Thesis Proposal Draft

O. Tate Kernell

1 Introduction

Tide modeling is an important component in many areas of scientific research. From coastal flooding and sediment transport to ocean circulation, the accurate modeling of tides has widespread value in the scientific community. Unstructured triangular meshes appear to be useful in modeling the ocean with finite element methods. In this thesis, we will focus primarily on the linearized rotating shallow-water equations with damping, which are used for predicting global barotropic tides. Our goal is to provide a good preconditioner for the linearized rotating shallow-water equation. However, in order to do this, we will start from the basic acoustic wave equation.

$$\begin{aligned} qu_t + \nabla p &= 0, \\ k^{-1}p_t + \nabla \cdot u &= 0, \end{aligned} \tag{1}$$

on some domain $\Omega \times [0,T] \subset \mathbb{R}^d \times \mathbb{R}$ with $d = 2, 3$, with the assumption that Ω is polyhedral. The parameter q, the material density, is a measurable function bounded above and below by q_* and q^*. The parameter k is the bulk modulus of compressibility, which we assume is bounded by positive k_* and k^*. For now, we will assume $q = 1 = k$. Additionally, we impose the boundary condition $u \cdot \nu = 0$ on $\partial\Omega$ where ν is the unit outward to Ω. We choose initial conditions

$$\begin{aligned} p(x,0) &= p_0(x) \text{ and} \\ u(x,0) &= u_0(x). \end{aligned} \tag{2}$$

Converting this system into weak form, we get

$$\begin{aligned} (u_t, v) + (\nabla p, v) &= (f, v), \quad v \in H_0(\text{div}) \\ (p_t, w) + (\nabla \cdot u, w) &= (g, w), \quad w \in L_0^2 \end{aligned} \tag{3}$$

where $u : [0,T] \to V \equiv H_0(\text{div})$ and $p : [0,T] \to W \equiv L_0^2$, along with the initial conditions (2). When we integrate by parts (3) changes to

$$\begin{aligned} (u_t, v) - (p, \nabla \cdot v) + \underbrace{\langle p, v \cdot \nu \rangle_{\partial\Omega}}_{=0} &= (f, v) \\ (p_t, w) + (\nabla \cdot u, w) &= (g, w), \end{aligned} \tag{4}$$

which leads to our final form

$$\begin{aligned} (u_t, v) - (p, \nabla \cdot v) &= (f, v) \\ (p_t, w) + (\nabla \cdot u, w) &= (g, w) \end{aligned} \tag{5}$$

Remark 1.1. *It is important to note that* $H(\text{div}) := \{v \in L^2,\ \text{div}(v) \in L^2\}$ *and* $H_0(\text{div}) := \{v \in H(\text{div}),\ v \cdot \nu|_{\partial\Omega} = 0\}$ *where* ν *is the outward normal.*

The semidiscrete mixed formulation of (5) is to find $u_h : [0,T] \to V_h$ and $p_h : [0,T] \to W_h$ such that

$$\begin{aligned} (u_{h,t}, v_h) - (p_h, \nabla \cdot v_h) &= (f, v_h) \\ (p_{h,t}, w_h) + (\nabla \cdot u_h, w_h) &= (g, w_h) \end{aligned} \tag{6}$$

for all $v_h \in V_h$ and $w_h \in W_h$ where $V_h \subset V$ and $W_h \subset W$. By discretizing (6) with Crank Nicolson and partitioning the time interval $[0,T]$ into timesteps $0 \equiv t_0 8 < t_1 < t_2 < ... < t_N$, where $t_i = i\Delta t$, we arrive at an approximation of the solution to the semidiscrete mixed formulation. We chose Crank Nicolson primarily because it is exactly energy conserving, but also it provides the benefit of being numerically stable and having improved convergence over explicit methods. Here, $u_h(t_n) \approx u_h^n \in V_h$ and $p_h(t_n) \approx p_h^n \in W_h$

$$\begin{aligned} \left(\frac{u_h^{n+1} - u_h^n}{\Delta t}, v_h\right) - \left(\frac{p_h^{n+1} + p_h^n}{2}, \nabla \cdot v_h\right) &= \left(f^{n+\frac{1}{2}}, v_h\right) \\ \left(\frac{p_h^{n+1} - p_h^n}{\Delta t}, w_h\right) + \left(\nabla \cdot \frac{u_h^{n+1} + u_h^n}{2}, w_h\right) &= \left(g^{n+\frac{1}{2}}, w_h\right) \end{aligned} \tag{7}$$

where $f^{n+\frac{1}{2}} = \frac{f(t_{n+1}) + f(t_n)}{2}$ and likewise for g. Letting $f = 0$ and $g = 0$ and multiplying by Δt, we get

$$\begin{aligned} \left(u_h^{n+1} - u_h^n, v_h\right) - \left(\frac{\Delta t}{2}\left(p_h^{n+1} + p_h^n\right), \nabla \cdot v_h\right) &= 0 \\ \left(p_h^{n+1} - p_h^n, w_h\right) + \left(\frac{\Delta t}{2} \nabla \cdot \left(u_h^{n+1} + u_h^n\right), w_h\right) &= 0 \end{aligned} \tag{8}$$

