Dual Reciprocity Method for studying thermal flows related to Magma Oceans

Tyler Drombosky
Ph.D. Student, Applied Mathematics Scientific Computation
Department of Mathematics
University of Maryland, College Park, MD
drombosky@math.umd.edu

Dr. Sawata Hier-Majumder
Assistant Professor, Department of Geology
University of Maryland, College Park, MD
sawata@umd.edu

Abstract

The tools to computationally model crystals settling in a magma ocean are currently not readily available. Being able to model such behavior may provide clues into the history of early Earth. New numerical simulations could lead to a better understanding of the settling. The equations that describe this settling are well understood, however, no suitable software package is currently available to solve such problems. This project will use the Dual Reciprocity Method, an extension of Boundary Element Method, to generate numerical solutions to a coupled system of Stokes flow and heat equations. The Dual Reciprocity Method allows for free boundary conditions as well as a reduction in problem dimensionality which decreases computation. The final product will combine algorithms from several papers to produce a robust, modular, and highly optimized software package.

1 Introduction

Earth's early history is marked by a giant impact with a Mars-sized object which led to the formation of the moon [1]. This impact event led to a substantial amount of melting of the Earth's interior. Subsequent cooling of the Earth involved extensive crystallization in this 'magma ocean' over a relatively short period of time. While the chemical evidence from ancient sources provide some clues on the rate of cooling, computational models of such phenomenon are sparse.

Constraints on the mode of crystal settling come from laboratory experiments [2], parameterized heat flux calculations [3][4][5][6], and chemical constraints [7]. The first two lines of evidence suggest extremely small crystals, contrary to common belief, also settle likely enhancing the efficiency of cooling. Direct estimation of heat flux and crystal size, however, are still not available. Chemical arguments suggest that the pattern of density-driven settling is also controlled by the vast pressures associated with a plant-scale magma ocean.

Modeling this physical behavior requires solving a coupled system of partial differential equations (PDEs), specifically the Stokes flow and heat equation. As closed forms solutions rarely exists for problems of interest, numerical solutions are an attractive option. Methods for solving various types of PDEs, while being robust, require PDEs to conform to certain forms and restrictions.

When solving PDEs involving coupled Stokes flow and heat transfer, it is standard practice to use a finite element method (FEM) solver. FEM algorithms are very robust and can generally converge with reasonable amounts of computation. Unfortunately there are two issues with FEM that make it unpractical for the above problem. First, as the crystals settle, the region of flow will be changing. This free boundary problem means the region would have to be rediscretized at every time step. This becomes extremely costly, especially at high dimensions. In addition to discretization issue, FEM is $\mathcal{O}(n^3)$ in $\mathbb{R}^3$. While this project will only focus on domains in $\mathbb{R}^2$, using a FEM algorithm would not scale well if adapted to work in $\mathbb{R}^3$.

Boundary Element Method (BEM) is an alternative technique for solving the same problem. The goal of BEM is to write the weak form of the PDEs employing reciprocal relations, integrating by parts, and making specific choices on test functions such that the integrals move entirely to the boundary. This has the immediate advantage of reducing the dimensionality by one. Also, BEM naturally handles free boundary problems. Unlike the domain meshes need for FEM, BEM discretizes the boundary. For any evaluation, only three easily computable pieces about the boundary are need.

First is the position of the node. Second, is the boundary value at the node, this can be Dirichlet or Neumann boundary condition or a prescribed condition such as the no-slip boundary condition for Stokes flow. Third is the vector normal to the boundary at the node. This can be quickly be approximated by using a local interpolation of nodes.

2 Approach

Dual Reciprocity Method (DRM) is an algorithm based off of BEM [8][9][10][11]. DRM, like BEM, provides an approximation on the boundary rather than in the region. This provides a reduction in dimensionality. This makes the problem of discretization of the region significantly easier. In the case of $\mathbb{R}^2$ regions, this means only having to discretize a curve. This reduction of dimensionality also allows for adding more mesh points with less computational penalty. DRM advantage over BEM is that it provides a way to move more complicated PDE to the boundary.

