as the memory requirements. Note that the matrix setup time for the MLACA-SVD is longer than that of FastHenry's MLFMA, due to the former evaluating matrix entries more rigorously, as well as due to algorithmic differences [34]. However, the quadrature recipes could be further optimized (as noted in Section IV) and more generally, the MLACA-SVD matrix setup time can be readily reduced through parallelization, to which the scheme is well suited.

Acknowledgment

The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the ODNI, IARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation herein.

References

[1] D. S. Holmes, A. L. Ripple, and M. A. Manheimer, "Energy-efficient superconducting computing—Power budgets and requirements," *IEEE Trans. Appl. Supercond.*, vol. 23, no. 3, Jun. 2013, Art. no. 1701610.

[2] M. A. Manheimer, "Cryogenic computing complexity program: Phase 1 introduction," *IEEE Trans. Appl. Supercond.*, vol. 25, no. 3, Jun. 2015, Art. no. 1301704.

[3] C. J. Fourie, "Digital superconducting electronics design tools—Status and roadmap," *IEEE Trans. Appl. Supercond.*, vol. 28, no. 5, Aug. 2018, Art. no. 1300412.

[4] M. Heiligman, "IARPA SuperTools program," IARPA, Washington, DC, USA, Rep. no. IARPA-BAA-16-03, [Online]. Available: https://www.iarpa.gov/index.php/research-programs/supertools.

[5] C. J. Fourie *et al.*, "Coldflux superconducting EDA and TCAD tools project: Overview and progress," *IEEE Trans. Appl. Supercond.*, vol. 29, no. 5, Aug. 2019, Art. no. 1300407.

[6] P. I. Bunyk and S. V. Rylov, "Automated calculation of mutual inductance matrices of multilayer superconductor intergrated circuits," in *Proc. Extended Abstr. Int. Supercond. Electron. Conf.*, 1993, p. 62.

[7] C. J. Fourie, "Full-gate verification of superconducting integrated circuit layouts with InductEx," *IEEE Trans. Appl. Supercond.*, vol. 25, no. 1, Feb. 2015, Art. no. 1300209.

[8] M. M. Khapaev, A. Y. Kidiyarova-Shevchenko, P. Magnelind, and M. Y. Kupriyanov, "3D-MLSI: Software package for inductance calculation in multilayer superconducting integrated circuits," *IEEE Trans. Appl. Supercond.*, vol. 11, no. 1, pp. 1090–1093, Mar. 2001.

[9] M. Khapaev, M. Y. Kupriyanov, E. Goldobin, and M. Siegel, "Current distribution simulation for superconducting multi-layered structures," *Supercond. Sci. Technol.*, vol. 16, no. 1, pp. 24–27, 2003.

[10] K. Jackman and C. J. Fourie, "Tetrahedral modelling method for inductance extraction of complex 3-D superconducting structures," *IEEE Trans. Appl. Supercond.*, vol. 26, no. 3, Apr. 2016, Art. no. 0602305.

[11] M. Kamon, J. K. White, and M. J. Tsuk, "FASTHENRY: A multipole-accelerated 3-D inductance extraction program," *IEEE Trans. Microw. Theory Techn.*, vol. 42, no. 9, pp. 1750–1758, Sep. 1994.

[12] "Whiteley research Inc.," [Online]. Available: http://www.wrcad.com.

[13] K. Nabors and J. White, "FastCap: A multipole accelerated 3-D capacitance extraction program," *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.*, vol. 10, no. 11, pp. 1447–1459, Nov. 1991.

[14] M. Bebendorf, "Approximation of boundary element matrices," *Numerische Mathematik*, vol. 86, no. 4, pp. 565–589, 2000.

[15] S. Kurz, O. Rain, and S. Rjasanow, "The adaptive cross-approximation technique for the 3-D boundary-element method," *IEEE Trans. Magn.*, vol. 38, no. 2, pp. 421–424, Mar. 2002.

[16] M. Bebendorf and S. Rjasanow, "Adaptive low-rank approximation of collocation matrices," *Computing*, vol. 70, no. 1, pp. 1–24, 2003.

[17] K. Zhao, M. N. Vouvakis, and J. F. Lee, "The adaptive cross approximation algorithm for accelerated method of moments computations of EMC problems," *IEEE Trans. Electromagn. Compat.*, vol. 47, no. 4, pp. 763–773, Nov. 2005.

[18] L. Grasedyck, "Adaptive recompression of $\mathcal{H}$-matrices for BEM," *Computing*, vol. 74, no. 3, pp. 205–223, 2005.

[19] M. Bebendorf and S. Kunis, "Recompression techniques for adaptive cross approximation," *J. Integr. Equ. Appl.*, vol. 21, no. 3, pp. 331–357, 2009.

[20] J. R. Phillips and J. K. White, "A precorrected-FFT method for electrostatic analysis of complicated 3-D structures," *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.*, vol. 16, no. 10, pp. 1059–1072, Oct. 1997.

