Building Toy Dams: Modeling Stream Flow Around Obstacles in a Porous Media

Julian Schmitt
Applied Mathematics
Harvard University
jschmitt@college.harvard.edu

Chelsea Chen
Applied Mathematics
Harvard University
chelseachen@college.harvard.edu

December 18th, 2021

1 Introduction

1.1 Motivation

With the advent of climate change, humanity faces increased risks from natural disasters including heat waves, more severe storms, and sea level rise. While places like New England are relatively sheltered from many of the most extreme events like hurricanes and crippling drought, they are modeled to see increases in the intensity of flooding events. Both heavy rains and rapid spring snow melt are expected to increase in the coming decades. For example, mean rainfall across Massachusetts is expected to increase by over 2 inches/year under the current emission trajectory in the next 50 years alone [1]. That's over 350 billion additional gallons of water which must find its way through streams and rivers back out to the ocean! 100-year flood events have almost doubled in frequency to once every 60-years [1] with conditions only predicted to be further exacerbated in the coming decades. During flood events, a high volume of rain is deposited in an extremely short period of time, which rushes into streams and rivers. If flow rates exceed river capacity, the river spills over into surrounding areas and can wreak havoc on roads, bridges, and basements, posing a significant threat to infrastructure and human safety.

As New England looks ahead to prepare for the increased frequency of these events, several methods to reduce the damage caused by these extreme events. One potential method is relocation - people would be asked to move away from high-risk areas and new roads would be constructed out of harm's way. Current structures could also be strengthened - bridges and roads could be fortified or raised to reduce washout. Another solution, the one we will focus on in this paper, is to build embankments or dams in the hopes of redirecting water away from structures or better containing flow to the riverbed. We look to answer the research question if well-placed dams can be sited to prevent water flow and new channels from forming through vulnerable areas. We model this behavior using a 2-D porous medium where we enforce a steady flow of water into and out of a region which is given some initial random smoothed topography and then fluid is forced to move based on pressure differentials in the media. To simulate our toy dams, we modify the initial topography by increasing the solid fraction of the material we're modeling to simulate a barrier. We then run the simulation forward and observe what happens to the region we were trying to protect.

1.2 Literature Review

The seminal paper for this project is *Flow-Driven Branching in a Frangible Porous Medium* by Derr et al. (2020), which introduces one of the first models of flow through branched networks based on continuum theory rather than nonlinear models [5]. This paper proposes a mathematical model built off of symmetry arguments and an original multiphase model introduced in *Flow-induced channelization in a porous medium* by Mahadevan et al. (2012) [10]. One primary difference between the model introduced in Mahadevan's 2012 paper and Derr's 2020 paper is that Mahadevan's model accounted for deposition of dislodged grains onto the microstructure through a three-phrase description while Derr's model focuses on a two-phase model that assumes loose grains are indistinguishable from fluid. In Derr's model, a domain with a rigid grain microstructure is defined along with macroscopic continuum fields including solid fraction $\phi(x,t)$ and volumetric fluid flux $q(x,t)$. Pressure gradients and resistance are balanced so that individual grains feel forces and thus are dislodged, leading to an evolving permeability with an erosion rate. Thus, flow, permeability, and pressure gradients are linked to show emerging branching morphologies and flow enhancement.

Going back in literature, we see that a few papers emerge as landmark proposals. In *Stability and the conservation of mass in drainage basin evolution* (1972), Smith and Bretherton proposed one of the earliest continuum models for overland flow, describing erosion flow and showing that disturbances at the microscopic scale will be most unstable [13]. Following this observation, Izumi and Parker explored a threshold condition for erosion and its inclusion/exclusion in the model in two papers: *Inception of channelization and drainage basin formation: upstream-driven theory* (1995) and *Linear stability analysis of channel inception: downstream-driven theory* (2000) [8] [9]. This model of water erosion and channelization for permeable ground has been experimented with and theorized over the years, one example being *Spontaneous Channelization in Permeable Ground: Theory, Experiment, and Observation* by Schorghofer et al. (2003) [11]. This paper connects observation with water-table experiments, demonstrating how an evenly distributed fluid flux across a region tends to aggregate together forming channels with regular patterns and fractal behavior. They find that these channels "compete" for fluid flux in a zero-sum game leading to several dominant channels. Most recently, Zareei et al. studied the dynamics of flow-networks in porous media using a pore-network model in *Temporal Evolution of Flow in Pore-Networks: From Homogenization to Instability* (2021). In this paper, the evolution of pores in networks are modeled to simulate erosion and flow resistance as well as clogging dynamics (deposition of material on the pore throats). Low-Reynold's fluid flow and Poiseuille's law are considered for a topological random network of nodes constructed using Delaunay triangulation. Models like these were necessary contextual supplements to our primary model inspiration from Derr et al.

1.3 Our Model in the Literature Space

Our paper fits into the literature as a natural extension of Derr et al. (2020) as we expand the idea of observing channel formation to actively editing the topography to selectively discourage channel formation in certain regions of the grid. One key difference in the implementation is that while Derr et al. assume the pressure updates instantaneously at each time step when fluid is added, we assume that fluid, and thus pressure updates with some velocity as it propagates through the media. We thought this would be an interesting implementation difference and is computationally simpler so that we can run finite difference schemes in Python without having to worry significantly about performance issues for even reasonably sized grids on the order of 200×200.

While current porous media simulations model everything from channel formation to deposition and delta formation across sloped topography, there does not, to our knowledge, exist a study which models explicit modification of the topography for the purpose of redirecting channel formation away from existing structures.

2 Methods

Our methods section is separated into two parts, a section devoted to the theoretical derivation of our modeled equations and a section describing the numerical implementation of many of these methods. We outline in detail the derivation of our model equations, and then explain how we implemented them across a grid of cells.

Our model equations are derived from several broad properties of fluids as outlined below:

1. **Fluid Mass Conservation:** We assume that fluid mass in general cannot be created or destroyed. We do not want forces in our model to result in fluid creation or destruction, nor do we want fluid mysteriously appearing along a boundary. We do assume that the fluid is somewhat compressible, one of the key differences between our implementation and that of Derr et al.

2. **Fluid Movement:** Fluid is allowed to move, or flux, between cells with a rate determined by the pressure differentials between neighboring cells as well as solid fraction, or amount of solid material in that cell. As pressure goes up we expect fluid to flux faster. As the solid fraction goes up, there is more to resist fluid flow and so our modeled fluid velocity, and therefore flux goes down.