Reshuffling terms leads to

$$\begin{aligned} \left(u_h^{n+1}, v_h\right) - \frac{\Delta t}{2}\left(p_h^{n+1}, \nabla \cdot v_h\right) &= \left(u_h^n, v_h\right) + \frac{\Delta t}{2}\left(p_h^n, \nabla \cdot v_h\right) \\ \left(p_h^{n+1}, w_h\right) + \frac{\Delta t}{2}\left(\nabla \cdot u_h^{n+1}, w_h\right) &= \left(p_h^n, w_h\right) - \frac{\Delta t}{2}\left(\nabla \cdot u_h^n, w_h\right) \end{aligned} \tag{9}$$

Let $\{\phi_i\}_{i=1}^{|W_h|}$ and $\{\psi_i\}_{i=1}^{|V_h|}$ be bases for W_h and V_h respectively. Then we can define mass matrices

$$\begin{aligned} M_{ij} &= (\phi_j, \phi_i), \\ \tilde{M}_{ij} &= (\psi_j, \psi_i). \end{aligned} \tag{10}$$

We can formulate (6) as

$$\begin{bmatrix} \tilde{M} & 0 \\ 0 & M \end{bmatrix} \begin{bmatrix} u_t \\ p_t \end{bmatrix} + \begin{bmatrix} 0 & -D^T \\ D & 0 \end{bmatrix} \begin{bmatrix} u \\ p \end{bmatrix} = \begin{bmatrix} F \\ G \end{bmatrix}, \tag{11}$$

where D is the differential operator div, and F and G are the vectors (f, v_h) and (g, w_h) respectively. Additionally, we can put (9) into matrix form, giving us

$$\begin{bmatrix} \tilde{M} & -\frac{\Delta t}{2} D^T \\ \frac{\Delta t}{2} D & M \end{bmatrix} \begin{bmatrix} u_h^{n+1} \\ p_h^{n+1} \end{bmatrix} = \begin{bmatrix} \tilde{F} \\ \tilde{G} \end{bmatrix} \tag{12}$$

where $\tilde{F} = -\left(u_h^n, v_h\right) + \frac{\Delta t}{2}\left(p_h^n, \nabla \cdot v_h\right)$ and $\tilde{G} = -\left(p_h^n, w_h\right) - \frac{\Delta t}{2}\left(\nabla \cdot u_h^n, w_h\right)$, which gives our fully discretized system.

2 Preconditioning

Returning to the PDE (1) with coefficients equal to 1, if we apply Crank Nicolson in the time derivative without discretizing in space, (1) becomes

$$\begin{aligned} \frac{u^{n+1}-u^n}{\Delta t} + \nabla\left(\frac{1}{2}\left(p^{n+1}+p^n\right)\right) = 0 \\ \frac{p^{n+1}-p^n}{\Delta t} + \nabla\cdot\left(\frac{1}{2}\left(u^{n+1}+u^n\right)\right) = 0 \end{aligned} \tag{13}$$

which leads to

$$\begin{aligned} u^{n+1} + \frac{\Delta t}{2}\nabla p^{n+1} = u^n - \frac{\Delta t}{2}\nabla p^n \\ p^{n+1} + \frac{\Delta t}{2}\nabla\cdot u^{n+1} = p^n - \frac{\Delta t}{2}\nabla\cdot u^n. \end{aligned} \tag{14}$$

Therefore, at each timestep we have a discretization of the coefficient operator $\mathscr{A}$, described as

$$\mathscr{A} = \begin{pmatrix} I & \epsilon\,\text{grad} \\ \epsilon\,\text{div} & I \end{pmatrix} \tag{15}$$

where $\epsilon = \frac{\Delta t}{2}$. By discretizing in the finite element space $H_0(\text{div}) \times L_0^2$, as seen in (12), we recover our finite dimensional coefficient operator, $\mathscr{A}_h$, defined as

$$\mathscr{A}_h = \begin{bmatrix} \tilde{M} & -\frac{\Delta t}{2}D^T \\ \frac{\Delta t}{2}D & M \end{bmatrix} \tag{16}$$

We claim $\mathscr{A}$ is an isomorphism mapping $H(\text{div})\times L^2$ onto $H(\text{div})^*\times L^2$, its dual space. Additionally, it is bounded on $H(\text{div}) \times L^2$ with bounded inverse. In the view of [] a common approach to preconditioning is to create an equivalent operator that is easier to invert. Equivalent in this sense means that $\mathscr{B}^{-1}\mathscr{A}$ is a nice operator from the initial space into itself rather than into its dual, where $\mathscr{B}$ is the preconditioner. If $\mathscr{B}^{-1}\mathscr{A}$ is bounded in the Hilbert space, we get mesh independent eigenvalue clustering []. Our goal is to find a preconditioner $\mathscr{B}$ which maps $H(\text{div})^* \times L^2$ onto $H(\text{div}) \times L^2$. This preconditioner will be explored below. However, we can also formulate this problem in an alternative way to be on the space $L^2 \times H^1$. This method is based on the Schur complement. Instead, we can implement hybridization to provide our solution. We will compare this to our explored method in a later section.

