THE SHAPE OF STONES

JONATHAN ALLEN, JOHN WANG

Abstract.

1. Introduction

2. Discrete Model

In this section, we shall describe a model for simulating the erosion of a stone. In particular, we think of the erosion of a stone as coming from shear forces. This discrete model for eroding the stone allows us to simulate random processes and view the shape of a stone after some number of time steps. We provide computer simulations for the erosion process and analyze the model through these simulations.

2.1. **Shearing A Stone.** The discrete model for stone erosion is based on the idea that stones are eroded via chipping. The process of chipping is analogous to dropping a stone from a specified height and applying a shear force to a particular point on the stone. Intuitively, one can think of a stone in a stream being eroded by being tossed and turned by the force of the river. This tossing and turning causes the stone to collide with other stones on the bottom of the riverbed, causing chipping to occur.

We model the interaction between two stones as a shear force - a force that causes two parts of a stone to move in opposite directions. The point where a stone collides with another stone is the point where the shear force is concentrated. This causes the stone to break into two pieces. In mechanics, the shear stress τ applied to the stone is given by the following equation:

$$\tau = \frac{F}{A} \tag{1}$$

Where F is the force applied and A is the cross-sectional area of material with area parallel to the applied force vector. If the shear stress τ is above the stress that the stone's material can withstand, then the stone will crack.

Date: October 28, 2013.

Notice that the stress is proportional to the cross-sectional area of meterial which is parallel to the force vector. Thus, we can see that if the amount of force applied is constant, the total area of the material that will be sheared if shearing occurs will be constant as well.

2.2. **Definitions.** This section will define how we represent a stone in the discrete model.

We will represent a stone as a polygon with n vertices. Formally, a polygon s can be defined as the set of vertices $s = \{\mathbf{v}_1, \ldots, \mathbf{v}_n\}$, where a vertex $\mathbf{v}_i = (x_i, y_i)$ is a tuple in R^2. In a polygon s, vertices $\mathbf{v}_i$ and $\mathbf{v}_{i+1}$ are connected by a line segment for $i \in \{1, 2, \ldots, n-1\}$. In addition, vertices $\mathbf{v}_n$ and $\mathbf{v}_1$ are also connected by a line segment.

The centroid $\mathbf{c}_s$ of a stone s is the center of mass of the stone when the stone has uniform density. In other words, the centroid is the point $\mathbf{c}_s \in \mathrm{R}^2$ where the following holds:

$$\int_A (\mathbf{r} - \mathbf{c}_s) dA = 0 \tag{2}$$

Where A is the area of the stone.

2.3. **The Model.** With this in mind, we shall now develop a model for the erosion of a stone based on shearing. The main idea of the model is that we will represent a stone as a two-dimensional polygon, and chip off a constant amount of area of the stone at every time step. In this way, we will attempt to capture the process of a stone colliding with another stone on a riverbed. Randomness will be introduced into the model by randomly selecting somewhere on the stone to start the shearing.

A chipping process for a probability distribution P and area A, called $Chip(\mathrm{P}, A)$, is represented as follows:

(1) Select a vertex $\mathbf{v}_j$. To select this vertex, we find the centroid $\mathbf{c}_s$ of stone s and choose an angle $\gamma \in [0, 2\pi)$ uniformly at random. Now, define the ray $\mathbf{l}$ as the ray with an initial point of $\mathbf{c}_s$ which extends outwards infinitely at an angle of γ from the horizontal. The vertex with the minimum perpendicular distance to $\mathbf{l}$ will be $\mathbf{v}_j$.
(2) Given vertex $\mathbf{v}_j$, define $\mathbf{l}_1$ as the line segment which connects $\mathbf{v}_j$ and $\mathbf{v}_{j-1 \pmod n}$. Similarly, define $\mathbf{l}_2$ as the line segment connecting $\mathbf{v}_j$ and $\mathbf{v}_{j+1 \pmod n}$.
(3) Select a point $\mathbf{p}_1$ at random from $\mathbf{l}_1$. To do this, we select a random $t \in [0, 1]$ such that $\mathbf{p}_1 = \mathbf{v}_j t + (1 - t)$

3. Continuous Model

Method. We next moved on to develop a continuous surface erosion model. Physically this model represents a surface being worn down, such as a bar of soap in water or ice melting.

