ArjunAir: Updating and parallelizing an existing time domain electromagnetic inversion program

Patrick Belliveau, Dr. Colin Farquharson, Dr. Ronald Haynes*

Memorial University of Newfoundland

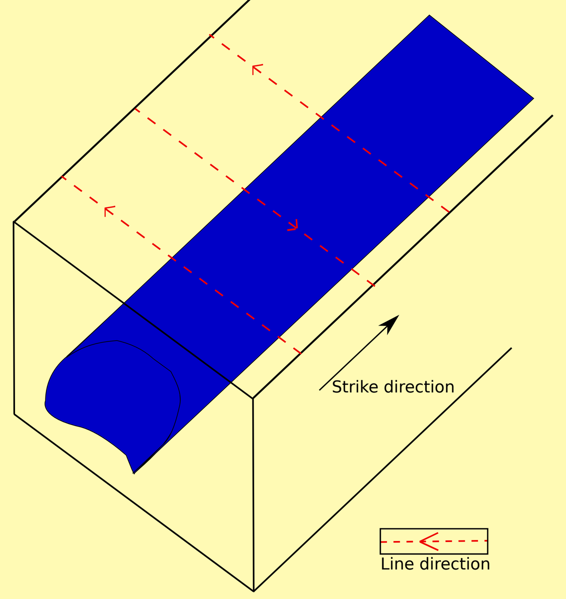
The Forward Model

Forward problem overview

Arjun Air is a 2.5D airborne electromagnetic inversion program developed by CSIRO in partnership with the AMIRA consortium [4]. It inverts for a 2D conductivity distribution. It solves for the along-strike components of the secondary electric and magnetic fields using a nodal finite element method with isoparametric quadrilateral elements. The secondary field Maxwell equations in the frequency domain are

$$\nabla \times \mathbf{E}^s = -i\mu\omega\mathbf{H}^s$$

$$\nabla \times \mathbf{H}^s - \tilde{\sigma}(\mathbf{E}^s = \mathbf{E}^p).$$

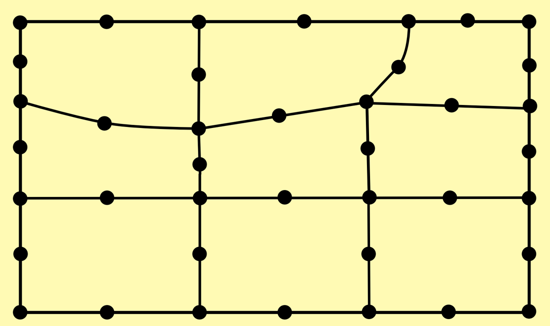


Typical 2.5D scenario

Breaking the equations into components and Fourier transforming with respect to the along-strike coordinate gives a system of two PDEs for the along strike field components

$$\nabla \cdot (\frac{\sigma}{k_e^2}\nabla \tilde{E}_y^s) - ik_y\nabla \cdot (\mathbf{A}\nabla \tilde{H}_y^s) - \sigma \tilde{E}_y^s = \sigma_a \tilde{E}_y^p - ik_y \nabla \cdot \left[\frac{\sigma_a}{k_e^2}\begin{pmatrix}\tilde{E}_x^p \\ \tilde{E}_z^p\end{pmatrix}\right],$$

$$\nabla \cdot (\frac{1}{k_e^2}\nabla \tilde{H}_y^s) + \frac{k_y}{\omega\mu}\nabla \cdot (\mathbf{A}\nabla \tilde{E}_y^s) - \tilde{H}_y^s = \partial_x\left(\frac{\sigma_a}{k_e^2}\tilde{E}_z^p\right) - \partial_z\left(\frac{\sigma_a}{k_e^2}\tilde{E}_x^p\right).$$



Isoparametric quadrilateral mesh

Computing the primary field

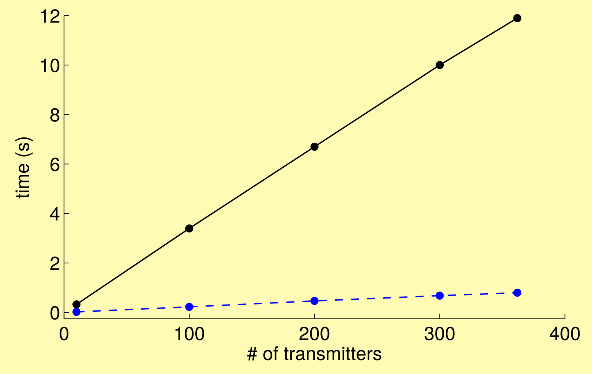
The frequency domain primary electric field is

