From January 2011 *High Frequency Electronics*

High-Frequency Algorithmic Advances in EM Tools for Signal Integrity—Part 1

By John Dunn
AWR Corporation

This two-part series discusses new advances in the algorithms underlying EM simulation techniques for signal integrity designers. Part 1 examines fast solution techniques for Method of Moments solvers

Only 30 years ago, electromagnetic (EM) simulation software was not a mainstream tool of the practicing electrical engineer, yet today it is essential for successful circuit design. EM simulation is used throughout the design process—creating and verifying models, checking the layout of the circuit for parasitic coupling issues, and verifying the final circuit's performance. The most obvious reason for the growing use of EM simulation is its increasing ability to simulate large circuits of practical interest, made possible by more powerful computers. What is not as well known is that a number of new conceptual breakthroughs in EM theory have also contributed greatly to the increased power of EM simulators.

The purpose of this two-part article is to introduce the reader to two of the more successful of these ideas. It is important that users of EM software understand the basic algorithms being used if they are to use the tools effectively and avoid common pitfalls. This first article focuses on advances in simulation methods for planar solvers, or as they are sometimes called, 3D planar or 2.5D simulators. We will see how the time required to solve a problem can be dramatically sped up by using fast, compressed, iterative solution methods. The methods have been developed over the past 20 years and are now reaching fruition in commercially available software. The second article explains how fast frequency sweeping allows for accurate solution of a problem over a frequency range with fewer frequency points.

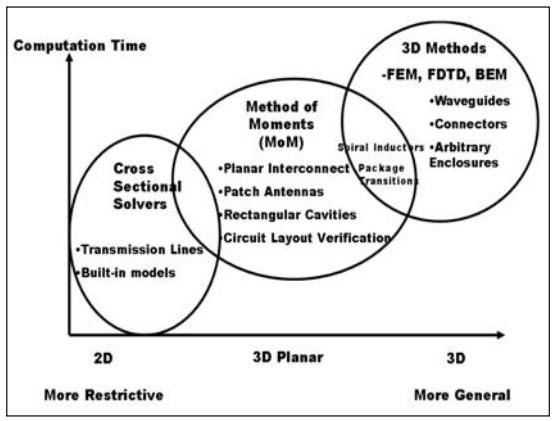

Figure 1 · Comparison of different classes of EM simulators.

The Basic Challenge

EM problems do not scale well with increased problem size. This means that as the problem becomes larger with more unknowns, the computational resources required increase at faster than a linear rate with the problem size. Planar simulators work by meshing the conductors in the circuit up into small cells, usually triangles or rectangles. The most popular EM technologies require the problem to scale by the power of 3, or $O(N^3)$. By this we mean that if the number of cells is doubled, the solution time increases by a factor of eight (2^3). Because of this faster-than-linear scaling, increasing the size of the problem quickly becomes impractical. The fast methods explained in this article scale as $O(N\log N)$, which is much better than $O(N^3)$.

Classes of EM Simulators

Three classes of EM simulation technology are shown in Figure 1—cross sectional

solvers, 3D methods, and the method of moments, which is the focus of this article. The horizontal axis indicates how general a geometry the solver can analyze, and the vertical axis shows relative computation time.

3D methods work by meshing the geometry into small cells, and the electric field is then solved in each cell. The cells are usually 3D triangular shapes known as tetrahedra, but other cell types such as cubes are possible. The finite element method is the most well known, but other good choices exist. The main advantage of these methods is their generality with regard to the geometries they can solve. The main disadvantage is they are computationally very intensive.

Cross sectional solvers are available in many commercial circuit simulators. Their purpose is to characterize transmission lines by taking the cross-section of the line and solving for its electrical parameters per unit length. This is accomplished by solving a simplified set of Maxwell's equations, the quasi-static approximation for the cross-sectional geometry. These solvers are extremely fast and make it easy to get a line's electrical properties. The user normally does not even know that he or she is using electromagnetics when one of these models is used. The disadvantage is that they work only for transmission line models.

Moment methods (MoM), the technique used in the middle bubble of Figure 2, work by solving for the currents on the circuit's conductors. Moment method codes are typically written so that they can solve for currents on horizontal conductors and vertical conducting walls and vias. This is the reason for the terms "2.5D" and "3D" planar solvers. Moment methods require a Green's function in order to calculate the coupling between all the current elements on the conductors. Unfortunately, the Green's function can only be solved for certain geometries. In particular, the conductors must be on infinite, homogeneous, dielectric layers.