There are a wide range of different erosion processes, and most of them are very complex. To simplify our models, all of our processes erode (move) each point along the surface in a direction normal to the surface at the point. This rarely happens in real life, because of non-homogeneous materials, nonlinear behavior, etc., but it is a reasonable assumption to make for a simplified model.

Rather than modeling erosion as a discrete process, as was done in the previous section, we can model it with the following differential equation

$$\frac{d\mathbf{x}}{dt} = g(\mathbf{x})\ \hat{\mathbf{n}}(\mathbf{x}) \tag{3}$$

for some function $g(\hat{\mathbf{x}})$ to be defined shortly. $\hat{\mathbf{n}}(\mathbf{x})$ is the unit vector normal to the surface at $\mathbf{x}$.

The method used to represent the continuous surface will be explained momentarily, but first we will take a moment to explain the various erosion processes that we modeled.

The first process is a soap bar being worn down by a person holding it in their hand, which we will call our *smoothing model*. In this process, the corners of the soap bar will get worn down quickest, because they are the first parts of the surface to come into contact with the person's hand. Technically the furthest protruding bumps will be worn down quickest, but to simplify the problem, we consider every point of the surface to wear down at a rate proportional to the curvature at the point. We will use a signed curvature, so that positive curvature represents an outward facing bump, and negative curvature represents an indentation in the surface. $g(\mathbf{x})$ for this model is the following (κ is the curvature)

$$g(\kappa(\mathbf{x})) = \tan^{-1}(\beta\,(\kappa(\mathbf{x}) - \alpha)) + \frac{\pi}{2} \tag{4}$$

For our model we chose parameters, $\alpha = 1,\ \beta = 5$.

Our second process is an object sitting in a bath of acid, which we will call our *bottom recession model*. In this process, the bottom of the object will recede quickest. $g(\mathbf{x})$ for this model is the following ($x_{2,min}$ is the lowest point on the object)

$$f(\mathbf{x}) = \frac{1}{\alpha\left(x_2 - x_{2,min}\right) + \beta} - \gamma \tag{5}$$

$$g(\mathbf{x}) = \begin{cases} f(\mathbf{x}) & \text{for} \quad f(\mathbf{x}) > 0 \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

For our model we chose parameters, $\alpha = 10,\ \beta = \gamma = 0.1$.

Example Surface. The example shape that we will use is shown in Figure 1.

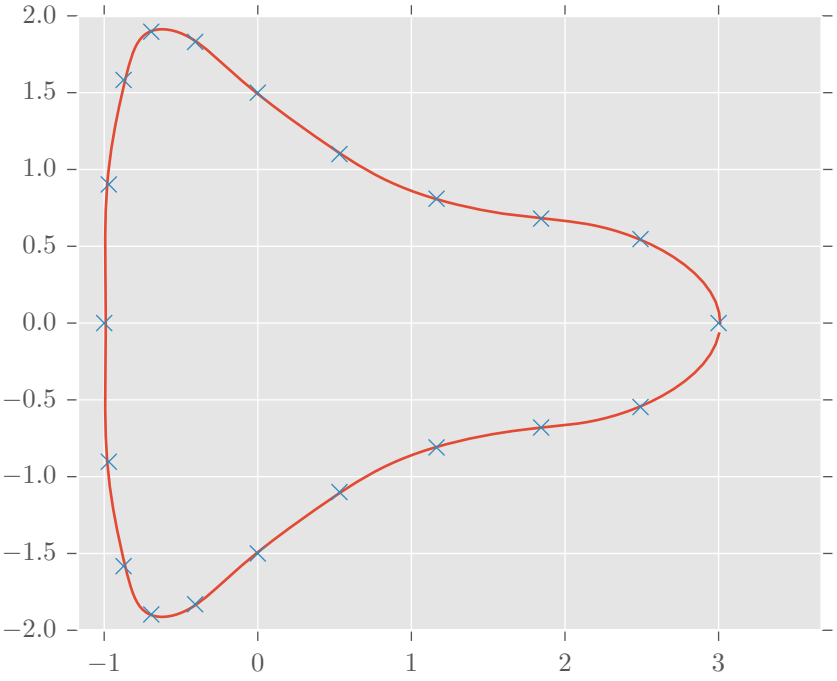

FIGURE 1. Example shape.