DRM works on PDE in the form

$$\mathcal{D}u = b, \tag{1}$$

where $\mathcal{D}$ is a generic non-linear operator with variable coefficients. A linear differential operator $\mathcal{L}$ is chosen such that the fundamental solution to the adjoint operator $\mathcal{L}^*$ is known. Then $\mathcal{D}$ can be written as a sum of the linear operator $\mathcal{L}$ and a residual operator $\mathcal{D}'$. Then (1) can be written

$$\mathcal{L}u = b - \mathcal{D}'u =: b'. \tag{2}$$

The combination of the original source term and residual operator are now treated as a new source term for the linear PDE. Considering the example of the non-homogeneous heat equation

$$\frac{\partial}{\partial t}u + \vec{v} \cdot \nabla u - \Delta u = b \tag{3}$$

where $\vec{v}(x)$ is the velocity in Ω computed by a stokes solver. The operator can then be broken up into

$$\mathcal{D} = \frac{\partial}{\partial t} + \vec{v} \cdot \nabla - \Delta \tag{4}$$

$$\mathcal{L} = -\Delta \tag{5}$$

$$\mathcal{D}' = \frac{\partial}{\partial t} + \vec{v} \cdot \nabla \tag{6}$$

$$b' = b - \frac{\partial}{\partial t} u - \vec{v} \cdot \nabla u \tag{7}$$

The boundary integral equation is then formulated for (3). This is done in the same manner as in BEM. Like traditional FEM, BEM multiplies a PDE by a test function and converts it into an integral equation. The formulation for the (3) is given

$$-\Delta u = b' \qquad x \in \Omega \tag{8}$$

$$-\Delta u \cdot w = b' \cdot w \tag{9}$$

$$-\int_\Omega \Delta u \cdot w \, \mathrm{d}\Omega = \int_\Omega b' \cdot w \, \mathrm{d}\Omega \tag{10}$$

where Ω is an open set. Then integration by parts is applied twice on the left hand side

$$\int_\Omega \Delta u \cdot w \, \mathrm{d}\Omega = \int_\Gamma w Du \cdot \vec{n} \, \mathrm{d}\Gamma - \int_\Omega Du \cdot Dw \, \mathrm{d}\Omega \tag{11}$$

$$= \int_\Gamma w Du \cdot \vec{n} \, \mathrm{d}\Gamma - \int_\Gamma u Dw \cdot \vec{n} \, \mathrm{d}\Gamma + \int_\Omega u \Delta w \, \mathrm{d}\Omega \tag{12}$$

where $\Gamma = \bar{\Omega} \backslash \Omega$ is the boundary. BEM makes the specific choice of $w = u^*$ where u^* is the fundamental solution of the dual of the Laplacian. That is u^* satisfies

$$\Delta u^* = -\delta(x - \xi) \tag{13}$$

for a choice of $\xi \in \Omega$. Then plugging (12) and (13) into (10)

$$u(\xi) = \int_\Gamma (u^* Du - u Du^*) \cdot \vec{n} \, \mathrm{d}\Gamma + \int_\Omega b' u^* \, \mathrm{d}\Omega. \tag{14}$$

To this point, the method has not differed from BEM and in fact (14) could be solved numerically if the region was discretized for integration. However, the goal of boundary methods is to avoid discretizing the region. DRM provides a way to approximate the integral over the region as an integral over the boundary. In DRM the source term $b' = b - \mathcal{D}'u$ is approximated by

$$b' = \sum_{q=1}^{\mathcal{N}} f^q \alpha_q \tag{15}$$

where f^q are chosen basis functions and α_q are coefficients. The basis functions f^q are chosen appropriately so u^q the solution to

$$\Delta u^q = f^q, \tag{16}$$

can easily be found. This eliminates the need for finding a particular solution for b'. Instead it is approximated by a linear combination of particular solutions for f^q which can easily be found. Using this approximation (8) becomes

$$-\Delta u = \sum_{q=1}^{\mathcal{N}} f^q \alpha_q \tag{17}$$

and (14) becomes

$$\begin{aligned} u(\xi) &= \int_\Gamma (u^* Du - uDu^*) \cdot \vec{n} \, \mathrm{d}\Gamma + \sum_{q=1}^{\mathcal{N}} \alpha_q \int_\Omega f^q u^* \mathrm{d}\Omega & (18) \\ &= \int_\Gamma (u^* Du - uDu^*) \cdot \vec{n} \, \mathrm{d}\Gamma - \sum_{q=1}^{\mathcal{N}} \alpha_q \int_\Omega \Delta u^q u^* \mathrm{d}\Omega & (19) \end{aligned}$$