It is possible to have an infinite ground plane on the bottom of the dielectric stackup, and a metallic cover if desired. A variation of the method also allows the problem to be bounded by a conducting rectangular box. The requirement of planar, dielectric layers is why the method is not as general as the 3D methods mentioned earlier. However, the planar simulators cover a wide variety of practical technologies—printed circuit boards, modules, and chips, for example. They can usually solve larger problems than full 3D simulators, as they are solving only for currents on the conductors compared to the electric fields throughout the entire space of the problem.

The Solution Time for Conventional Planar Solvers

The method of moments solves for the unknown currents on the conductors. This is accomplished by first meshing the conductors into a finite number of cells and then solving for the unknown current on each cell. The unknown current is simplified on each cell, and a linear approximation is mostly used. The job of the simulator is to find the unknown linear varying current on each cell. A matrix equation is set up and solved for the unknown currents.

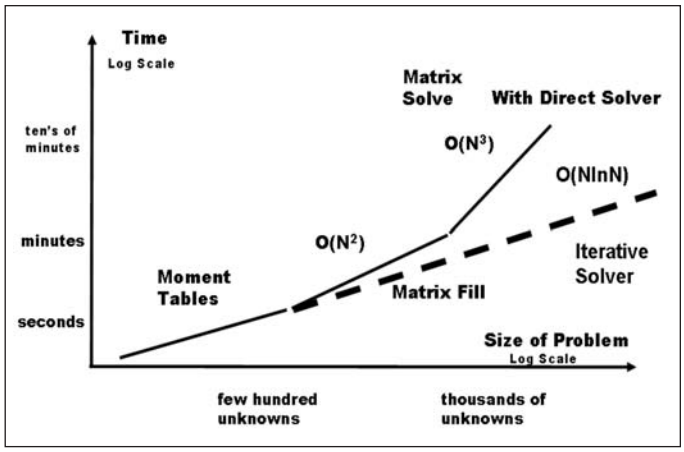

Figure 2 · Simulation times for the MoM.

Each matrix entry gives the interaction between the currents and charges of two cells by using the Green's function. This set of interactions can be thought of as mutual capacitance and inductance between the cells. In addition, there are self-capacitance and inductance terms and resistance terms. Radiation and phase delay between the elements is also included. After the interactions between the mesh elements have been calculated, the matrix can be formed. If there are N cells in the problem, the size of the matrix is $N \times N$ and there are N^2 unknowns. The matrix is solved for the unknown currents on the conductors.

How long does it take for MoM codes to simulate a given size structure? The answer is shown in Figure 2. The Green's functions are first calculated for the given geometry, which takes about $O(N)$ time, where N is the number of meshes (Note that $O(N)$ means order N). The time it takes to simulate with N unknowns is $K \times N$ where K is a positive number. Compare this to $O(N^2)$: The time is equal to $M \times N^2$, where M is a positive number. For a large problem, the $O(N^2)$ process will dominate the $O(N)$ process, but the constant K could be larger than M, so the $O(N)$ could dominate the time of simulation for a small problem. The matrix next needs to be filled (i.e., each element in the matrix must be calculated), which takes $O(N^2)$.

The matrix equation must then be solved, and with conventional techniques, and this requires $O(N^3)$. For a large problem, the solution time will dominate. MoM is an $O(N^3)$ process, so if the number of meshes is doubled, the simulation will take eight times longer. The amount of

consumed memory is also a factor, as the matrix must be stored while the matrix equation is solved. Virtual memory is used when the physical memory is consumed, swapping data in and out of the hard drive. This is a very slow process, taking milliseconds instead of microseconds each time memory is accessed. In practice, the simulation becomes impractical once the computer begins to use virtual memory.

To determine how large a problem can be solved on today's computers, assume each unknown is represented by 16 bytes of data (double-precision complex floating point). 1000 unknowns requires 1000 × 1000 × 16 = 16 Mbytes of RAM, and 10,000 unknowns takes 1.6 GBytes of RAM. A 32-bit computer can access up to 4 GBytes of RAM, with much of that reserved for the operating system and other applications, and a practical limit for RAM available for the simulation is 2 GBytes, which corresponds to about 10,000 cells. This memory limitation is overcome by 64-bit operating systems, and 16 GBytes of RAM is typically installed in these computers.