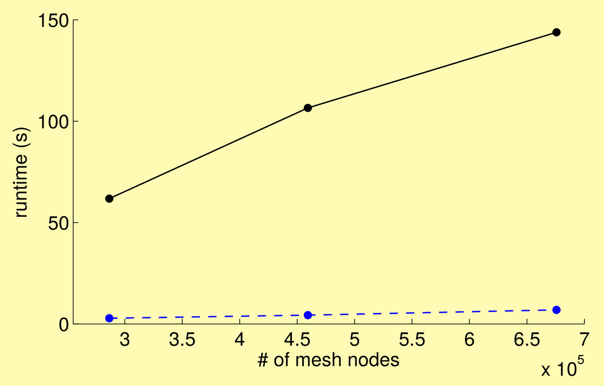
$$\mathbf{E} = \frac{i\omega\mu}{4\pi r^3}\left[y\cos\theta\hat{\mathbf{x}} + (z\sin\theta - x\cos\theta)\hat{\mathbf{y}} + y\sin\theta\hat{\mathbf{z}}\right]$$

We need it in the spatial wavenumber domain. We compute Fourier transform numerically. E.g.

$$\tilde{E}_y = \frac{i\omega\mu}{4\pi}(z\sin\theta - x\cos\theta)\int_{-\infty}^{\infty}\frac{\mathrm{e}^{-ik_y y}}{(\rho^2+y^2)^{3/2}}\mathrm{d}y$$

where, $\rho^2 = x^2 + z^2$. These integrals were originally computed in ArjunAir by digital filtering at every mesh point for every transmitter. However, depend only on ρ and not on the across strike coordinates individually. Compute integral at set of ρ values and interpolate.





Primary field interpolation speedup

Improvements

Parallel Computing Paradigms

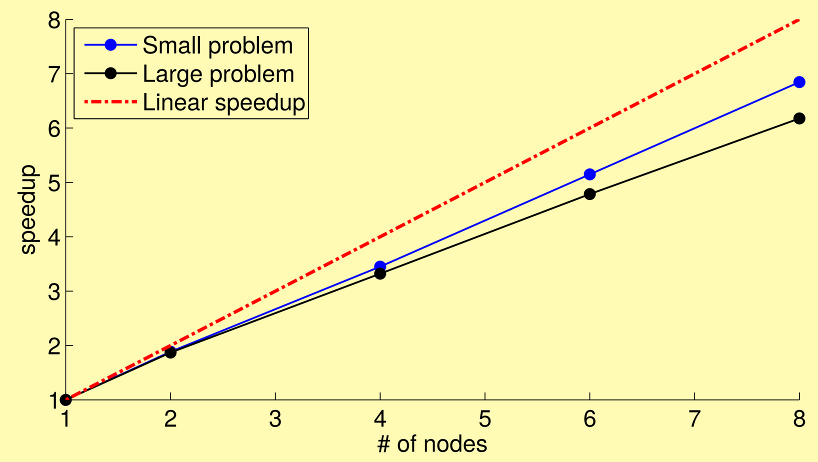
	Single instruction	Multiple instruction
Single data	SISD	MISD
Multiple data	SIMD	MIMD

Flynn's taxonomy

We used MPI for distributed memory parallelization and OpenMP for shared memory. All code was written in Fortran.

