Forward Solving in Electrical Impedance Tomography with Algebraic Multigrid Wavelet Based Preconditioners

A. Borsic\textsuperscript{1}, R. Bayford\textsuperscript{2}

\textsuperscript{1}Thayer School Of Engineering, Dartmouth College, 8000 Cummings Hall Hanover, NH 03755, US
\textsuperscript{2}Department of Natural Sciences, Middlesex University, London
E-mail: Andrea.Borsic@Dartmouth.edu

1. Abstract

Electrical Impedance Tomography is a soft–field tomography modality, where image reconstruction is formulated as a non-linear least-squares model fitting problem. The Newton-Raphson scheme is used for actually reconstructing the image, and this involves three main steps: forward solving, computation of the Jacobian, and the computation of the conductivity update. Forward solving relies typically on the finite element method, resulting in the solution of a sparse linear system. In typical three dimensional biomedical applications of EIT, like breast, prostate, or brain imaging, it is desirable to work with sufficiently fine meshes in order to properly capture the shape of the domain, of the electrodes, and to describe the resulting electric filed with accuracy. These requirements result in meshes with 100,000 nodes or more. The solution the resulting forward problems is computationally intensive. We address this aspect by speeding up the solution of the FEM linear system by the use of efficient numeric methods and of new hardware architectures. In particular, in terms of numeric methods, we solve the forward problem using the Conjugate Gradient method, with a wavelet–based algebraic multigrid (AMG) preconditioner. This preconditioner is faster to set up than other AMG preconditioners which are not based on wavelets, it does use less memory, and provides for a faster convergence. We report results for a MATLAB based prototype algorithm an we discuss details of a work in progress for a GPU implementation.

2. Background

Electrical Impedance Tomography is a form of soft field tomography where image reconstruction is typically formulated as a non-linear least squares Tikhonov regularized inverse problem:

$$\sigma_{\text{rec}} = \arg \min_{\sigma} \frac{1}{2} \| V(\sigma) - V_{\text{meas}} \|^2 + \alpha \frac{1}{2} \| L (\sigma - \sigma^*) \|^2$$

(1)

where $\sigma_{\text{rec}}$ is the vector of conductivities to be estimated $\sigma$ is the vector of model conductivities, $V(\sigma)$ is the vector of electrode voltages resulting from the forward solver, $V_{\text{meas}}$ is the vector of measured electrode voltages, $\alpha$ is the Tikhonov factor, $L$ is a regularization matrix, and $\sigma^*$ is a reference conductivity distribution. Equation (1) is usually solved in a iterative fashion, by application of the Gauss Newton method:

$$\delta \sigma_n = -(J_n^TJ_n + \alpha L^T L)^{-1} [J_n^T (V(\sigma_n) - V_{\text{meas}}) - \alpha L^T L (\sigma_n - \sigma^*)]$$

