EOS 491 // EOS 518: Assignment 1.2

Sedimentary Transport (Numerical Model)

September 29, 2020

Working with your assigned partner(s), please answer all the questions below. Spend time working on this assignment before Friday's class, when you will be asked to show Blake and Jon your work to date and get guidance / feedback. You are not excluded from working with other groups, but each person will submit their own copy of the assignment.
Due date: October 06, 2020 by 11:30AM, via upload PDF to Brightspace

Name: ______________________________

Question	1	2	3	**Total:**
Marks:	5	5	2	12
Score:				

Bulk sediment transport: diffusion

The change of elevation over time is proportional to the second partial derivative of the topography with respect to space (the curvature). This equation is sometimes referred to as the *hillslope application* of the **diffusion equation**:

$$\frac{\partial h}{\partial t} = K \frac{\partial^2 h}{\partial x^2} \qquad \text{(diffusion equation)}$$

The diffusion equation is derived from combining the **continuity equation**, which expresses the conservation of mass:

$$\frac{\partial h}{\partial t} = -\frac{\partial S}{\partial x} \qquad \text{(continuity equation)}$$

with an expression of sediment transport rate, S, as a **diffusive flux**. In this formulation, S is linearly proportional to slope:

$$S = -K \frac{\partial h}{\partial x} \qquad \text{(diffusive flux)}$$

In part **1.1** of this assignment we looked at a specific application of bulk (diffusive) transport with an *analytical* solution. This you will be designing a numerical model of bulk transport using an *implicit* finite difference scheme (specifically the Crank-Nicolson algorithm). The questions below are designed to showcase that your model is correctly solving the diffusion equation.

Crank-Nicholson implicit scheme

The Crank-Nicolson method is numerically stable, implicit, and second-order in time – $O(\Delta t^2)$. The method solves for the next time-step iteration of the system by taking the average of the central-difference estimate of the second partial derivative of the topography with respect to space at both the current time step and the future timestep. Red terms are unknown at the current timestep (this is the *implicit* part of the numerical method).

$$\frac{{\color{red}h_i^{t+\Delta t}} - h_i^t}{\Delta t} = \frac{K}{2}\left(\frac{h_{i-\Delta x}^t - 2\ h_i^t + h_{i+\Delta x}^t}{\Delta x^2} + \frac{{\color{red}h_{i-\Delta x}^{t+\Delta t}} - 2\ {\color{red}h_i^{t+\Delta t}} + {\color{red}h_{i+\Delta x}^{t+\Delta t}}}{\Delta x^2}\right)$$

It is useful to define r as (this number is sometimes called the Fourier number):

$$r = K \frac{\Delta t}{2\ \Delta x^2}$$

to simplify the equation to the following form:

$${\color{red}h_i^{t+\Delta t}} - h_i^t = r\ (h_{i-\Delta x}^t - 2\ h_i^t + h_{i+\Delta x}^t + {\color{red}h_{i-\Delta x}^{t+\Delta t}} - 2\ {\color{red}h_i^{t+\Delta t}} + {\color{red}h_{i+\Delta x}^{t+\Delta t}})$$

This equation forms a system of linear equations that can be rearranged and solved using linear algebra. The general form of your problem (once you have collected unknowns on one side and knowns on the other) will be:

$$A{\color{red}h^{t+\Delta t}} = Bh^t + b^t$$

where A and B are square matrices with whose side-length is equal to the length of h and b is a vector of boundary conditions (additional flux in or out of each finite element in h_i). The matrix equation above is solved by multiplying both sides by A^{-1}:

$${\color{red}h^{t+\Delta t}} = A^{-1}(Bh^t + b^t)$$

Questions

Make a model with the following initial boundary conditions. Your topographic profile should cover 10 kilometers. The initial topography will be randomly generated as a cumulative sum of 1000 random draws from a normal distribution with $\mu = 0$ and $\sigma = 1$. Your left boundary condition will be +10 m, and your right boundary condition will be −20 m. The diffusivity, K, should be set to 2×10^2 (m^2/yr).

```
import numpy as np

#set your random number seed to 2:
np.random.seed(2)

#initial topography (1000 length covers 10 km with dx=10):
H=np.cumsum(np.random.normal(0,1,1000))+10

#fix right and left boundaries
left_boundary = 10
right_boundary = -20

#set diffusivity
K = 2e2
```

(if you are not using python, initiate your initial topography with a similar approach)

Question 1 (5)
Make a plot showing topography (y-axis) versus distance (x-axis) for your model at the start, after 500 years, and after 50,000 years.

Question 2 (5)
Starting with the equilibrium topography (the 50,000 year surface) from question 1, introduce a constant sediment flux of 10 m^2 per year at the coastline (where topography first drops below 0). Make a plot showing the new topography (y-axis) versus distance (x-axis) for your model after 50 years, and after 1,000 years.

Question 3 (2)
Show that the amount of sediment added (your flux multiplied by time) is equal to the total change in topography.