The equations for the shape are the following (T_i is the i$^{\text{th}}$ Chebyshev polynomial)

$$x_1(s) = \cos(s)\left(2T_0(\left(\frac{s-\pi}{\pi}\right)) + T_2\left(\frac{s-\pi}{\pi}\right)\right)$$
$$x_2(s) = \sin(s)\left(2T_0(\left(\frac{s-\pi}{\pi}\right)) + T_4(\left(\frac{s-\pi}{\pi}\right))\right)$$

Surface Equations. Our surface is modeled with two vectors, $x_1, x_2 \in \mathbb{R}^N$ for some resolution N. Then $\mathbf{x} \coloneqq \left[x_1^T, x_2^T\right]$, with x_1 and x_2 being the two coordinates of points sampled counterclockwise along the surface.

$$\text{(Velocity)} \qquad \dot{\mathbf{x}} \coloneqq \frac{d\mathbf{x}}{dt}$$

$$\text{(Acceleration)} \qquad \ddot{\mathbf{x}} \coloneqq \frac{d^2\mathbf{x}}{dt^2}$$

$$\text{(Curvature)} \qquad \kappa = \frac{\dot{\mathbf{x}} \times \ddot{\mathbf{x}}}{\dot{x}^3}$$

Numerical Modeling. We chose to model the surface parametrically with a Discrete Fourier series using spectral methods. Our processes tend to smooth the surfaces out over time, so spectral methods would capture this much better than a local approximation of the surface using finite differences which only act locally at points. All of our surface are in $\mathbb{R}^2$, so we will use the Real Fast Fourier Transform (equations in the appendix). The following equations are derived in most introductory spectral method texts, so they will be stated without proof here

$$\dot{\mathbf{x}} = RFFT^{-1}(diag(i\,[0 : N/2, 0])\ RFFT(\mathbf{x})) \tag{7}$$

$$\ddot{\mathbf{x}} = RFFT^{-1}\big(diag(i^2[0 : N/2, 0]^2)\ RFFT(\mathbf{x})\big) \tag{8}$$

Note: $diag(\mathbf{a})$ is the diagonal matrix with the elements of $\mathbf{a}$ along the diagonal.

Using the FFT, the surface normal and curvature can now be easily found

$$\hat{\mathbf{n}} = [-\dot{x}_2, \dot{x}_1] \tag{9}$$

$$k = \frac{\dot{x}_1\ddot{x}_2 - \dot{x}_2\ddot{x}_1}{\left(x_1^2 + x_2^2\right)^{3/2}} \tag{10}$$

Point Collision Problems. Unfortunately, the simplicity gained from modeling the surface with Fourier series is lost when the surface starts to shrink. As can be seen in Figure 2, the process works well for a short time period, but over time a sharp point, and the surface actually folds over itself (a physical impossibility).