(2)
where \( n \) is the iteration number, \( \delta \sigma_n \) is the conductivity update for iteration \( n \) and \( J_n \) is the Jacobian of the forward operator \( V(\sigma) \) calculated for \( \sigma = \sigma_n \). EIT reconstruction consists therefore in the repeated application of (2) to update an initial conductivity guess, typically a uniform conductivity distribution, forming therefore a reconstructed image. Three main computational steps are associated with (2): the computation of the forward solution \( V(\sigma_n) \), the computation of the Jacobian \( J_n \), and the solution of the dense linear system represented by (2) itself. Typically the solution of the forward problem and the computation of the Jacobian matrix are the most computationally intensive steps of the reconstruction and different papers in literature address the problem of speeding up forward solutions. Techniques such as Finite Elements and Finite Differences are normally used for discretizing the imaging domain and for solving the associated PDE, resulting in a sparse linear system. The system of sparse equations can be solved either by direct or by iterative methods. Iterative solvers, based on the Conjugate Gradients Method with preconditioning, have been studied by Soleimani [1] and by Horesh [2]. Soleimani considered an Algebraic Multigrid Method based on the traditional Algebraic Multigrid Method (AMG) of Ruge and Stüben [3]. The AMG (see next Section for more detail) preconditioning showed in this case considerable gains in terms of speed of convergence with respect to preconditioning the Conjugate Gradients (CG) method with incomplete Cholesky factorizations. Horesh considered an inverse-based multi-level preconditioning from the ILUPACK library, showing again significant gains compared to traditional preconditioning as the incomplete Cholesky factorization. Borsic [4] considered direct solvers. In the context of tomographic applications, where the same forward problem is solved for several excitations, direct solvers can prove to be faster than iterative solvers. Direct solvers are based on matrix factorizations, and once this is accomplished, multiple solutions for different right hand sides can be computed with very little cost. For three-dimensional problems, with meshes with approximately 100,000 nodes, Borsic showed that PARDISO [5] can be significantly faster than the iterative methods proposed in literature. Direct solvers have the inconvenience that they require more memory than iterative solvers, as they need to store the factored matrix, and they lend themselves less well to parallelization. Bayford [6] proposed the use of algebraic multigrid methods for preconditioning iterative solvers, where the interpolation between different levels of the grid uses wavelet-based operators (WAMG) [7]. Such an algorithm lends itself very well to parallelization, as iterative algorithms are trivial to parallelize and as wavelet-based multigrid preconditioners have also properties that are desirable. Traditional AMG methods [3] require the inspection of the whole system matrix to determine certain weight factors. If the system matrix is stored across different computing nodes, this requires a certain amount of inter communication. WAMG methods do not require this inspection, and the resulting communication between different parallel execution units. In our work we consider therefore WAMG methods as ideal candidates for the implementation on GPUs, thanks to their good parallelization properties.

3. Algebraic Multigrid Preconditioning for Conjugate Gradients

Algebraic Multigrid Methods can be used as preconditioners for iterative methods like the Conjugate Gradients algorithm. In Figure 1 we report the pseudo code for the Preconditioned Conjugate Gradients algorithm (PCG) [8]. In explicit preconditioning, the conjugate search direction \( d(k) \) is computed by solving \( M^{-1}r(i) \) (see Figure 1), where \( M \) is the preconditioner and \( r(i) \) is the residual at step \( i \). AMG methods can used to precondition PCG by computing \( M^{-1}r(i) \) with an AMG v-cycle [3]. The v-cycle, whose code is reported in Figure 2, is a multigrid method for approximating the solution of a linear system. The v-cycle is based on the setup of an interpolation scheme based on restriction operators \( P^T \) and prolongation operators \( P \) that allow to respectively interpolate variables onto a coarser grid or to project them onto a finer grid. The operators allow also to establish a hierarchy of matrices \( M_{k+1} = P_k^T M_k P_k \), which are a coarser and coarser (smaller in size) representations of \( M_k \). In general the multigrid method
Preconditioned Conjugate Gradients Algorithm: Solve $Ax = b$

$r(0) = b - Ax(0)$
$d(0) = M^{-1}r(0)$
$\alpha(i) = \frac{r^T(i)M^{-1}r(i)}{d^T(i)M^{-1}d(i)}$
$x(i+1) = x(i) + \alpha(i)d(i)$
$r(i+1) = r(i) - \alpha(i)Ad(i)$
$\beta(i) = \frac{r^T(i+1)M^{-1}r(i+1)}{r^T(i)M^{-1}r(i)}$
$d(i+1) = M^{-1}r(i+1) + \beta(i)d(i)$

Figure 1. Pseudo code for the Preconditioned Conjugate Gradient algorithm.