employing Green's second identity yields

$$\int_\Omega (\Delta u^q u^* - \Delta u^* u^q) \, \mathrm{d}\Omega = \int_\Gamma (Du^q u^* - Du^* u^q) \cdot \vec{n} \, \mathrm{d}\Gamma \tag{20}$$

because of (13) this relationship becomes

$$\int_\Omega \Delta u^q u^* \, \mathrm{d}\Omega = -u^q(\xi) + \int_\Gamma (Du^q u^* - Du^* u^q) \cdot \vec{n} \, \mathrm{d}\Gamma \tag{21}$$

Substituting (21) into (14) yields the Dual Reciprocity equation

$$u(\xi) = \int_\Gamma (u^* Du - uDu^*) \cdot \vec{n} \, \mathrm{d}\Gamma + \sum_{q=1}^{\mathcal{N}} \left(u^q(\xi) - \int_\Gamma (Du^q u^* - Du^* u^q) \cdot \vec{n} \, \mathrm{d}\Gamma \right) \alpha^q \tag{22}$$

It is given this name because two reciprocity equations are used, once to convert the integral with the linear term to the boundary and then again to convert the approximation of the source term to the boundary. The advantage of DRM is that all integrals now are on the boundary.

This same procedure can be applied to the Stokes flow equation to obtain the boundary integral equation

$$\vec{v}(\xi) = \int_\Gamma \Delta f \cdot \underline{J} \, \mathrm{d}\Gamma + \frac{1-\lambda}{1+\lambda} \int_\Gamma \vec{n} \cdot \mathbf{K} \cdot v \, \mathrm{d}\Gamma \tag{23}$$

where f represents the surface tension and buoyancy, λ is the ration of viscosity between the particle and fluid, and $\underline{J}$ a second order tensor and $\mathbf{K}$ a third order tensor are the fundamental solutions for the adjoint operator. Since u depends on $\vec{v}$ but $\vec{v}$ does not depend on u, a solution for $\vec{v}$ is approximated first using (23) then this approximation is used in computing u using (22).

Once the boundary integral equations are derived, variables non-dimensionalized and the integrals are discretized. The discretization will be done using the method of collocation. In this method, the integral is broken up into N disjoint, connected segments $\Gamma_1, \ldots, \Gamma_n$ such that $\Gamma_i \cap \Gamma_j = \emptyset$ for $i \neq j$ and $\cup_{i=1}^{N} \Gamma_i = \Gamma$. Next, $x_i \in \Gamma_i$ are selected and used as the ξ values. This yields a system of N equations and N unknowns that can be solved.

Moving ξ to the boundary presents an issue. As stated earlier $\xi \in \Omega$, however to use the method of collocation, ξ must be moved to the boundary. This can be done in a limiting process. This results in both weakly and strongly singular integrals. When integrating on Γ_i with respect to the collocation point $x_j \notin \Gamma_i$, the integral is non-singular and is approximated using an eight point Gauss quadrature method. If $x_i \in \Gamma_i$, the integral is singular and it can be solved using radial approximation methods [12][13].

3 Implementation

The software package will be developed in Fortran 90 with compiler level optimizations focused on single threaded x86 processors. Fortran 90 was chosen as the programming language for several reasons. First, Fortran 90 is an object orientated language. Types and modules allow for use of custom data types and methods. This feature is crucial for constructing modular software. Second, there is a large library of Fortran 90 code developed. These packages, both widely distributed and local, are highly modular and will cut down considerably on development time. Finally, Fortran 90 has significant speed advantages, even when compared to other low level languages such as C.

The target platform for the software is a single threaded x86 system. The specific test system is a 4 node cluster. Each cluster houses two quad core processor with each processor sharing 8 gigabytes of main memory. While an instance of this software will not use more than one of the 32 cores, this system will be convenient if a need arrises to run several time consuming simultaneous at once. However, as developed, the software will be able to compile and run efficiently in any x86 environment.