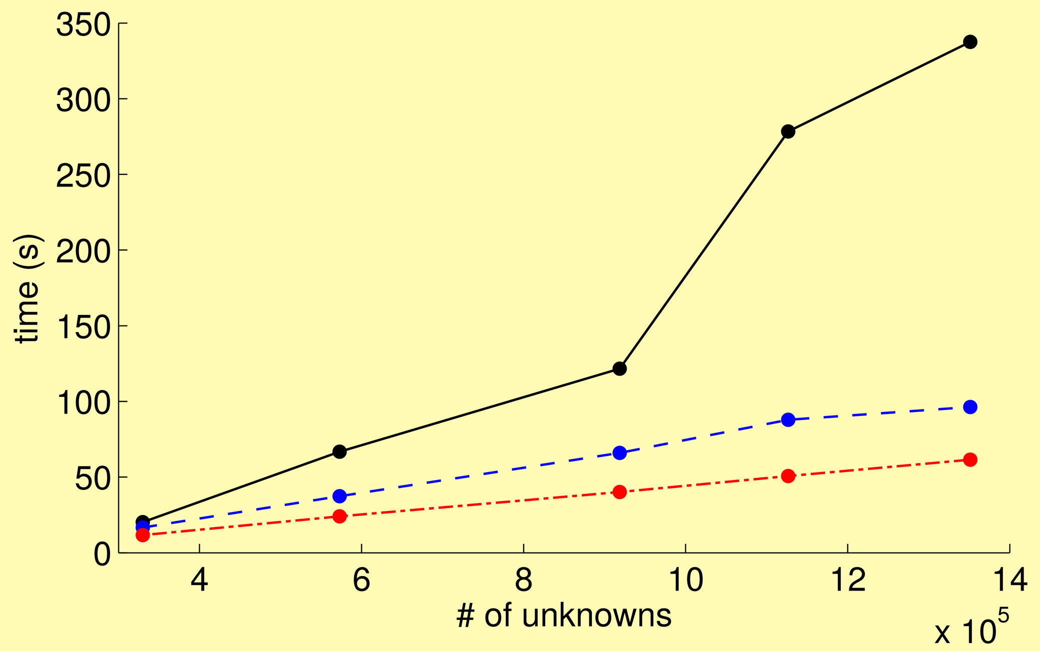
Sources of parallelism

Arjun Air solves for the along strike secondary fields at 21 values of the spatial wavenumber. Solves at different wavenumbers are independent. We parallelize over wavenumbers.