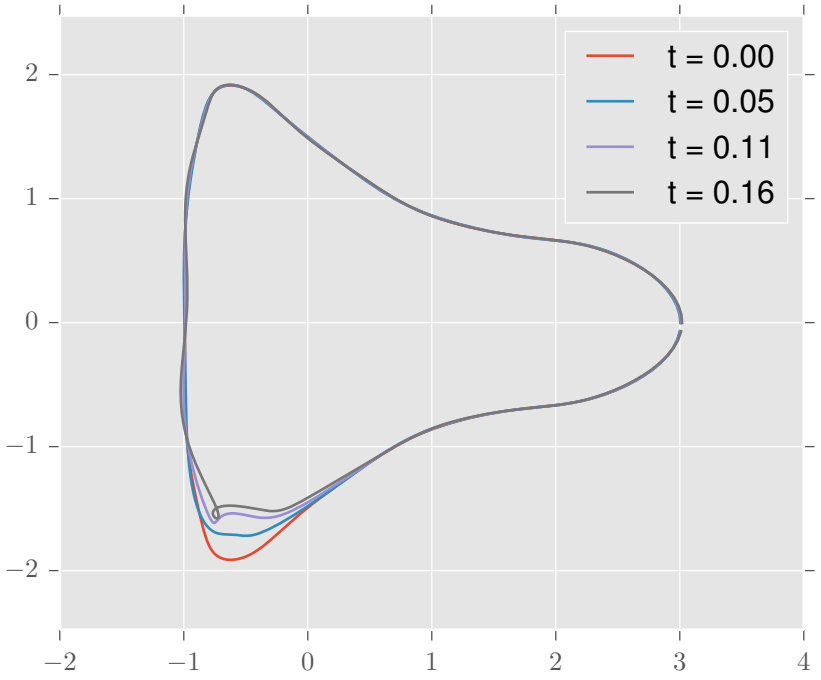

FIGURE 2

To see why this occurs, we will show a zoomed in portion of the same figure, but with the evaluation points shown.

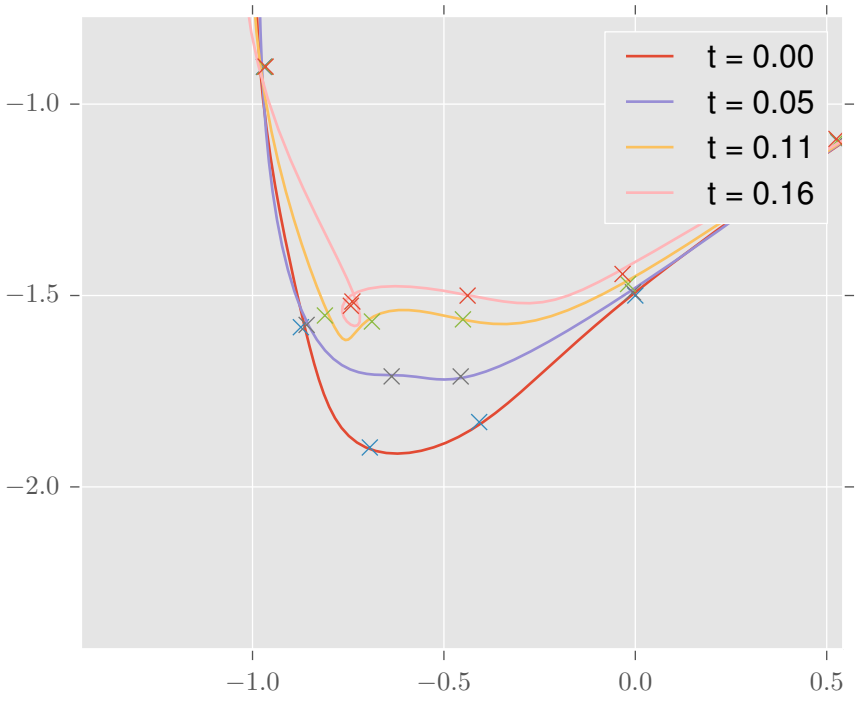

FIGURE 3

Looking closely, it can be seen that the sharp corner forms when evaluation points get too close to each other. Our guess is that this occurs because we are using discrete time stepping, so there are errors in the approximation of the shape's change with time. This is fine when the evaluation points are far from each other, but when they get close to each other, the errors become significant and affect the direction of the surface unit normal vectors.

Redistributing Points. The most obvious way to fix this problem would be to redistribute the points evenly on the surface at every step. This wouldn't be too difficult of a task, since we can easily integrate with the FFT.

Unfortunately, if we arbitrarily move points around, we lose spectral accuracy. Spectral methods . This method turns out to work, but I will hold off explaining it

High Frequency Damping. Our first approach to improve the algorithm was to use fewer Fourier terms than evaluation points, in an

attempt to better approximate the shape while ignoring the higher frequencies. This unfortunately performed even worse than our original algorithm.

Partial High Frequency Damping. In the *bottom recession model*, x_1 should be changing slowly and smoothly with time, since the recession model primarily moves points upwards. In Figure 4, x_1 is plotted, and we can see that this is not the case in our simple algorithm and large high frequency oscillations appear as time increases.