Fast Iterative Solvers for MoM

The biggest bottleneck in simulating large problems with MoM simulators is solving the matrix equation. A direct solve, which requires the direct solution of the matrix, is $O(N^3)$, so doubling the number of unknowns increases the simulation time by a factor of 8. This produces a near order-of-magnitude increase in solution time and memory requirements and does not lend itself well to larger physical problems. Fortunately, new developments in solver technology have overcome several hurdles and have reduced solution time. Globally, these techniques attempt to bypass a direct solution of the entire matrix and in doing so avoid the $O(N^3)$ computation cost.

The methods work by iteratively solving a smaller, approximate representation of the matrix. These methods rely on a well-conditioned matrix. The condition number of a matrix is a figure of merit that indicates how easily the matrix can be solved with finite precision arithmetic on a computer. A poorly conditioned matrix (large condition number) indicates that the solution will not be accurate, or perhaps not even solvable. EM numerical methods usually result in poorly conditioned matrices because of the nature of the equations. The biggest challenge for developers of fast EM solvers is overcoming the poor conditioning of the matrices that arise for various reasons. To understand these poorly conditioned matrices and how to mitigate them, it helps to focus on one typical cause of poor conditioning, the so-called DC catastrophe.

MoM codes solve for currents and charges on conductors, and there is one equation that involves both current and charge. Current and charge are not independent quantities for frequency domain problems. A current changing in space means that there is a charge buildup.

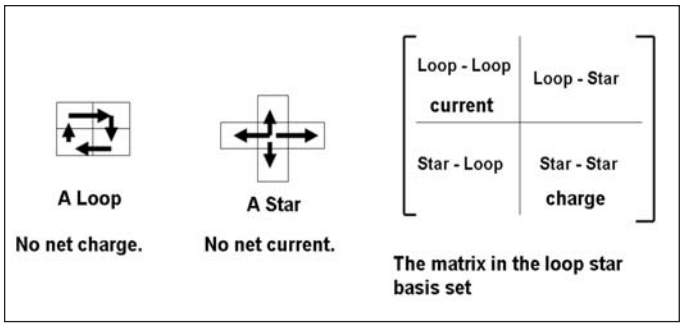

Figure 3 · Loop-star preconditioning of cells.

For example, when current goes around a bend in a line there is charge buildup at the corner of the bend.

At DC, charge and current are independent of each other, with charge the source of electric fields and current the source of magnetic fields. It is only when the frequency is non-zero that electric and magnetic fields produce each other and current is related to charge. If solving one equation for current and charge at DC, this equation has no unique solution. At low frequencies, the solution is problematic and the condition number of the matrix is very large. The ultimate example of this problem is at DC, where there is not even a unique theoretical solution—resulting in the "DC catastrophe." This problem is common to almost all EM simulators and at low frequencies is not practically solvable.

"Preconditioners" that operate on the problem in a way that improves the condition number before trying to solve it can be used to overcome this problem. The idea is to take the meshes (or basis functions) and recombine them into something that allows the charge and current terms to be separated in the equation. This allows the designer to approximately solve for the two separate problems of charge and current, producing the approximate DC electrostatic and magnetostatic solutions. These approximate solutions can then be used as a good way to precondition the original matrix to make its solution practical.

There are many forms of preconditioners. This discussion will focus on the loop-star type, which gets its name because of its regrouping of the cells, as shown in Figure 3. The loop of cells has no net charge, as the current is flowing in a closed loop. The star grouping has no net current, as the current modeled by this configuration of cells flows outward from the center. In the loop-star preconditioner, the problem can be reformulated by using the loop configuration and the star configuration as two new basis functions to describe the problem, This is much like the way vectors (1,0) and (0,1) can be used to describe points in the Cartesian plane. The preconditioner should thus give the matrix a lower condition number because the matrix produced by the new basis should have more zero

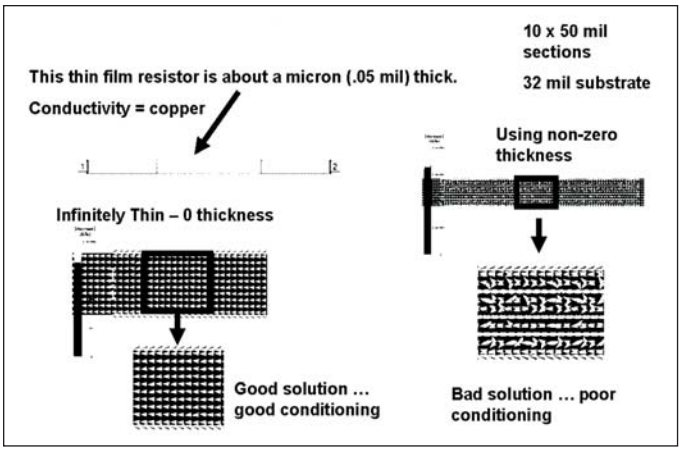

Figure 4 · Thick line that is poorly conditioned.

