Optimization of computational speed for BE method of coil design

C. T. Harris[1], W. B. Handler[1], and B. A. Chronik[1]
[1]Physics and Astronomy, University of Western Ontario, London, Ontario, Canada

Introduction

The Boundary Element (BE) method [1] is proving to be an extremely powerful and versatile tool in the electromagnetic design of gradient, shim, and shielding coils. Because of this, we have been strongly motivated to optimize the computational implementation of the algorithm. We have found that the speed of the technique can be significantly increased relatively easily. In this study, we describe the most important steps in optimizing the implementation of this algorithm, and we show that it is fully capable of producing detailed coil designs in only tens of seconds.

Method

We have gone back through our code archives and picked four time points. At each of which we were using the BE method for magnet design. The first version corresponded to the original implementation, written purely in Matlab in an object-oriented class-based code. In the second version the two slowest functions were rewritten in "C". Specifically, we reprogrammed the calculation of the node basis functions (equation 9 of [2]), and the calculation of the power dissipation matrix (equation 4 of [1]). In the third version, the next slowest function, i.e. calculating the field matrix (equation 14 of [2]), was also written in C-code. In the fourth version of code we implemented a parallelization scheme for calculating the field matrix using Grand Central Dispatch, an addition to the gcc library on the apple platform that has been submitted to be used in ansi c. This fourth version is our current version of the code. In this version, the most time-consuming step involves the matrix inversion. This function utilizes the matrix inversion method in Matlab, which is already optimized and parallelized for speed.

In order to evaluate the progressive improvement in calculation speed between different versions of code, a standard problem of designing a gradient coil over a cylinder was used. Three distinct finite element meshes were created over a cylinder 40 cm in diameter with a total length of 1.2 m (z-direction) using COMSOL. The meshes consisted of 310, 1398, 4282 node points with 588, 2732, 8460 triangular elements, and were denoted as 'Fine', 'Extra Fine', and 'Extremely Fine' (using the COMSOL meshing nomenclature) respectively. Four separate sets of target points were chosen so as to produce an x-gradient coil. They consisted of 16, 64, 400, and 8000 points. This was done in order to gauge how computation time increases with problem complexity. Therefore, a total of 48 simulations were completed and the calculation time stored for each case. All calculations were done on a 2009 I7 powermac with 16 GB ram.

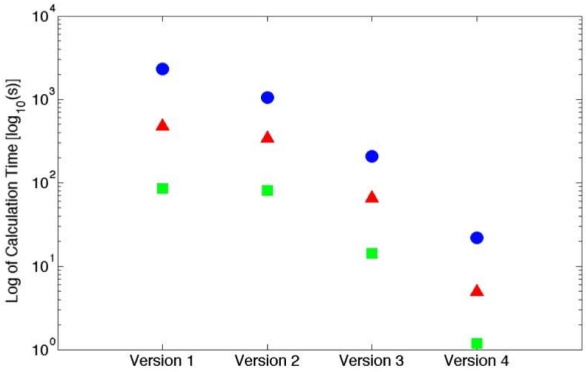

Figure 1. Logarithm of calculation time for each version of Boundary Element method code. Times shown for 8000 target points and an 'Extremely Fine' mesh (blue circle); 'Extra Fine' mesh (red triangle); and 'Fine' mesh (green square).

Results

Figure 1 displays the calculation times for each of the four versions of code for 8000 target points and the extremely fine mesh (blue circle); the extra fine mesh (red triangle); and the fine mesh (green square). Figure 2 shows the resultant stream function calculated with the final iteration of code for the 'Fine' mesh and 8000 target points. Super-imposed over the stream function is the coil wire pattern. We found that the calculation speed between the original version of code and our speed optimized code increased by a factors of approximately 30, 60, and 105 for the 'Fine', 'Extra Fine', and 'Extremely Fine' meshes respectively, regardless of whether using 16, 64, or 400 target points. However, when the number of target points was increased to 8000 (Fig. 1), the calculation speed increased by a factor of approximately 70 (~1.4 min to ~1.2 sec) and 95 (~8 min to ~5 sec) for the 'Fine' and 'Extra Fine' meshes respectively. The speed-up factor for the 'Extremely Fine' mesh and 8000 target points remained at approximately two orders of magnitude (from ~39 min to ~22 sec).

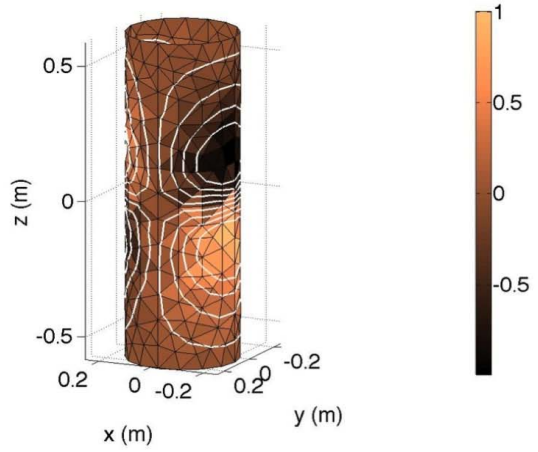

Figure 2. Wire pattern (white) for x-gradient coil produced with speed optimized code for the 'Fine' mesh and 8000 target points. Super-imposed underneath is the coil's stream function.

Discussion

It is common practice in coil design problems to begin with a coarse mesh and then increase the mesh 'fineness' as a more satisfactory solution is identified. With optimized code, the calculation time for a 'Fine' mesh and 8000 target points is on order of 1 second. This means that it is now possible to use the BE method within an optimization loop over multiple geometries or sets of target points.

References and Acknowledgements

The authors acknowledge support from NSERC Canada and NIH R01 MH080913 (PI Spielman).

1. M. Poole, R. Bowtell, *Concepts Magn. Reson. B.* **31B**, 162-175 (2007).
2. Lemdiasov R, Ludwig R, *Concepts Magn. Reson. B.* **26B**, 67-80 (2005).