3. **Erosion Properties of the Solid Fraction ϕ:** We assume that pressure on the grains in a cell gives rise to erosion. Higher pressure means that grains in a given cell are more likely to erode. We tweak this interpretation slightly and add a component to adjust the erosion rate based on how much solid fraction is left in the area. This is based on the idea that solid material supports itself, making it harder

to erode. We assume there is some critical fraction near which we shift rapidly from an easy-to-erode scheme to a hard-to-erode scheme.

2.1 Flow Boundary Conditions and Ramp Up

At the outset of the implementation, we enforce zero flux across the boundary of our grid except for our sources and sinks. This ensures that when resolving the pressure field, fluid will, in general, move from the source to the sink, producing erosion along the way. This is implemented numerically when calculating the flux in both the x and y directions; we simply set the boundary fluxes to be zero except where we explicitly set non-zero flux to encourage flow into and out of the field. The non-zero boundary flux sums exactly to 0, as otherwise there would be a continual buildup of pressure at each step due to fluid conservation.

In order to model realistic erosion, we assume that the flux into and out of the media ramps up over time. This is necessary as, if we simply assumed that the flux suddenly increased dramatically, the pressure would be very high at the boundary and erosion in our results would progress in an uninteresting pattern. See Figure 1 below.

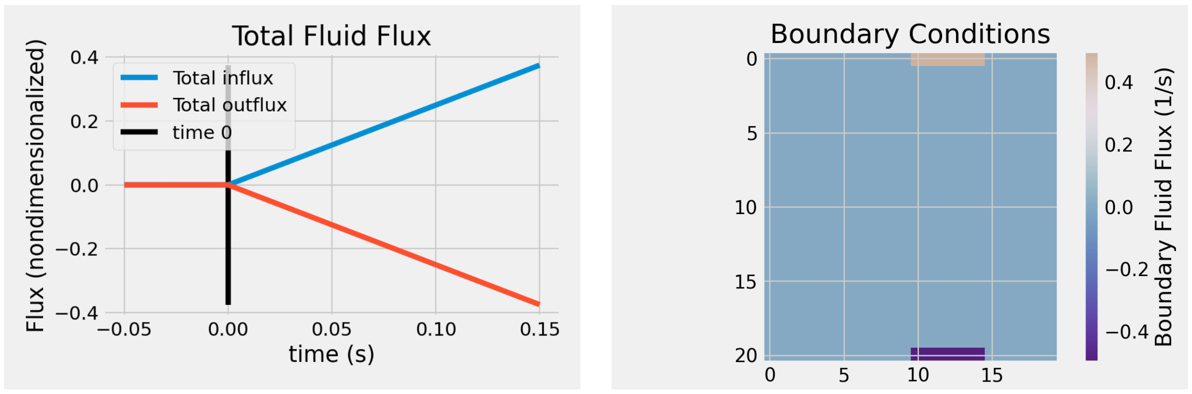

Figure 1: Fluid forcing in space and time. As an example of fluid flux, we see that initially we add no perturbations to the fluid field and then gradually ramp up fluid influx and fluid outflux with time. The right graphic depicts a possible boundary fluid flux in the y-direction. Note that this measures fluid flux and *not* the fluid density field.

2.2 Deriving Conservation of Mass

We will derive our conservation of mass equation by using a general equation for an intensive (mass independent) property of a substance. Given such a property L and its volume Q (with bounding area $\partial\Omega$), we use the following set of equations to derive conservation of mass. For this problem we assume the intensive property of interest is fluid density, ρ.

2.2.1 Reynold's Transport Theorem and Divergence Theorem

Reynold's Transport Theorem is stated as follows:

$$\frac{d}{dt}\int_{\Omega} LdV = -\left(\int_{\partial\Omega} L\vec{v}\cdot\vec{n}dA + \int_{\Omega} QdV\right). \tag{1}$$

The integral on the left tells us the rate of change of L inside volume Ω. The two integrals on the right model flux out of the region $\partial\Omega$ and the sink of the property within the region respectively. $\int_{\partial\Omega} L\vec{v}\cdot\vec{n}dA$ represents the volume amount of L governed by the speed at which the fluid moves normal to the boundary, hence the velocity vector $\vec{v}$ and normal vector$\vec{n}$. $\int_{\Omega} QdV$ represents the volume amount of L leaving through sinks/sources inside ∂Q. Physically this might correspond to a fluid evaporation rate or another sink such as a bathtub drain which might be localized.

With the divergence theorem we can rewrite the flux term $\int_{\partial\Omega} L\vec{v}\cdot\vec{n}dA$ as a volume-based integral as the flux described above is exactly a divergence. Thus we have that our total rate of change to L inside volume Ω is:

$$\frac{d}{dt}\int_{\Omega} LdV = -\int_{\Omega} \nabla\cdot(L\vec{v}+Q)dV. \tag{2}$$

We also make use of the Leibniz integral rule, or differentiation under the integral sign, to interchange the order of integration and differentiation on the left-hand side of the equation:

$$\frac{d}{dt}\int_a^b f(x,y,t)dt = \int_a^b \frac{d}{dt}f(x,y,t)dt. \tag{3}$$

2.2.2 Continuity

Applying Leibniz to the divergent form of the Reynolds Transport equation we can write the following:

$$\int_{\Omega}\frac{d}{dt}LdV = -\int_{\Omega} \nabla\cdot(L\vec{v}+Q)dV \tag{4}$$

We note that as we are integrating over the same region with respect to volume, assuming that this is true for all regions Ω.

$$\implies \frac{d}{dt}L = \nabla\cdot(L\vec{v}+Q) \tag{5}$$

2.2.3 Conservation of Mass

For our model we assume that fluid density, and therefor mass is conserved, and as density ρ is an intensive property we can substitute this into the equation derived above. In our implementation we will neglect Q and assume that evaporation or other sinks do not act quickly enough on the fluid to make a substantial difference. This is a reasonable assumption for the flash-flood scenarios we are trying to model because evaporation does not occur significantly on hour time scales, and soils often have little capacity once saturated, a common scenario in flood events. Thus, setting $L = \rho$ and $Q = 0$ we arrive at our mass conservation equation:

$$\frac{d\rho}{dt} + \nabla \cdot (\rho\vec{v}) = 0. \tag{6}$$

This becomes one of our modeled equations.

2.3 Using Darcy Flow and Deriving Darcy's Law for Fluid Propagation

Darcy Flow is given by

$$q = -\frac{k}{\mu}\nabla p \tag{7}$$

where q is the instantaneous flow velocity, k is the permeability, μ is the viscosity of the fluid, and ∇p is the drop in pressure between regions. This general equation says that the flow rate is directly proportional to the drop in pressure, which we use in our implementation.