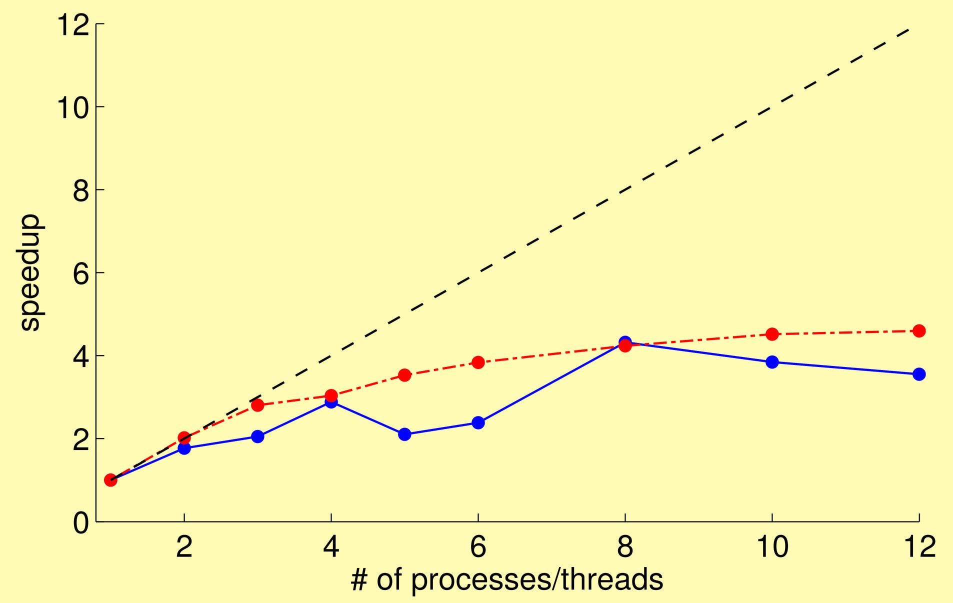


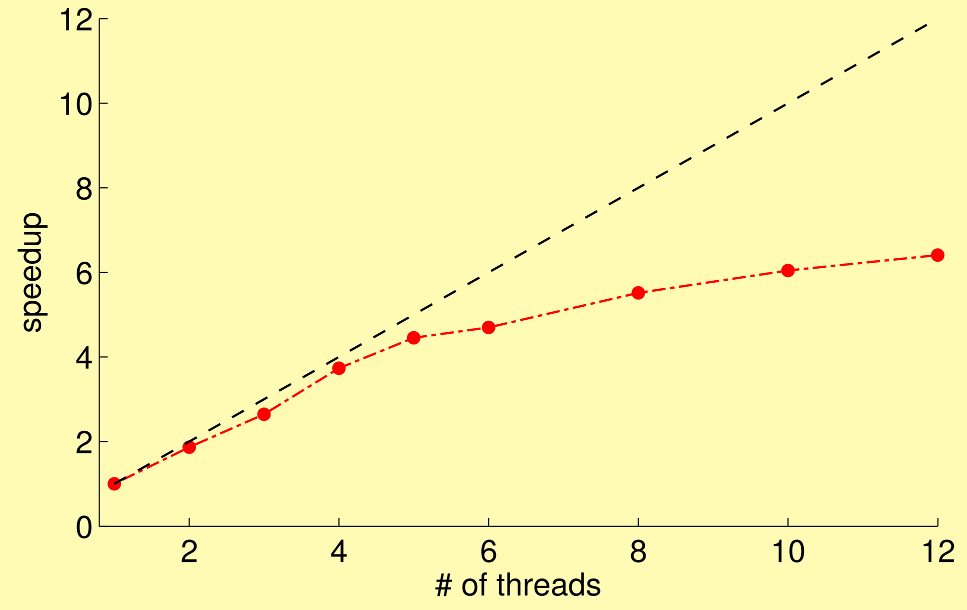
Solving the finite element equations

Need significant computing resources to solve more than a few wavenumber domain problems in parallel. However, we can also look for parallelism and higher performance within each solve. After the primary field, solving **Ax = b** is the main bottleneck. We used the MuMPS [1] and Pardiso [3] sparse direct solvers. Additionally, wherever possible, we used BLAS and LAPACK routines from the Intel MKL library to speedup linear algebra computations.



Improvements





Top) MuMPS and Pardiso speedup on a large mesh with only 100 sources. Bottom) Pardiso spee dup with 362 sources

Inversion algorithm

ArjunAir's inversion algorithm is a variant of the Levenberg-Marquardt algorithm. It uses singular value damping to stabilize the inversion. The singular value decomposition is expensive both to compute and to store in memory. We formulated the inverse problem using the standard Levenberg-Marquardt algorithm. At each iteration it solves the linearized least squares minimization problem

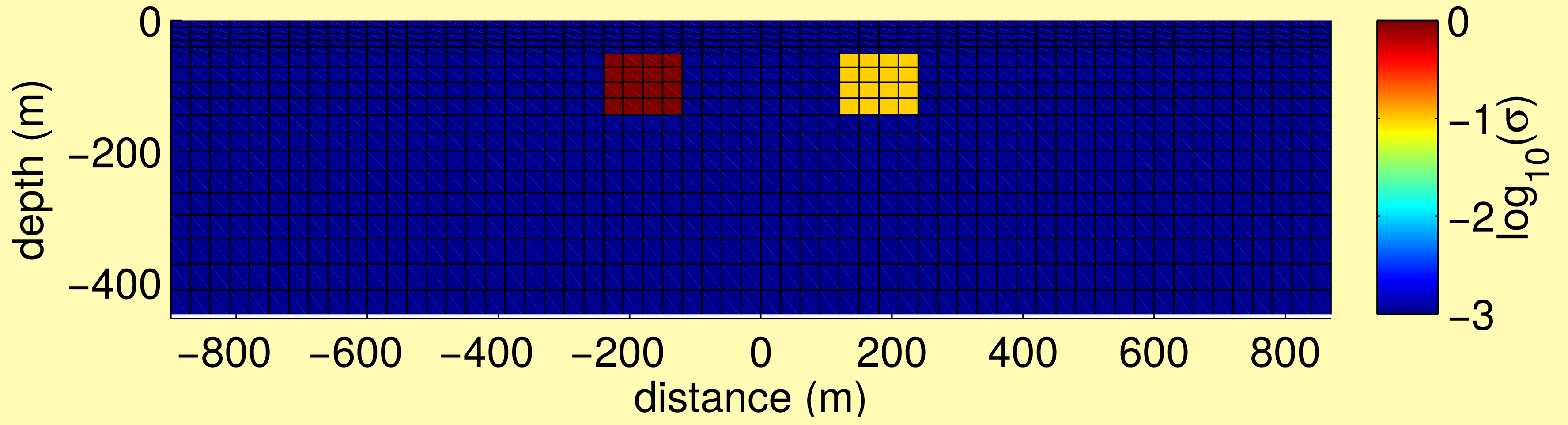
$$\underset{\boldsymbol{\delta}_\mathbf{m}}{\text{minimize}} \quad \|\boldsymbol{\varepsilon} - \mathbf{J}\boldsymbol{\delta}_m\|^2 + \nu^2\|\boldsymbol{\delta}_m\|^2$$