The software will be created in several steps. First, existing code for solving the poisson and Stokes equations in steady-state using BEM will be merged. The softwares were written at different times for different purposes. As is expected in this case, there are several software engineering challenges that must be solved involving unifying data structures. In addition, both solvers use similar libraries that are not always identical. While it would be possible to include both copies of these libraries, this approach is sloppy and does not align with the goal of making the software highly modular. Thus, these libraries will be standardized and merged as necessary.

After the poisson and Stokes code are merged, work will begin on developing DRM code. The only difference between DRM and BEM is the transformation of the source term to the boundary. The code for DRM will accordingly be highly modular. Once written it can be added to the existing merged code with little effort.

As mentioned earlier, DRM requires computing integrals with strong and weak singularities. While these singularities can be computationally difficult, there are many documented methods for obtaining accurate approximations to such problematic integrals. However, in practice testing is needed to determine which ones are actually successful for approximating the specific integral.

Unlike FEM, the linear systems produced by DRM and BEM are not sparse. In fact, the matrix generated by DRM is, in general, dense and asymmetric. Because of this, solving the linear system will be done using a straightforward LU decomposition with pivoting for stability through the forward and backward substitution.

A solution to overcome the dense matrix is to employ the fast multipole method (FMM) [14]. One of the reasons that DRM has a dense matrix is because each node requires unique information from every other node. This is an $\mathcal{O}(N^2)$ process which creates the dense and asymmetric matrix. FMM creates poles throughout the region and then "connects" the nodes to the poles and then poles to poles in a tree structure. Data is then shared between nodes via the poles. Doing this requires $\mathcal{O}(N)$ to set up the tree. This creates a sparse matrix with $\mathcal{O}(N \log N)$ entries. With this sparsity the linear system can then be solved with more efficient and robust methods such as GMRES. If time permits a FMM module will be added to the software package.

The software will run the Stokes solver first followed by either a poisson or heat equation solver depending on the development stage. Each solver will run in the same order of time. As presented the algorithm will run in $\mathcal{O}(N^2)$ time where N is the number of discrete nodes on the boundary. This dominated by the LU decompositions and computation of the entries in the mass matrix. Evaluating each of the $\mathcal{O}(N^2)$

integrals takes constant time. When the integral is non-singular, eight point Gauss quadrature is used. For the approximation of the singular integrals, an iterative method is used, but convergence can be expected in $i \in o(N)$ iterations. This can therefore be thought of as a constant time operation with respect to N.

Using FMM, the runtime of the entire software runtime reduces to $\mathcal{O}(N \log N)$ as setting up FMM takes one time work of $\mathcal{O}(N)$ and $\mathcal{O}(N \log N)$ is required for computing the entries in the mass matrix. Solving the linear system can then be done with sparse methods that converge in $\mathcal{O}(N)$ time.

4 Validation and Testing

The software be validated and tested at two points in the development cycle. The first time will be after the poisson and Stokes solvers have been merged to create a steady-state Stokes flow and heat solver. As the name of the solver implies, it will be able to solve steady state equations for temperature gradients in a flow. The solver will first be tested against problems where there is no heat source and then in problems where there is no flow.

For an isoviscous region, the steady state-solver degrades to a poisson solver. There exists analytic solutions to the poisson equation on a circle which can be used to validate this part of the code. Testing of the Stokes solver will follow a similar testing method. With an isothermal region the solver becomes a Stokes solver. For constant surface tension, analytical solutions to the Stokes equations can be found. With a heat source and flow, asymptotic approximations using matching asymptotes provide an approximation of steady state thermal flow for low Péclet numbers [15]. This approximation can be used to gauge first order accuracy of the numerical solutions.

Once the DRM code is implemented a similar round of validation and testing will occur. A heat equation problem and transient stokes flow problem will be approximated. In both cases, analytic solutions and numerical solutions are available from other sources. Once these two elements of the DRM have been tested, a transient problem with both flow and a source term will be approximated. At this point no analytic or asymptotic solution exists. For validation a function $\tilde{u}$ will be created randomly with appropriate initial and boundary conditions. Then $\tilde{b} := \mathcal{L}\tilde{u}$ is computed. Finally, the solver is used to approximate to solution to $\mathcal{L}u = \tilde{b}$ with the chosen initial and boundary conditions. The approximation of u can then be compared to the exact solution $\tilde{u}$. In this way the convergence and compute scaling can also be tested.