Darcy's law relates fluid flux to the pressure gradient and permeability of the solid medium ϕ. Using the pressure gradient above we can determine how much fluid we expect to flow between cells based on the solid fraction using Darcy. A brief derivation of this relation starts with broader fluid equations that help us model fluid density ρ_f, velocity v_f based on the solid and fluid fractions ϕ_s and ϕ_f, respectively as well as the velocity of both the solid and fluid, v_s and v_f, and fluid mass, m_f.

The equations that drive our derivation are the Navier-Stokes equations, which is also seen as Newton's second law of motion for fluids. Newton's second law tells us that the rate of change of momentum of a property is equal to the applied force. For a fluid, the general control volume meets the condition that the net momentum flux is equal to the forces. The Navier-Stokes momentum equations more generally tell us that, in the case of a compressible Newtonian fluid, we have:

$$\rho\big(\frac{\partial u}{\partial t} + u \cdot \nabla u\big) = -\nabla p + \nabla \cdot \big(\mu(\nabla u + (\nabla u)^T) - \frac{2}{3}\mu(\nabla \cdot u)I\big) + F \tag{8}$$

where u is our fluid velocity, p is our fluid pressure, ρ is our fluid density, μ is our viscosity, and I, F are different incorporated forces. We can simplify this equation rapidly by plugging in our terms and using some reasonably mild assumptions to derive our next modeled equation:

$$\rho_f\big(\frac{\partial v_f}{\partial t} + v_f \cdot \nabla v_f\big) = -\phi_f \nabla p + \Gamma(\phi_s, \phi_f)(v_s - v_f) + m_f \nabla^2 v_f \tag{9}$$

$$\implies \rho_f(\frac{\partial v_f}{\partial t}) = -\phi_f \nabla p + \Gamma(\phi_s, \phi_f)(v_s - v_f) + m_f \nabla^2 v_f \text{ (nonlinearity)} \tag{10}$$

$$\implies \rho_f(\frac{\partial v_f}{\partial t}) = -\phi_f \nabla p + \Gamma(\phi_s, \phi_f)(v_s - v_f) \text{ (tractability)} \tag{11}$$

$$\implies \rho_f(\frac{\partial v_f}{\partial t}) = -\phi_f \nabla p + \Gamma(\phi_s, \phi_f)(-v_f) \text{ (assumption that solid is brittle and stiff)} \tag{12}$$

$$\implies \rho_f(\frac{\partial v_f}{\partial t}) = -\phi_f \nabla p - \lambda \frac{\phi_s^2}{(1-\phi_s)^2}(v_f) \text{ (substitution of } \Gamma = \frac{\phi_s^2}{\phi_f^2} = \frac{\phi^2}{(1-\phi)^2}) \tag{13}$$

$$\implies \frac{\rho_f}{\phi_f} \cdot (\frac{\partial v_f}{\partial t}) = -(\nabla p + \lambda \frac{\phi^2}{(1-\phi)^3} v_f) \text{ (division by } \phi_f = (1-\phi)) \ . \tag{14}$$

The right hand side, a product of fluid density and fluid acceleration, is small for most fluids compared to the pressure gradient so we can assume this is close to zero and drop the term. Rearranging solves for the fluid velocity:

$$0 \approx -\big(\nabla p + \lambda \frac{\phi^2}{(1-\phi)^3} v_f\big) \tag{15}$$

$$\nabla p = \lambda \frac{\phi^2}{(1-\phi)^3} v_f \tag{16}$$

$$v_f = \lambda \frac{(1-\phi^3)}{\phi^2} \nabla p \equiv \lambda \kappa(\phi) \nabla p \tag{17}$$

Thus deriving another of our final modeled 2D equations. Here the assumption is that fluid velocity is proportional to the pressure gradient and a function of the solid fraction ϕ.

2.4 Pressure Gradient

In order to calculate a pressure gradient we first calculate the pressure at grid cells based on the flux derived earlier. We assume that deviations from the baseline fluid density ρ_0 will give rise to pressure, and can thus define pressure as follows:

$$p(x,y) = \zeta \left(\frac{\rho_1(x,y)}{\rho_0(x,y)} - 1 \right) \tag{18}$$

Here ζ is a factor which we can adjust to change how quickly the material will erode. We set $\rho_1 = \rho_0$ so that initially there is not pressure as $\frac{\rho_1}{\rho_0} = 1$ and thus $p = \zeta(1-1) = 0$. As fluid is added and removed from the media the fraction $\frac{\rho_1}{\rho_0}$ changes, resulting in pressure differentials.

We can calculate the gradient of this pressure field to update the solid fraction ϕ.

$$\nabla p = \begin{bmatrix} p_x(x,y) \\ p_y(x,y) \end{bmatrix} \tag{19}$$

As we are only considering an erosion model and not a deposition model, we can make the following assumptions about the rate of erosion:

1. Below some threshold no erosion occurs. We write this as a function of the pressure.

2. Erosion above this threshold increases with the pressure gradient which we can express as follows:

$$\frac{\partial \phi}{\partial t} = -\phi \cdot \max\{0, |\nabla p|^2 - \Lambda(\phi)\} \tag{20}$$

where Λ is a function of the solid fraction that describes how easy it is to erode. For our model, we chose $\Lambda(\phi)$ to be a hyperbolic tangent activation function dependent on a ϕ^* parameter, which determines how quickly the media will erode. The hyperbolic tangent was chosen because of it's shape, not a physical property of the system. In general, we assume that the material is self-supporting, meaning that a higher solid fraction is more difficult to erode. We plot this below:

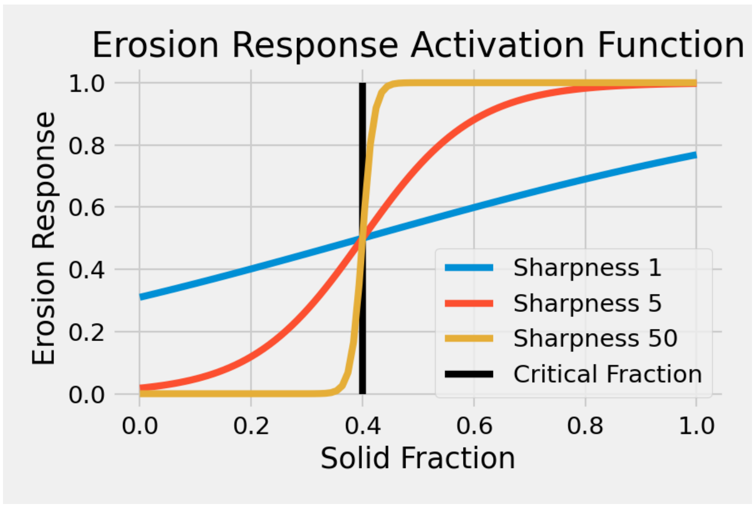

Figure 2: Erosion response function. The sharpness parameter can be tuned as a function of the media material itself. In general, we would expect stronger, more rigid materials to have a higher sharpness (w) parameter.

