Erosion Model

Final Report

Team 45

Mi Deng
Sam Kester

Las Cruces High School
Las Cruces, NM 88005

Teacher: Mr. Greg Marez

April 2007

Table of Contents

Abstract

Erosion, especially erosion from rain, is a major force in land degradation. It displaces soil into the waterways and leads to loss of arable land and starvation. We modeled daily erosion with JAVA on a given piece of forested land over days as trees were being destroyed. Our model also includes many other factors: the area under consideration, the number and type of trees, the amount of rain it receives, the typical wind speed, and slope of the land. We use iterations to model each day individually. This model reduces all the variables into numbers and equations which are user inputs. By doing this, we can predict erosion from given land use and conversely determine the needed land use to reduce erosion to desired levels.

Project Background

The currently accepted equation for estimating of soil erosion from rain is the Universal Soil Loss Equation or USLE. It is A=RKLSCP where A is erosion in tons/hectare/year, R is the rainfall erosivity factor, K is the soil erodability factor, L is slope length factor, S is the slope in %, C is the cover or plants growing on the soil, and P is the land management factor (terraces, windbreaks, forest floor conditions). This model was originally designed for farmers and sub factors for P were added for forests.

Although the numbers from this equation are sound, we feel that it generalizes too much. Firstly, erosion happens on a daily basis when it rains. Even though many days do not rain and each individual storm only adds a fraction to the yearly total, it is more accurate to consider a smaller increment. Also, rains are seasonal in many areas and erosion is seasonal and has wilder fluctuations than predicted because USLE can output one unchanging number per one tract of land. Many of the variables are also too general. R can be described in several variables that describe total rainfall, raindrop size, and wind. Similarly, C can be described more specifically by measuring protective effects of the roots and leaves on soil erosion. Nevertheless, all the other variables are simple enough or in the case of soil erodability, slope, and the land management factor, cannot be made more accurate without a 3-D topographical map, which defeats the purpose of equation modeling.

Secondly, another problem with USLE is that the numbers are factors which must be determined experimentally. This causes the numbers to be removed from reality until put into perspective and prevents easy analysis of natural data that can be determined from field data gathering because the conversion between that and erosion must indirectly be converted to these factors.

Computational Principles

In a forest, trees are the most effective plants in stopping erosion. Among the tree, it is the roots that hold down the soil, so in our model, we only considered trees and their roots as important. In other words, we assume that erosion would occur unless a circular area held by trees' roots completely prevented it. (In reality, tree leaves slow the rain's velocity and decaying leaves block water from flowing downhill.) However, the area protected by tree roots is difficult to measure because tree roots grow in all directions. Thus, we must extrapolate by finding correlations between root spread and the surface area of other parts of the tree.

Such correlations do exist between the tree root area and the crown spread. Crown spread is defined as the circular area of the tree's crown as measured by the average of the areas of rotating the axes by 45 or 90 degrees. Since the area is circular, the tree crown also has a diameter and by the widest estimates, the root's diameter is 2-4 times greater than the crown's diameter, so the area is 4-16 times bigger. However, the calculation of a tree's crown is difficult because like the roots, it is a tangled, branching network. Again, we must find something much easier to measure on a tree.

Fortunately, crown spread has a linear correlation to diameter at breast height of a tree (dbh). This is defined as the diameter of the tree's trunk 3 feet from the ground. This is much more easily determined and since there's a linear correlation to crown spread, which is a factor of the root spread, we can determine root spread from a relatively simple measurement.

However, since there is variation in tree maturity and tree type in a given ecosystem, we only found the maximum dbh of a tree type which we assumed as a representative of the ecosystem and introduced a random number function that sets maturity as a decimal with 1 being the maximum. The user inputs a minimum maturity and a variable on the range of the trees' maturity. By doing so, the dbh is measures the average maturity of all the trees in a forest.

Thus, the equation for the erodable area is:

$A=t-([a\{mx+b\}]/2)^2*\pi*y$, where a is the ratio from root to crown diameter, x is the dbh (which is random), m and b are the linear correlation function parameters, y is the number of trees, π is circumference/diameter or 3.14159265…, and t is the total area. All of the above variables are user inputs.

Displacement is soil that's removed from its original location while erosion is soil that's carried by water to the bottom of the slope. Displacement isn't erosion; it is always is much greater and is caused primarily by rain and will still occur on flat land while erosion cannot occur without slope. Displacement is important because displaced soil is removed from a biological matrix that keeps it in place and blocks future water penetration into the soil, meaning that displaced soil is added to erodeable area in successive iterations. In other words, displacement loosens and clogs the soil allowing tomorrow's erosion of the displaced soil.