From methods described in Mardal/Winther, we want our preconditioner to be a block diagonal operator suggested by the mapping properties of the coefficient operator of the system. The canonical preconditioner for our specific coefficient operator $\mathscr{A}$ is derived from the spaces it maps and is as follows:

$$\mathscr{B} = \begin{bmatrix} \beta\, I - \alpha\,\text{grad}\,\text{div} & 0 \\ 0 & \gamma\, I \end{bmatrix} \tag{17}$$

Here, if $\alpha = \beta = \gamma = 1$, $\mathscr{B}$ is the canonical Riesz map preconditioner, that maps the dual space back to our original space. (Need to talk about stable finite element discretizations and stability following from inf/sup condition). Similarly, the discrete preconditioner is of the form

$$\mathscr{B}_h = \begin{bmatrix} \alpha(\nabla\cdot, \nabla\cdot) + \beta(\cdot,\cdot) & 0 \\ 0 & \gamma(\cdot,\cdot) \end{bmatrix} \tag{18}$$

For this preconditioner, our goal is to look at its eigenvalues as a function of α, β, and γ. We want to choose these parameters so that our preconditioner is robust in terms of Δt, thus eliminating the dependence on ϵ. First of all, we plan on proving a theorem about the boundedness of this preconditioner and its inverse. Next, we hope to show how the condition number varying with respect to Δt, α, β, and γ. Lastly, we will prove a theorem showing that with the correct choice of α, β, and γ, we can control the condition number so it is of order 1. Note that our coefficient operator $\mathscr{A}$ is a bounded map with bounded inverse from $H(\mathrm{div}) \times L^2$ into its dual. We can then premultiply with $\mathscr{B}$, the Riesz map, thus giving that $\mathscr{B}^{-1}\mathscr{A}$ is a bounded operator. Additionally, this operator will be mesh independent. We will use this on the coefficient operator of the system (**??**) and observe how well it performs. If the preconditioner is easily invertible (using AMS or others), then we have found a good preconditioner for this system.

3 Shallow Water Equations

Our main goal is to create a preconditioner for the for the linearized rotating shallow-water equations with nonlinear damping. Our plan is to incrementally build up the acoustic wave equation by adding appropriate terms (damping, coriolis, etc.) and adapting the canonical preconditioner along the way. The shallow-water equations are as follows

$$\begin{aligned} u_t + \frac{f}{\epsilon}u^{\perp} + \frac{\beta}{\epsilon^2}\nabla(\eta - \eta') + g(u) &= F \\ \eta_t + \nabla \cdot (Hu) &= 0, \end{aligned} \tag{19}$$

where u is the nondimensional two dimensional velocity field tangent to Ω, $u^{\perp} = (-u_2, u_1)$ is the velocity rotated by $\pi/2$, η is the nondimensional free surface elevation above the height at state of rest, $\nabla\eta'$ is the (spatially varying) tidal forcing, ϵ is the Rossby number (which is small for global tides), f is the spatially-dependent non-dimensional Coriolis parameter which is equal to the sine of the latitude (or which can be approximated by a linear or constant profile for local area models), β is the Burger number (which is also small), H is the (spatially varying) nondimensional fluid depth at rest, and ∇ and $\nabla\cdot$ are the intrinsic gradient and divergence operators on Ω, respectively. write out the reforming of the equations of the mixed form and look at what the linear algebra will look like for those systems of equations and what we want to prove about them If the preconditioner is not easily invertible, however, we will attempt to find a more computationally suitable, spectrally equivalent preconditioner.

Of course, in order to quantify the effectiveness of our preconditioner, we plan on comparing it to other known methods. This work will be done in Firedrake. Some methods we intend to use are the Schur Complement (with Slate), AMS, and ADS.

Lastly we will ask important questions, such as: What are the bounds on the eigenvalues of the preconditioner? Do they behave well with respect to the parameters? Do we have a mesh independent eigenvalue bound? In regards to implementation, we are developing $H(\mathrm{div})$ preconditioners in Firedrake. Both our code development and numerical results will be using Firedrake and PETSc.

4 Theorems

5 Preliminary Results

Once the appropriate Riesz map preconditioner was determined for the coefficient operator in (**??**), we were able to write script in Firedrake to test some preliminary results. Using Hypre AMS, we implemented the canonical preconditioner. We compare the basic Riesz map versus hybridzation with a fixed ratio between the time step and mesh size.

Table 1: Riesz Map with HypreAMS

# Cells	**Iterations**
2	2
4	6
8	34
16	45
32	40
64	16
128	8
256	4

Table 2: Preconditioning with Hybridization using SLATE

# Cells	**Iterations**
2	1
4	3
8	3
16	4
32	4
64	4
128	3
256	3