2.5 Complete Model

The complete 2-D theoretical model that we simulated is comprised of the following set of equations. We've built them into a flow chart in Figure 3 to visualize at what step they are used. The flow chart (aside from the initialization step) is repeated until the total time is reached.

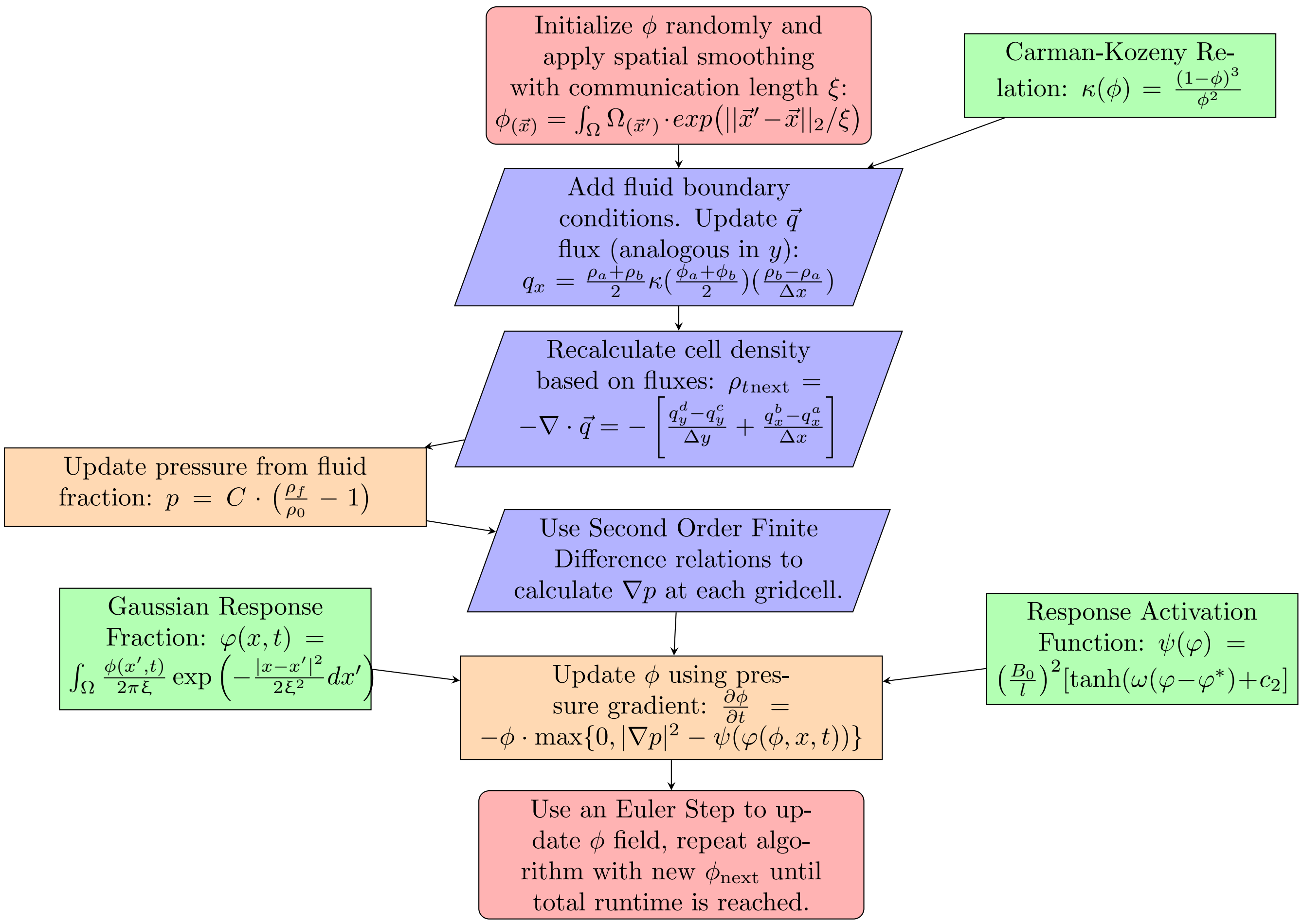
Initialize ϕ randomly and apply spatial smoothing with communication length ξ: $\phi_{(\vec{x})} = \int_\Omega \Omega_{(\vec{x}')} \cdot exp\big(||\vec{x}' - \vec{x}||_2/\xi\big)$
Carman-Kozeny Relation: $\kappa(\phi) = \frac{(1-\phi)^3}{\phi^2}$
Add fluid boundary conditions. Update $\vec{q}$ flux (analogous in y): $q_x = \frac{\rho_a+\rho_b}{2}\kappa(\frac{\phi_a+\phi_b}{2})(\frac{\rho_b-\rho_a}{\Delta x})$
Recalculate cell density based on fluxes: $\rho_{t\text{next}} = -\nabla \cdot \vec{q} = -\left[\frac{q_y^d - q_y^c}{\Delta y} + \frac{q_x^b - q_x^a}{\Delta x}\right]$
Update pressure from fluid fraction: $p = C \cdot \big(\frac{\rho_f}{\rho_0} - 1\big)$
Use Second Order Finite Difference relations to calculate ∇p at each gridcell.
Gaussian Response Fraction: $\varphi(x,t) = \int_\Omega \frac{\phi(x',t)}{2\pi\xi} \exp\Big(-\frac{|x-x'|^2}{2\xi^2}dx'\Big)$
Response Activation Function: $\psi(\varphi) = \big(\frac{B_0}{l}\big)^2[\tanh(\omega(\varphi-\varphi^*)+c_2]$
Update ϕ using pressure gradient: $\frac{\partial \phi}{\partial t} = -\phi \cdot \max\{0, |\nabla p|^2 - \psi(\varphi(\phi,x,t))\}$
Use an Euler Step to update ϕ field, repeat algorithm with new ϕ_{next} until total runtime is reached.

Figure 3: Algorithmic flowchart for a single Δt update step. Helper functions are in green while full grid implementation steps are in orange and purple.