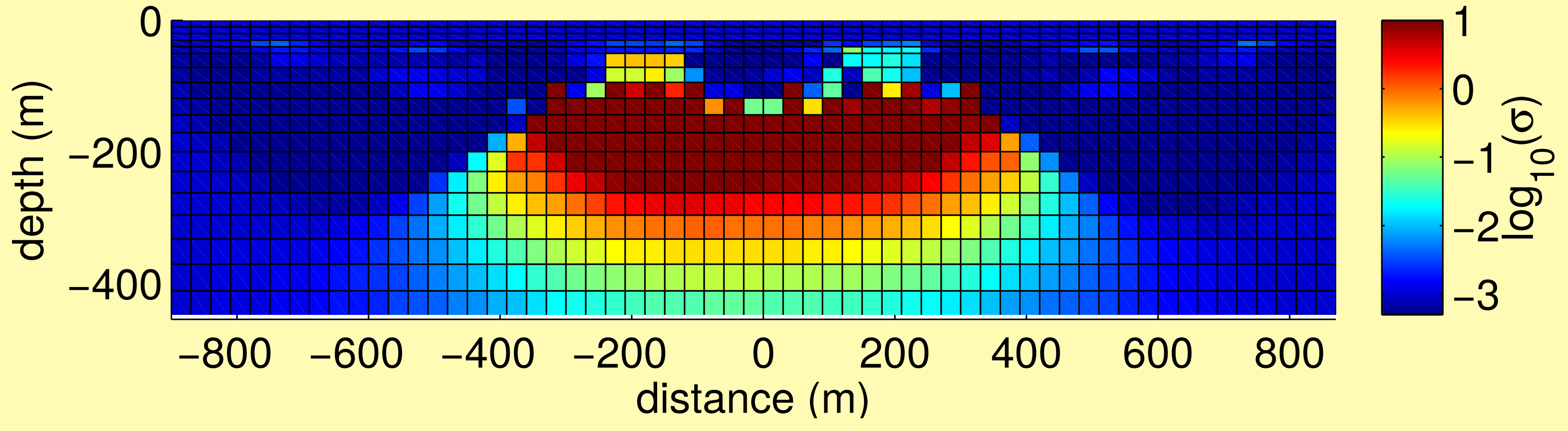
This leads to the system of equations

$$(\mathbf{J}^T\mathbf{J} + \nu^2\mathbf{I})\boldsymbol{\delta}_m = \mathbf{J}^T\boldsymbol{\varepsilon},$$

which we solve using the reference implementations of the LSQR iterative solver [2].

Conclusions





We have significantly improved ArjunAir run times through the use of parallel computing and more efficient sequential algorithms. It makes ArjunAir a more useful code but inversion results are still too dependent on initial guess. The use of minimum structure inversion should improve inversion results without sacrificing speed. A minimum structure inversion code using the ArjunAir forward solver has been mostly written.

References

[1] P. AMESTOY , I. DUFF, J. K OSTER, AND J.-Y. L'E XCELLENT, A fully asynchronous multifrontal solver using distributed dynamic scheduling, SIAM Journal of Matrix Analysis and Applications, 23 (2001), pp. 15–41.

[2] C. PAIGE AND M. SAUNDERS, LSQR: An algorithm for sparse linear equations and sparse least squares, ACM Transactions on Mathematical Software, 8 (1982), pp. 43–71.

[3] O. SCHENK AND K. GARTNER, Solving unsymmetric sparse systems of linear equations with PARDISO, Journal of Future Generation Computer Systems, 20 (2004), pp. 475–487.

[4] G. WILSON , A. RAICHE, AND F. SUGENG, 2.5D inversion of airborne electromagnetic data, Exploration Geophysics, 37 (2006), pp. 363–371.