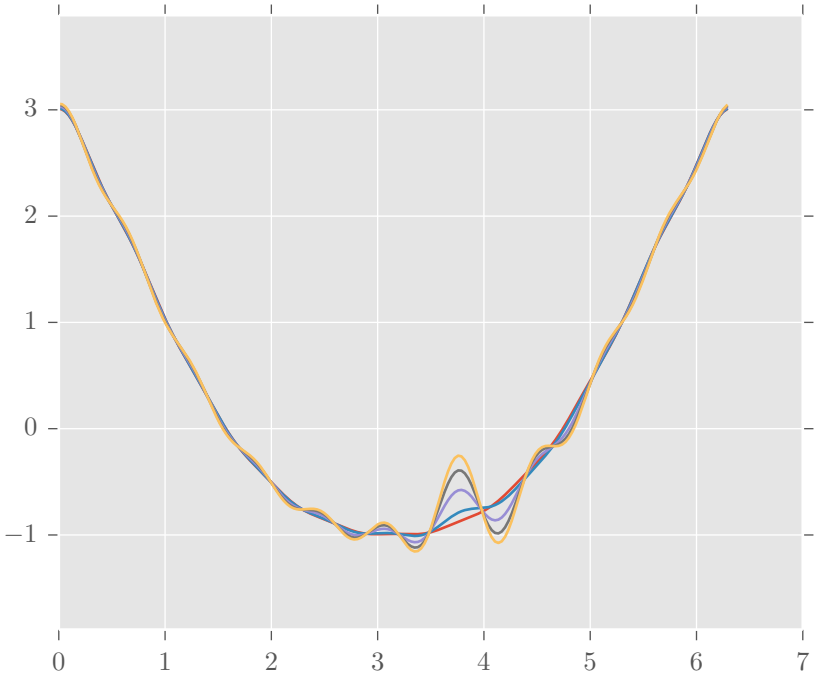

FIGURE 4

This led us to try to artificially dampen the high frequencies in x_1, by manually zeroing out the high frequency Fourier coefficients. This worked reasonably well, but was not a very robust method. Our shape is nice, because x_1 can be represented accurately with very few Fourier terms, but this would not be the case for more complicated shapes and other models, such as the *smoothing model* that affects x_1 and x_2 equally.

Continuous Redistribution. To redistribute points, we need to find some smooth function that will map points spaced evenly on the surface to evenly distributed *theta*. This can be achieved by

$$\frac{d\mathbf{x}}{dt} = g(x)\ \hat{\mathbf{n}} + f(\mathbf{a}_t) \tag{11}$$

$$\mathbf{a}_t = \frac{\ddot{\mathbf{x}} \cdot \dot{\mathbf{x}}}{\|\dot{\mathbf{x}}\|} \frac{\dot{\mathbf{x}}}{\|\dot{\mathbf{x}}\|} \tag{12}$$

$\mathbf{a}_t$ is the component of $\ddot{\mathbf{x}}$ tangent to the surface. $\dot{\mathbf{x}}$ is inversely proportional to the density of points on the surface. We want to shift points away from denser areas, which means moving in the direction of larger $\dot{\mathbf{x}}$.

As can be seen in Figure 5, this dramatically improves our algorithm.