2.5.1 Full Modeled Equations for Reference

We in include the full set of modeled equations for reference:

$$\frac{\partial \phi}{\partial t} = -\phi \cdot \max\{0, |\nabla p|^2 - \psi(\varphi(\phi,x,t))\} \qquad \text{Update Solid Fraction Rule} \tag{21}$$

$$v_f = -\kappa(\phi)\nabla p_n \qquad \kappa(\phi) = \frac{(1-\phi)^3}{\phi^2} \tag{22}$$

$$0 = \frac{\partial \rho_f}{\partial t} + \nabla \cdot (\rho_f \cdot v_f) \qquad \text{Updates Density} \tag{23}$$

$$p = C \cdot \big(\frac{\rho_f}{\rho_0} - 1\big) \qquad \text{Updates Pressure} \tag{24}$$

$$\psi(\varphi) = \big(\frac{B_0}{l}\big)^2[c_1 \tanh(\omega(\varphi - \varphi^*) + c_2] \qquad \text{Threshold Behavior} \tag{25}$$

$$\varphi(x,t) = \int_\Omega \frac{\phi(x',t)}{2\pi\xi} \exp(-\frac{|x-x'|^2}{2\xi^2}dx') \qquad \text{Gaussian Response Fraction} \tag{26}$$

The following modeled equations relate the flux field, which occurs on the boundary to the fluid density, which relies on subscript notation. We let a and b indicate edges to the right and left of cell x respectively, and c and d indicate cells below and above cell y, respectively. For more details see Figure 4 in the numerical section.

$$q_x = \frac{\rho_a + \rho_b}{2}\kappa(\frac{\phi_a + \phi_b}{2})(\frac{\rho_b - \rho_a}{\Delta x}) \qquad \text{Calculate } q_x \text{ at the edges} \tag{27}$$

$$q_y = \frac{\rho_c + \rho_d}{2}\kappa(\frac{\phi_c + \phi_d}{2})(\frac{\rho_d - \rho_c}{\Delta y}) \qquad \text{Calculate } q_y \text{ at the edges} \tag{28}$$

$$\rho_t = -\nabla \cdot q = -\left[\frac{q_y^d - q_y^c}{\Delta y} + \frac{q_x^b - q_x^a}{\Delta x}\right] \qquad \text{Calculate next density} \tag{29}$$

How these equations work as a system: We begin by updating our fluid velocity at each cell using equation 22 using the pressure from the previous pressure gradient. Intuitively, we can say that the velocity of a fluid is proportional to the product of the pressure and how much of the solid fraction is acting as a friction force. Using the updated velocity field we can then solve for the density field in equation 23. Once we've solved for the new density we can update the pressure field using equation 24. We can then use the new updated pressure rule in 21 to update the solid fraction along with the threshold behavior function and our Gaussian response fraction. Lastly, we use equations 27 and 28 to calculate our horizontal and vertical fluxes by taking a weighted average of density, multiplying that by a function of an average of our solid fraction, and then multiplying that by a density derivative. This goes into 29 to calculate our updated density. Using the updated density we calculate the pressure field in 24, then calculating the gradient we derive the update rule for ϕ in 21.

2.6 Numerical Model

We discretize the equations above in order to implement them in our numerical simulation. We chose Python for the implementation as it was our shared language despite obvious performance drawbacks. The general workflow is identical to the steps described in section 2.5. Implementation steps include time-integrating with a forward Euler finite-difference method and solving our partial differential equation. We set boundary conditions, update our density, calculate our pressure, and update our solid fraction to create our simulation.

2.6.1 Grid Discretization

Following the grid discretization in Derr's seminal paper, we divide up our domain $\Omega = [a_x, b_x] \times [a_y, b_y]$ into $m \times n$ cells of width Δx and Δy which are given by:

$$\Delta x = \frac{b_x - a_x}{m} \qquad \Delta y = \frac{b_y - a_y}{n} \tag{30}$$

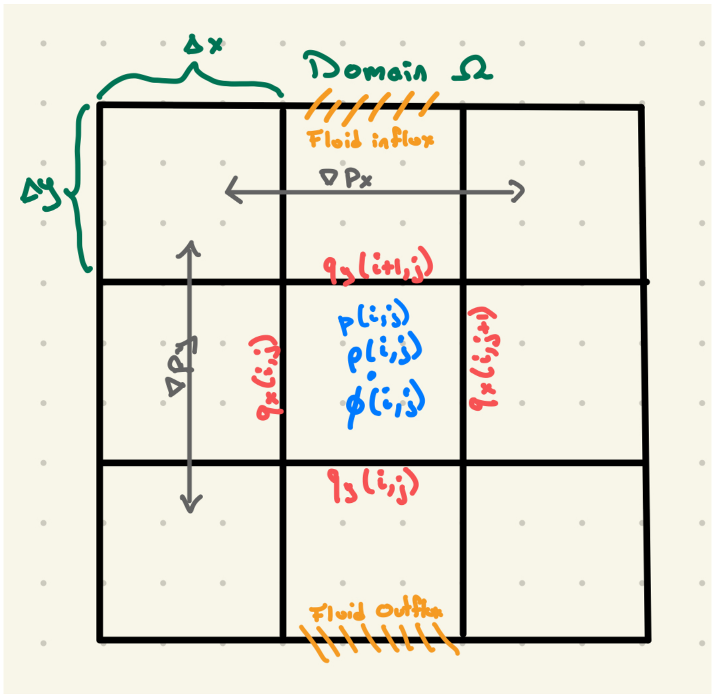

Figure 4: Domain Notation and Indexing. This figure serves as a reference for where we located pressure, fluid density, flux, and fluid forcing. It also shows the cells used for the second order central finite difference calculation of the pressure gradient in the x and y directions.

For our implementation, we set $m = n$ choosing to work with a square domain. Similar to Derr et al., we store the solid fraction and fluid pressure at cell centers. When considering the flux between cells, it made sense to store these parameters along the cell boundaries, specifically the edges which made sense from a mass conservation perspective as doing so ensures that we keep track of all the fluid. If there is a flux out of one cell, storing the directed value at the boundary allows us to measure it as flux *into* the neighboring cell.

2.6.2 Calculating Fluid Flux

As each cell is surrounded by 4 edges, we need two matrices q_x and q_y to store the flux. We derived these flux matrices using the previous fluid matrix ρ_t and the solid fraction ϕ_t. Broadly, we relate the fluid flux on an edge as the average pressure, the average solid fraction, and the pressure gradient of the two adjacent cells. This is given by the update equations 27 and 28. Looking carefully at the grid in figure 4, we note that there are actually $n+1$ rows in the q_y matrix and $n+1$ columns in the q_x matrix in order for the edge cells to have both a top and bottom boundary condition. When updating the flux matrices we explicitly set the fluxes at all boundary edges to zero. This is similar to the ghost node implementation done in the "Rocking out in Pierce Hall" homework problem we completed earlier in the semester. We note that we could also choose to set the flux across any boundary we chose and mass will still be conserved. Thus, for some edge (i, j) in the x-flux field $\mathcal{Q}^x_{(i,j)}$ we can express this fluid flux as a function of the fluid and solid fractions at the two neighboring grid-points. These points would be indexed at $\Omega_{(i,j-1)}$ and $\Omega_{(i,j)}$ as $\mathcal{Q}^x$ (see Figure 4).