presents very rapid convergence properties compared to other iterative methods, and can be used on its own to solve systems of equations or as a preconditioner, as in our case. Figure 3 reports convergence results from a MATLAB based implementation of the wavelet AMG (WAMG) preconditioned PCG solver. The graphs represent the plot of the residual during the iterations of the PCG algorithm. As a comparison we report convergence graphs for PCG with incomplete Cholesky preconditioning with drop tolerances of $10^{-1}$, $10^{-2}$, and $10^{-3}$ and for the unpreconditioned CG algorithm. The left part of Figure 3 is relative to the forward solution of a EIT problem on a mesh with approximately 59,000 nodes, and the right part to a mesh with approximately 95,000 nodes. WAMG shows a much faster convergence than the other preconditioners, solving the problems in less than 20 iterations. The setup of the PCG with incomplete Cholesky factorization requires lengthier times and much more memory than WAMG. For example, for the mesh with 95,000 nodes, WAMG requires 2.50MB of additional memory for the structures of the v-cycle, while the incomplete Cholesky factorization, with a drop tolerance of $10^{-3}$, requires 25.27MB of additional memory to store $M$. The system matrix of this problem stores in 10.72MB. WAMG requires thus 33% additional memory, while the incomplete Cholesky factorization, with a small threshold, requires 235% of additional memory to store the preconditioner, which is an undesirable effect of the “fill-in” phenomenon, where the factored matrix has a less sparser structure then the original matrix. The WAMG method requires thus shorter setup times and less memory than incomplete Cholesky factorizations, and results in better convergence rates.

4. Conclusions
Given the positive results form the MATLAB based prototype of the WAMG PCG, and thanks to its good parallelization properties, we are currently implementing the algorithm on a GPU architecture. We are using an NVIDIA Tesla S1070, a multi-GPU computing solution with 4 GPUs, each with 240 cores, which is capable of running 3840 threads at the same time. GPUs have excellent performances for single precision computation, and they are less powerful in double precision, but they can still offer significant gains compared to CPUs also in double precision. We are in the process of implementing the main linear algebra functions that are needed to implement the full WAMG PCG algorithm. The main operations involved are element-wise products of vectors, sparse-matrix times vector, and sparse-matrix times sparse-matrix operations. For the linear algebra blocks we have implemented at this stage we have obtained speed gains in excess of 10 times compared to running the same code on a octal Xeon workstation. These results are obtained so far on a single GPU of the 4 GPUs on board of the Tesla S1070. We expect that
AMG V-Cycle: Solve \( d_k = M_k^{-1} r_k \)

procedure v-cycle(\( M_k, r_k, d_k, k \))
\[ d_k = \text{smoother}(M_k, r_k, d_k) \]
if ( \( k \neq \text{max-grid-level} \))
\[ z_k = r_k - A_k d_k \]
\[ M_{k+1} = P_k^T M_k P_k \]
\[ d_{k+1} = 0 \]
\[ v\text{-cycle}(M_{k+1}, P_k^T z_k, d_{k+1}, k + 1) \]
\[ d_k = d_k + P_k d_{k+1} \]
\[ d_k = \text{smoother}(M_k, r_k, d_k) \]

Figure 2. Pseudo code for the AMG V-Cycle.

(a) Convergence plot, 59K unknowns problem.  (b) Convergence plot, 95K unknowns problem.

Figure 3. Convergence plots for two forward EIT solutions, on the left for a mesh with 59,000 nodes, and on the right for a mesh with 95,000 nodes. The graphs show the reduction for the residual for the Conjugate Gradients method with no preconditioning (CG), with incomplete Cholesky preconditioning with drop tolerances of \( 10^{-1}, 10^{-2}, \) and \( 10^{-3} \) (CHOLINC), and with wavelet AMG preconditioning (WAMG). WAMG shows better performance compared to CG and CHLINC, while requiring smaller setup times and less memory.

the final multi-GPU implementation will result in an FEM solver with superior computational performances thanks to the numerical and parallelization properties that WAMG PCG shows.

References
[4] Borsic A, Hartov A and Paulsen K D 2008 9th International Conference on Biomedical Applications of Electrical Impedance Tomography (EIT 2008), Hanover NH