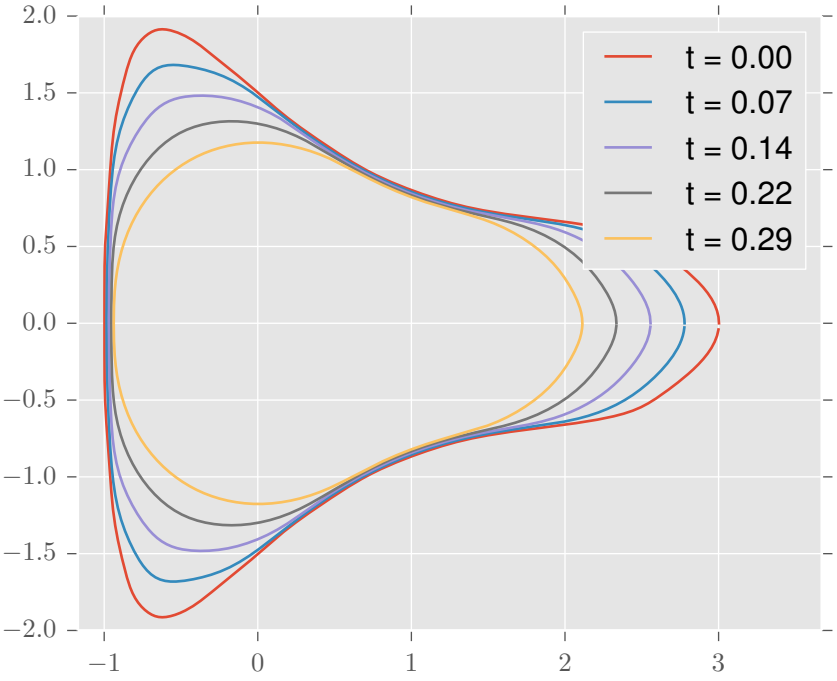

FIGURE 5

If we plot the evaluation points (Figure 6, it can be seen that the points redistribute themselves over time to stay evenly spaced on the surface.

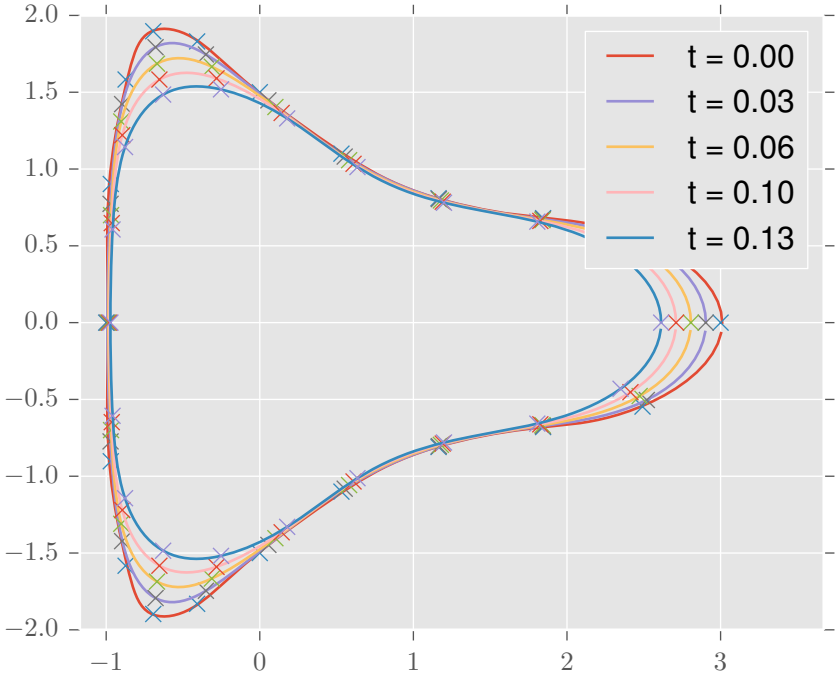

FIGURE 6

APPENDIX

Real Fourier Transform. The Real Fourier transform is defined as

$$\begin{aligned}\hat{\mathbf{x}} &\coloneqq RFFT(x)_k \\ &\coloneqq \sum_{n=0}^{N-1} e^{-2\pi i k(n/N)} x_n \qquad \text{for} \quad k = 0, \ldots, N/2+1\end{aligned}$$

Note: the terms FFT and Fourier transform are used interchangeably in the paper. Technically they are not synonyms, but it is common practice to use the terms interchangeably.

And the inverse transform is defined as

$$\begin{aligned}x &\coloneqq RFFT^{-1}(\hat{\mathbf{x}}) \\ &\coloneqq \frac{1}{N}\left(\sum_{k=0}^{N/2+1} e^{2\pi i k(n/N)} X_k + \sum_{k=N/2+2}^{N-1} e^{2\pi i k(n/N)} \bar{X}_k\right) \qquad \text{for} \quad n = 0, \ldots, N-1\end{aligned}$$

4. Conclusion