2.6.3 Calculating the Pressure Gradient

The pressure gradient is the key driver of erosion as pressure, and not fluid volume for example, is able to dislodge grains in the media. We assume that there is some critical pressure threshold, which is a function of the solid fraction, that needs to be met for erosion to occur. In our numerical model, we place the pressure at the center of the grid cell (see Figure 4) and then calculate the gradient at that cell using a second-order centered finite difference approximation of the stationary pressure field p in both the x and y directions. Thus for a particular grid point $\Omega_{(i,j)}$ we can write the components as:

$$\nabla p^x_{(i,j)} = \frac{p\big(\Omega_{(i,j+1)}\big) - 2p\big(\Omega_{(i,j)}\big) + p\big(\Omega_{(i,j-1)}\big)}{\Delta x^2} \tag{31}$$

$$\nabla p^y_{(i,j)} = \frac{p\big(\Omega_{(i+1,j)}\big) - 2p\big(\Omega_{(i,j)}\big) + p\big(\Omega_{(i-1,j)}\big)}{\Delta y^2} \tag{32}$$

$$\implies \nabla p_{\Omega_{(i,j)}} = \begin{bmatrix} \nabla p^x_{(i,j)} \\ \nabla p^y_{(i,j)} \end{bmatrix} \tag{33}$$

We then use the norm of $\nabla p_{\Omega_{(i,j)}}$ in our update rule for the solid fraction below. One complication method with the above equation is that they only work for cells that are *not* edge or corner cells. For these cells we update using the second order forward finite difference formula for the bottom and left borders of Ω (the cells which are the first in their row or column). Similarly, we apply a second order backward finite difference formula for the right and top borders of Ω (the cells which are the last in their row or column). The forward and backwards difference zero-indexed equations become the following (we omit the y-direction as these are analogous):

$$\nabla p^x_{(i,0)} = \frac{p\big(\Omega_{(i,2)}\big) - 2p\big(\Omega_{(i,1)}\big) + p\big(\Omega_{(i,0)}\big)}{\Delta x^2} \qquad \text{Forward S.O. Difference} \tag{34}$$

$$\nabla p^x_{(i,n-1)} = \frac{p\big(\Omega_{(i,n-1)}\big) - 2p\big(\Omega_{(i,n-2)}\big) + p\big(\Omega_{(i,n-3)}\big)}{\Delta x^2} \qquad \text{Backwards S.O. Difference} \tag{35}$$

We can repeat a similar process for ∇p^y components at the boundary to derive the full pressure gradient at the edge cells.

2.6.4 Updating the Solid Fraction

Using the pressure gradient above, we determine the solid fraction update rule by approximating the derivative $\frac{\partial \phi_{(i,j)}}{\partial t}$, which is only a function of $\nabla p_{(i,j)}$ and $\phi_{(i,j)}$ itself. Numerically this becomes:

$$\frac{\partial \phi_{(i,j)}}{\partial t} = -\phi_{(i,j)} \cdot \max\{0, |\nabla p_{(i,j)}|^2 - \psi(\phi_{(i,j)})\} \tag{36}$$

$$\text{where} \quad ||\nabla p_{(i,j)}||_2^2 = {\nabla p^x_{(i,j)}}^2 + {\nabla p^y_{(i,j)}}^2 \tag{37}$$

Using this derivative we can calculate the update rule using a simple forward Euler step. We add a superscript to ϕ to denote the time step relation:

$$\phi^{t+1}_{(i,j)} = \phi^{t}_{(i,j)} + \Delta t \cdot \left(\frac{\partial \phi^{t}_{(i,j)}}{\partial t} \right) \tag{38}$$

Significantly, we note that the derivative is never positive, which makes sense for an erosion model as we are not considering deposition. Furthermore scaling by $\phi_{(i,j)}$ in equation 36 means that the solid fraction erodes proportionally to the amount of material left and therefore erosion slows down when the solid fraction approaches zero.

3 Results

We ran our simulation with a forward time step of $\Delta t = \frac{1}{2000}$ seconds for 400 steps over a domain Ω of dimension 100×100, resulting in a total simulated run of a fifth of a second. In Python, execution took ~ 50 seconds. We were discouraged from trying larger grid-scales due to the $\mathcal{O}(n^2)$ scaling and because we found that the simulation produced reasonable results. All of the code for replicating these simulations can be found at our GitHub project repository, Julians42/AM205_Final_Project.

3.1 Model Run Without Barriers

We first built our model to run simulations without the insertion of a dam. The results of such a run are shown below in Figure 5. We found that erosion was strongly dictated by variability in the domain Ω as erosion spread from the source and sinks along paths in the media which had the lowest solid fraction. Interestingly, the fluid pressure begins to show wavy lines of alternating positive and negative deviations - we imagine that, with the absence of a significant solid fraction, the fluid begins to slosh around in the empty spaces. A somewhat concerning observation is that the stripes are all oriented from the top left to the bottom right. We would expect them to be symmetrically driven from the sources and sinks, so perhaps these are an artifact of some numerical bug in our code. Or perhaps a zoo would hire us to repaint their zebras...

3.2 Adding the Dam

As we found that erosion propagates from the source and the sink, we concluded that if we were to implement a barrier to redirect erosion it would need to have structures on both sides and as a result a square barrier would work well. A model run with the barrier in place is shown in figure 7.

We find that the dam seems to be able to redirect the flow only for a few milliseconds before eroding. Thus, increasing the solid fraction is not a particularly effective method of dam construction, possibly because the flux is related to the average across neighboring cells so the erosion is able to "chip away" at the