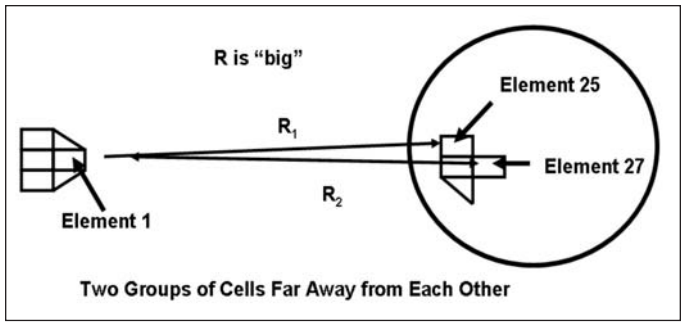

Figure 5 · Two groups of cells in a multipole method.

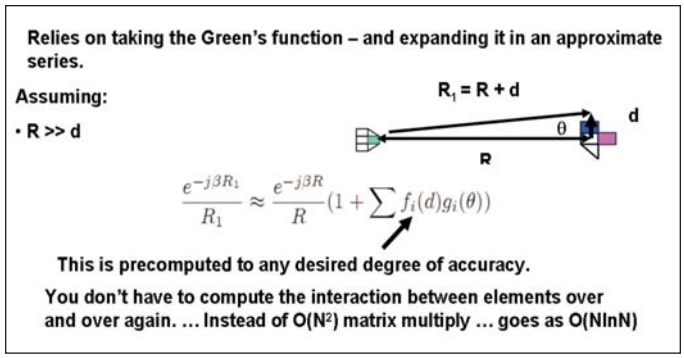

Figure 6 · Expanding out the Green's function in the multipole method.

entries then the initial formulation.

Poorly conditioned matrices can occur in a variety of other ways besides low-frequency issues. For example, there could be a situation in which a transmission line has non-zero thickness. There are currents on both the top and bottom sides of the line, but if the line is not very thick, the currents are almost on top of one another (Figure 4).

If the line becomes infinitely thin, there are two unknown currents to be solved for but only one equation for the total current, so there is no unique solution to the problem. Figure 4 shows how the currents become very poorly behaved as a result of the poor conditioning of the matrix. It is important to avoid thick lines that are too thin, or in the case of boundary element methods, regions that are too thin. Modern planar simulators therefore need to employ a variety of preconditioners to cover the various causes of poorly conditioned matrices.

The Compressed, Fast Solve Algorithm

Once the matrix has been properly conditioned, the next challenge is to solve the better-conditioned problem in faster than $O(N^3)$ time. Methods have been developed that can go as fast as $O(N \ln N)$ time. The two most common methods are fast multipole techniques and compressed matrix techniques. These methods are based on the fact that the interaction between groups of cells that are far apart have two important properties: the interactions between the groups are small, and they are all about the same value. To understand this concept, referring to the two groups of cells separated by a large distance in Figure 5, imagine the size of the interaction between a cell in the left group and a cell in the right group, R. Compare it to the interaction between the same cell in the left group and a neighboring cell in the right group. They (R1 and R2) are probably about the same. This fact is exploited in fast multipole methods (Figure 6) in which the Green's function is expanded out as a series in d, where d is the local distance of the second group from its center of mass.

The fast multipole method has become extremely popular for antenna and radar codes but has three limitations when applied to SI problems. First, the method relies on large distances compared to wavelength, which is normally not the case for an SI problem. Second, the method relies on expanding the Green's function out into a series of functions, but codes for SI engineers have more complicated Green's functions, making this procedure difficult. Third, fast multipole methods can become numerically unstable when the size of the problem is not electrically large.

Fast matrix compression methods are more commonly used in the SI community. These methods are similar to multipole methods because they rely on groups of cells that are far away and have about the same interaction strength as shown in Figure 7. However, they do not rely on analytical manipulations of the Green's function. Rather, they numerically exploit the fact that the matrix entries for a given block of the matrix are about the same because of the previous observation that in this given block the interactions are between two far away cells. For example, Figure 7 shows a block in the matrix with cells 25 and 27 interacting with cell 1. Elements A1,25 and

A1,27 will have about the same values as the interaction is between cells in two different blocks.