5 Project Schedule

- Phase I (present - early November):
  - Organize poisson and Stokes solvers
  - Analyze incompatibilities in data structures and libraries
  - Devise solution to incompatibilities, recoding as necessary
  - Merge solvers into one software
- Phase II (November - December):
  - Validate and test steady state solver
  - Optimize and fix bugs as necessary
  - Run simulations and archive results
- Phase III (December - early February):
  - Incorporate DRM code
  - Incorporate FMM (optional)
- Phase IV (February-March):
  - Validated and test DRM solver
  - Optimize and fix bugs as necessary
  - Run simulations and archive results
  - Prepare final report and presentation

6 Deliverables

This project will yield a highly modular and optimized source code and software for solving transient thermal Stokes flow problems with source terms in $\mathbb{R}^2$ using DRM. There is also the possibility of having an incorporated FMM solver in the final code base. Great care will be made to ensure the code is well organized and easily modifiable for future projects.

A report and presentation that cover the algorithm will be presented to the class. Furthermore, simulations of physical phenomena will be recorded and written in a

paper to be submitted for publication. Those results will be presented in Berlin, Germany at 12^{th} International Workshop on Modeling of Mantle Convection and Lithospheric Dynamics in Summer 2011.

References

[1] R. M. Canup and E. Asphaug, "Origin of the Moon in a giant impact near the end of the Earths formation", *Nature*, Volume 412, 2001, Pages 708-712

[2] D. Martin and R. Nokes. "A Fluid-Dynamical Study of Crystal Settling in Convecting Magmas", *Journal of Petrology*, Volume 30, Issue 6, 1989, Pages 1471-1500

[3] V. Solomatov and D. Stevenson, "Suspension in Convective Layers and Style of Differentiation of a Terrestrial Magma Ocean", *Journal of Geophysical Research*, Volume 98, Issue E3, 1993, Pages 5375-5390

[4] V. Solomatov and D. Stevenson, "Nonfractional Crystallization of a Terrestrial Magma Ocean", *Journal of Geophysical Research*, Volume 98, Issue E3, 1993, Pages 5391-5406

[5] V. Solomatov and D. Stevenson, "Kinetics of Crystal Growth in Terrestrial Magma Ocean", *Journal of Geophysical Research*, Volume 98, Issue E3, 1993, Pages 5407-5418

[6] V. Solomatov, "Magma Oceans and Primordial Mantle Differentiation", *Treatise on Geophysics*, Volume 9, 2007, Pages 91-120

[7] L. T. Elkins-Tanton, E. M. Parmentier, and P. C. Hess, "Magma ocean fractional crystallization and cumulate overturn in terrestrial planets: Implications for Mars", *Meteoritics & Planetary Science*, Volume 38, Issue 12, 2003, Pages 1753-1771

[8] D. Nardini and C. A. Brebbia, "A new approach to free vibration analysis using boundary elements", *Boundary Element Methods in Engineering*, Volume 7, Issue 3, 1983, Pages157-162

[9] D. Nardini and C. A. Brebbia, "Transient dynamic analysis by the boundary element method", *Boundary Elements*, 1983, Pages 719-730

[10] D. Nardini and C. A. Brebbia, "Boundary integral formulation of mass matrices for dunamic analysis", *Topics in Boundary Element Research*, Volume 2: Time-Dependent and Vibration Problems, 1985, Pages 191-208

[11] L. Gaul, M. Kögl, M. Wagner, *Boundary Element Methods for Engineers and Scientists*, Springer, Berlin, 2003

[12] C. Pozrikidis, *Boundary integral and singularity methods for linearized viscous flow*, Cambridge University Text, New York, NY, 1992

[13] Xiao-Wei Gao, "Numerical evaluation of two-dimensional singular boundary integrals-Theory and Fortran code", *Journal of Computational and Applied Mathematics*, Volume 188, Issue 1, 2006, Pages 44-64

[14] Y. J. Liu and N. Nishimura, "The fast multipole boundary element method for potential problems: A tutorial", *Engineering Analysis with Boundary Elements*, Volume 30, Issue 5, May 2006, Pages 371-381

[15] G. Leal, *Laminar Flow and Convective Transport Processes*, Butterworth-Heinemann, 1992