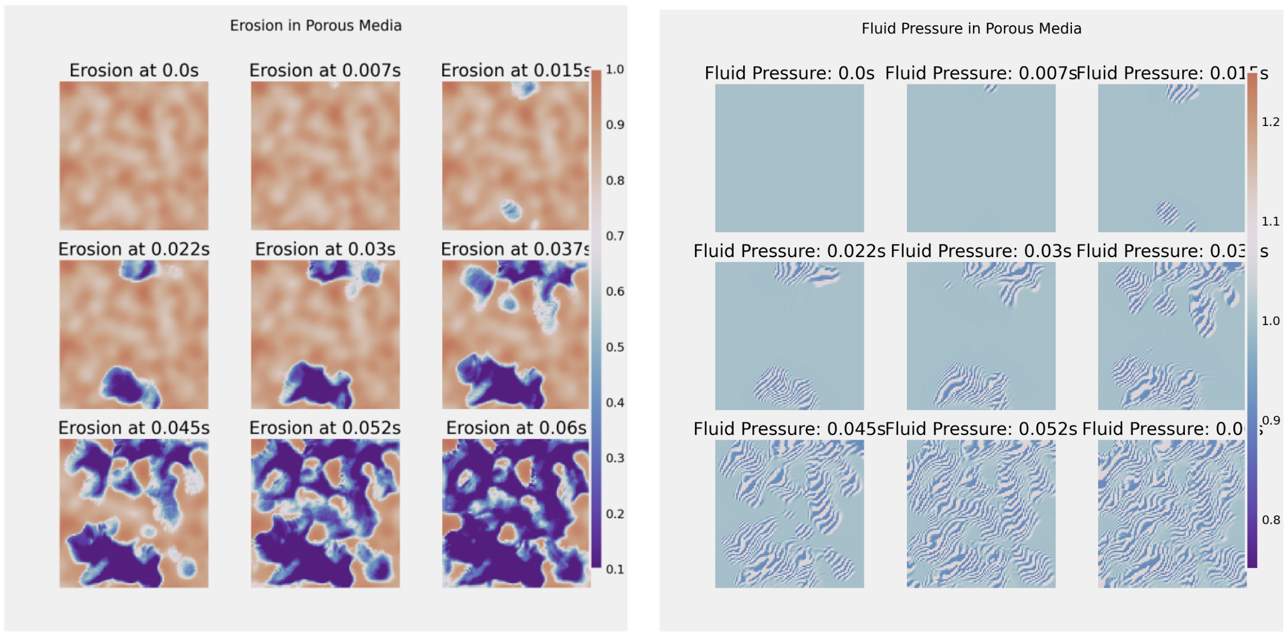

Figure 5: Erosion in simple smoothed porous media. Fluid is driven in through the center top of the domain and pulled out at center bottom. We show the corresponding fluid pressure fields at each time step on the right

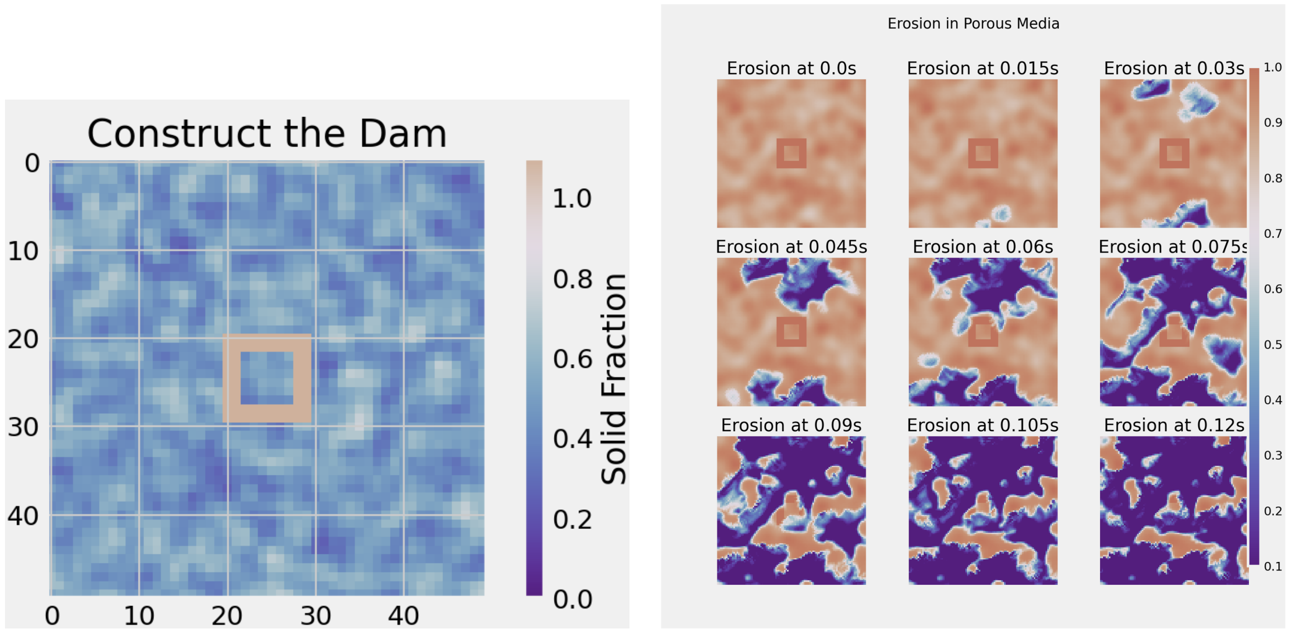

Figure 6: Dam Implementation to redirect erosion away from a valuable region - for example a house - we see that the dam is able to initially redirect the flow for a few milliseconds, but then fails. Clearly the dam technology needs improvement!

dam. We could consider adjusting the fluid flux or erosion dynamics in the dam region to see how erosion might progress under a different regime, for instance, we could decrease the fluid flux or modify ψ to more significantly increase the amount of pressure it takes to erode the dam.

4 Discussion and Future Extensions

We were hoping to see behavior more similar to that found in Derr et. al. 2020 of clear branching patterns and stream formation. Instead, it seems like the erosion we're doing is just forming lakes. Determining how to modify our model to more closely match these branching patterns is a clear next step for this project.

4.1 Adaptive Time-Stepping

One potentially highly effective next step towards improving the accuracy and computational efficiency of our model is to employ adaptive time-stepping. Currently, we have chosen a Δt based on the CFL condition, $\Delta t \propto \frac{1}{\Delta x}$, and adjusted the scaling of that condition through repeated simulation runs to ensure that nothing breaks. To avoid this ad-hoc method, and to potentially reduce the number of time steps and improve computational speed of the simulation, we could employ adaptive time stepping where we update our Δt based on the error between a single step of size Δt and two smaller steps of size $\frac{\Delta t}{2}$. Numerically this resembles the following, where f is a wrapper function that computes all the steps in Figure 3 at once. Our initial update rule is given by:

$$\phi^{t_{n+1}} = \phi^{t_n} + \Delta t f(t^n, \phi^{t_n}) \qquad \text{Original Update Rule} \tag{39}$$

A higher order update rule, which will allow us to derive an estimate of our error is given using two smaller time steps is:

$$\phi^{t_{n+1/2}} = \phi^{t_n} + \frac{\Delta t}{2} f(t^n, \phi^{t_n}) \tag{40}$$

$$\phi_*^{t_{n+1}} = \phi^{t_{n+1/2}} + \frac{\Delta t}{2} f(t^{n+1/2}, \phi^{t_{n+1/2}}) \tag{41}$$

We can then compare these two methods to get an error estimate via

$$\mathcal{E} \approx |\phi^{t_{n+1}} - \phi_*^{t_{n+1}}| \tag{42}$$

Then, after defining an error tolerance τ, the next time step can be determined as follows:

$$\Delta t_{n+1} \longleftarrow 0.9 \cdot \Delta t_n \cdot \left(\frac{\tau}{\mathcal{E}}\right) \tag{43}$$

The factor of 0.9 is to ensure that the higher order errors do not take us outside our desired error bound. Additionally, we could also employ some max/min relations to ensure the time step never go too big or too small. Richardson extrapolation and Runge-Kutta methods could also be employed to improve model speed and accuracy.