Since the elements in the block all have about the same values, it is possible to represent the elements with many fewer members. Using mathematical algorithms very similar to those from compression standards like JPEG or MPEG, the number of members needed to represent the block can be reduced. For the world of images, compression methods permit digital or video cameras to represent pictures using less data than the uncompressed original. The matrix representing the photograph or video is represented by a series of reduced matrices that provide an approximation that is "good enough" in most cases. Of course, what is actually good enough is in the eye of the beholder, but suffice it to say that this technique can be applied also to SI problems to gain measurable improvements in EM performance while achieving acceptable accuracy.

Once the matrix blocks have been reduced in size using these compression algorithms, the matrix is solved using an iterative solver. This is perhaps the key difference from the direct solve technique. The iterative solver never solves the original matrix equation and is therefore not necessarily bound by the $O(N^3)$ scaling but instead multiplies the compressed matrix by a vector. If performed correctly, the entire process can potentially proceed in $O(N\ln N)$ time. It should be noted that to get $O(N\ln N)$ scaling the actual matrix is never calculated, as this takes $O(N^2)$ time. Rather, successive approximations of the matrix are calculated in a consistent manner. Achieving overall improvement with a particular implementation of an iterative solver depends on whether the whole is better than the direct solver.

All the pieces of the iterative solver must when working in concert take less time than the direct solver with its brute force approach. The exact increase in speed achieved by the iterative solver depends on the details of the geometry being studied. The method relies on groups of blocks that are far from one another. Geometries in which there are a large number of nearby cells (overlapping shapes on different layers in a package, for example) will not compress as well as geometries in which most shapes are spread out on one layer.

With all the components of an iterative solver working properly and for problems of a few thousand unknowns, an increase in speed should be achieved when compared to the direct solver shown in Figure 2. However, the direct solver continues to perform well, especially at RF and microwave frequencies with several hundred unknowns. This point should not be overlooked because some modern commercial EM solvers incorporate both direct and iterative solvers with algorithms that analyze a particular problem and then select the fastest or most accurate method.

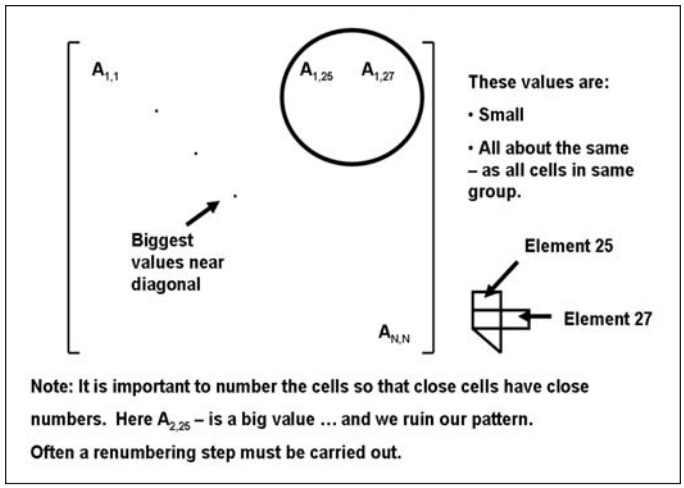

Figure 7 · A matrix block between two groups of cells.

Challenges Remaining

Fast solvers promise to give designers a tool that can solve complex problems that are simply not approachable with traditional methods. However, a number of research challenges remain. As discussed in this article, preconditioners must be used for the matrices to be well conditioned and the methods to work. Right now, there is no one best preconditioner. It is also not always clear which one will work well for a given problem. As the methods enter into more into mainstream usage, this problem will become more relevant to the designer. Therefore, the challenge for EM tool developers is to make the methods more robust and automated.

Next Month

The second article in this two-part series will describe ways to reduce solution times by using fewer frequencies while maintaining the frequency resolution of the resulting dataset. It will address intelligent selection of simulation frequencies such as fast frequency sweeping algorithms that reduce the number of frequencies required to obtain the desired frequency range as well as efficient methods to map EM solutions from the frequency-domain to the time-domain.

Author Information

Dr. John Dunn is a senior application engineer with AWR whose area of expertise is electromagnetic simulation and modeling. He was a principal engineer at Tektronix for four years before joining AWR and was a professor of electrical engineering at the University of Colorado for 15 years. He received his BS degree in physics from Carleton College, and his MS and PhD degrees in applied physics from Harvard University.

Readers should address questions or comments to: info@awrcorp.com