[21] A. C. Yucel, I. P. Georgakis, A. G. Polimeridis, H. Bağcı, and J. K. White, "VoxHenry: FFT-accelerated inductance extraction for voxelized geometries," *IEEE Trans. Microw. Theory Techn.*, vol. 66, no. 4, pp. 1723–1735, Apr. 2018.

[22] T. Orlando and K. A. Delin, *Foundations of Applied Superconductivity*. MA, USA: Addison-Wesley, 1991.

[23] Y. Saad and M. H. Schultz, "GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems," *SIAM J. Sci. Statist. Comput.*, vol. 7, no. 3, pp. 856–869, 1986.

[24] S. Rjasanow and O. Steinbach, *The Fast Solution of Boundary Integral Equations*. Cham, Switzerland: Springer, 2007.

[25] L. Grasedyck and W. Hackbusch, "Construction and arithmetics of $\mathcal{H}$-matrices," *Computing*, vol. 70, no. 4, pp. 295–334, 2003.

[26] M. Bebendorf and R. Grzhibovskis, "Accelerating Galerkin BEM for linear elasticity using adaptive cross approximation," *Math. Methods Appl. Sci.*, vol. 29, pp. 1721–1747, 2006.

[27] K. Frederix and M. Van Barel, "Solving a large dense linear system by adaptive cross approximation," *J. Comput. Appl. Math.*, vol. 234, no. 11, pp. 3181–3195, 2010.

[28] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 3rd ed. Baltimore, MD, USA: The Johns Hopkins Univ. Press, 1996.

[29] P. J. Davis and P. Rabinowitz, *Methods of Numerical Integration*, 2nd ed. Orlando, FL, USA: Academic, 1984.
[30] D. R. Wilton, S. M. Rao, A. W. Glisson, D. H. Schaubert, . M. Al-Bundak, and C. M. Butler, "Potential integrals for uniform and linear source distributions on polygonal and polyhedral domains," *IEEE Trans. Antennas Propag.*, vol. 32, no. 3, pp. 276–281, Mar. 1984.
[31] N. Takeuchi, T. Ortlepp, Y. Yamanashi, and N. Yoshikawa, "Novel latch for adiabatic quantum-flux-parametron logic," *J. Appl. Phys.*, vol. 115, no. 10, 2014, Art. no. 103910.
[32] I. V. Vernik, S. B. Kaplan, M. H. Volkmann, A. V. Dotsenko, C. J. Fourie, and O. A. Mukhanov, "Design and test of asynchronous eSFQ circuits," *Supercond. Sci. Technol.*, vol. 27, no. 4, 2014, Art. no. 044030.
[33] J. M. Tamayo, A. Heldring, and J. M. Rius, "Multilevel adaptive cross approximation (MLACA)," *IEEE Trans. Antennas Propag.*, vol. 59, no. 12, pp. 4600–4608, Dec. 2011.
[34] D. Brunner, M. Junge, P. Rapp, M. Bebendorf, and L. Gaul, "Comparison of the fast multipole method with hierarchical matrices for the Helmholtz-BEM," *Comput. Model. Eng. Sci.*, vol. 58, no. 2, pp. 131–160, 2010.

Ben A. P. Nel received the bachelor's degree in electrical and electronic engineering and the master's degree in electronic engineering from Stellenbosch University, Stellenbosch, South Africa, in 2016 and 2019, respectively. His main research interest is numerical methods for electromagnetic analysis of superconducting integrated circuits.

Matthys M. Botha (S'99–M'04) received the bachelor's degree in electrical and electronic engineering and the Ph.D. degree in computational electromagnetics from Stellenbosch University (SU), Stellenbosch, South Africa, in 1998 and 2002, respectively.

In 2003, he joined the Center for Computational Electromagnetics and Electromagnetics Laboratory (CCEML), University of Illinois at Urbana-Champaign, Champaign, IL, USA, as a Postdoctoral Research Associate. At the end of 2004, he returned to South Africa, where he worked as a Research Fellow at SU and later joined EM Software and Systems-S.A. (Pty) Ltd., developers of FEKO. In 2011, he was appointed within the Department of Electrical and Electronic Engineering, SU, where he currently holds the position of Professor. His main research interest is computational electromagnetics.

Dr. Botha was the Technical Program Chair for the 8th International Workshop on Finite Elements for Microwave Engineering in 2006. He was Vice-Chair of the local organizing committee for ICEAA'12-IEEE APWC-EEIS'12, held in 2012; and was the Regional Delegate for the European Association on Antennas and Propagation (EurAAP) for Africa and the Middle East from 2014 to 2017. He is a member of the ICEAA conference series' Scientific Committee and is an Associate Editor of the IEEE ANTENNAS AND PROPAGATION MAGAZINE.