4.2 Addressing Inconsistency in Total Fluid

When running our model we observed that the total amount of fluid in the media fluctuated on the order of 1 or 2% of the total volume enclosed. This behavior began after a several steps and is unexpected given our modeled equations are supposed to be conservative. Further work needs to be done to investigate where the model is picking up this additional fluid flux.

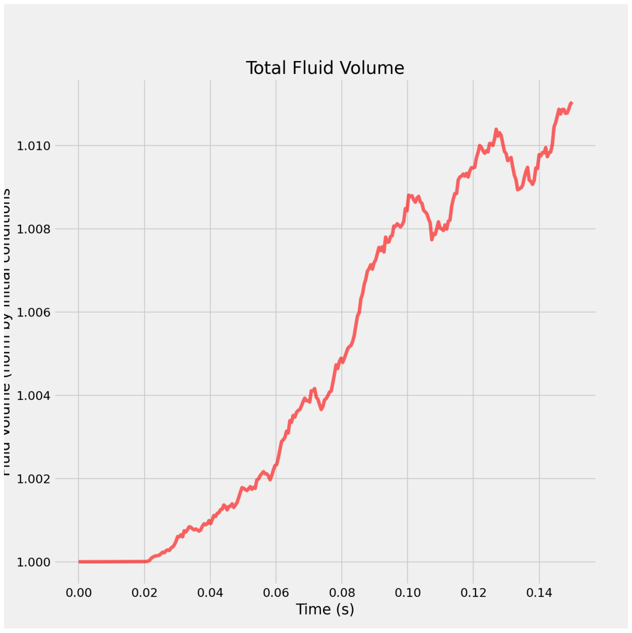

Figure 7: Surprising behavior - on the order of 1 to 2% of the total fluid is created or destroyed on a model run. The direction and magnitude varied based on run.

5 Conclusion

We implemented a two-dimensional compressed model to observe erosion behavior, in particular with the construction of dams, to address our original research question. We found that, while there was a short period of time where "inhabitants" may be protected from flooding and other types of water overflow, the constructed dam currently would not be enough to redirect the fluid. Upon reviewing the modeled assumptions this is likely due to the response function ψ in 25 as the tanh activation function ultimately limits the strength of the media. If we were to revise this function to consider media values larger than 1, we would possibly see an improvement in water redirection.

In the future, we hope to first analyze improvements in terms of model accuracy. Several additional parameters that we'd like to spend more time tuning include our threshold response function parameters, the erosion factor which determines how quickly the media erodes as a function of pressure, and how adjusting the location of fluid influx and outflux locations impact how erosion progresses. Another extension that we'd be interested in pursuing is simulating inflow and outflow patterns such as those of an actual river. In Derr's paper, the authors were able to visualize meaningful channel formation through the media. We'd have liked to have time to adjust parameters sufficiently to recreate these effects. If we could do that with fluid motion similar to those of natural waters around barriers, we expect that our results would be been both more realistic and beautiful.

Most of all, we hope that the trajectory of this work continues; the application of models like ours to natural disasters and other real-life scenarios holds significant meaning to us as both a way to understand the world around us and improve the lives of her inhabitants.

5.1 Acknowledgements

Thank you so much to all the AM205 teaching staff, especially Chris Rycroft, Nick Derr, and Jovana Andrejevic for helping with this project. We're so grateful for all the time and support you've given us throughout the semester.

References

[1] The Center for Agriculture Food and the Environment in the College of Natural Sciences at UMass Amherst. *Massachusetts Wildlife Climate Action Tool: Precipitation Changes.* URL: https://climateactiontool.org/content/precipitation-changes. (accessed: December 13, 2021).

[2] Yuwei Bao et al. "Mathematical Models and Simulations of Reconfigurable Flow Networks: Erosion, Deposition, Filtration and Growth". In: ().

[3] Daniel Brayton and Diane Munroe. *After Irene: Adaptation, Policy, and Management.* URL: https://www.middlebury.edu/media/view/422751/original/final_2012_es_irenereport_web.pdf. (accessed: December 17, 2021).

[4] Comsol. *MNavier-Stokes Equations.* URL: https://www.comsol.com/multiphysics/navier-stokes-equations. (accessed: December 17, 2021).

[5] Nicholas J Derr et al. "Flow-driven branching in a frangible porous medium". In: *Physical Review Letters* 125.15 (2020), p. 158002.

[6] Andrew Gibiansky. *Computational Fluid Dynamics.* URL: https://andrew.gibiansky.com/blog/physics/computational-fluid-dynamics/. (accessed: December 17, 2021).

[7] Andrew Gibiansky. *Fluid Dynamics: The Navier Stokes Equations.* URL: https://andrew.gibiansky.com/blog/physics/fluid-dynamics-the-navier-stokes-equations/. (accessed: December 17, 2021).

[8] Norihiro Izumi and Gary Parker. "Inception of channelization and drainage basin formation: upstream-driven theory". In: *Journal of Fluid Mechanics* 283 (1995), pp. 341–363.

[9] Norihiro Izumi and Gary Parker. "Linear stability analysis of channel inception: downstream-driven theory". In: *Journal of Fluid Mechanics* 419 (2000), pp. 239–262.

[10] Amala Mahadevan et al. "Flow-induced channelization in a porous medium". In: *EPL (Europhysics Letters)* 98.5 (2012), p. 58003.

[11] Norbert Schorghofer et al. "Spontaneous channelization in permeable ground: Theory, experiment, and observation". In: *Journal of Fluid Mechanics* 503 (2004), pp. 357–374.

[12] Hansjörg Seybold, José S Andrade, and Hans J Herrmann. "Modeling river delta formation". In: *Proceedings of the National Academy of Sciences* 104.43 (2007), pp. 16804–16809.

[13] Terence R Smith and Francis P Bretherton. "Stability and the conservation of mass in drainage basin evolution". In: *Water Resources Research* 8.6 (1972), pp. 1506–1529.

[14] Ahmad Zareei, Deng Pan, and Ariel Amir. "Temporal Evolution of Flow in Pore-Networks: From Homogenization to Instability". In: *arXiv preprint arXiv:2106.09745* (2021).