USLE's R factor is split as follows. Rain is assumed to be uniform over the plot of land, and three variables modify the rain's quantity: the probability of rain, which is determined by creating a random number less than the user input decimal between 0 and 1, the intensity of rain in cm/hr, and the duration in hours. The latter two variables are also user inputs. From this, the total rain is determined. However, different sized raindrops have different mass and terminal velocity, causing them to hit the ground with different energies. For our model, we assumed that there were 3 basic types of spherical raindrops: large ones that are .508 cm in diameter and fall at 32 m/s, medium ones that are .2032 cm in diameter and fall at 22 m/s, and small ones that are .02032 cm in diameter and fall at 2 m/s. The user inputs the percentages of each type of raindrop. Since water's density is 1, we can easily determine the volume and the mass and the energy the rain exerts on the ground when it hits the ground in joules. However, storms also produce wind. Both rain and wind are vectors; we can add them and find the combined vector force of rain and wind. We assumed that wind and rain are perpendicular. $E=mv^2/2$, so m=volume of given raindrop size*percentage of total land*total erodeable land*total rain per storm (which can be 0) while $v^2/2$=(rain velocity^2+wind velocity^2)/2 or in variables arranged in the same order: $E_r=4/3*\pi*(D/2)^3*p/100*A*R*(v^2+w^2)/2*5/18$, where 5/18 is a unitary conversion factor.

However, the land itself absorbs much of this energy and the slope of the land affects the energy's effect of the rain. USLE's soil erosivity factor or K covers the absorption of energy by the earth. Now, deem the counterclockwise angle the combined rain and wind vector makes from the perpendicular of flat ground A and the ground's angle from the flat ground in a clockwise direction as B. If we draw a hypothetical perpendicular vector to the angled ground, we find that the vector on that triangle that points downwards on the slope is $\sin(A-B)*E_r$.

Thus, the equation for soil displacement is:

$D= \sin(A-B)*E_r*K$ where everything is either an input variable or is derived from such.

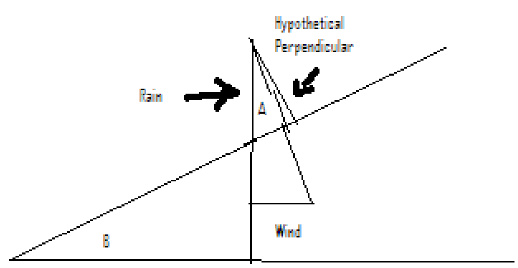

Figure 1. Angles A and B and the Hypothetical Perpendicular Vector

To model the erosion of the soil, we must model the flow of water as it moves downhill, carrying dirt with it. The equations that model this are the Navier-Stokes Equations (or more precisely, we are using the solutions to the differential equations). These equations describe the flow, average velocity, and force exerted against the surface by a flowing liquid. However, the NSE (Navier-Stokes Equations) assumes that the flowing liquid's force does not alter the surface it flows against. Obviously, that is false since the modification of the earth and soil is erosion. If one assumes such modification of the surface, one can no longer find a solution to the differential equations of the

Navier-Stokes Equations. To sidestep this issue, we exploit a computer's ability to make lots of calculations per second to find numeric solutions. In other words, we assume that for some short time interval, the difference between the ideal state of the NSE and reality is negligible and can be ignored. Every time interval, we update the velocity, density, viscosity, and distance remaining to the bottom of the sloped land while another variable records the erosion of every time interval. Meanwhile, that updated data is then the inputs for the next time step, allowing the program to run itself and iterate. The variable that stores the sums of the erosion is then multiplied by the number of time increments in one hour and by the number of hours. This also allows for the consideration of the water from a previous time step's rain, shielding the ground from current rain. We chose 1 second to be our time step.

The equations are:

$V=\rho g h^2 \sin(B)/(3\mu)$

$E=(V/h)(3\mu)*L*V*K*M$, where: $\rho=1.0$ (water's density), $g=9.8$ m/s^2, $h=R_i/3600*0.01$; R_i=Rain intensity per hour in cm, B is the angle previously mentioned, $\mu=0.001$kg/m/s (water's viscosity), L is the length of area considered, K is the USLE soil factor as previously mentioned, and M is the soil moisture factor where 0 is completely dry.

Our iteration method works as follows:

$\rho=\rho_m/\rho_v$ and at every iteration, $\rho_m=\rho_m+ E/A*1000+h$ where A is the total erodeable area as previously mentioned. At every iteration, $\rho_{v=}\rho_v+E/A*1000*(1/\rho_s)+dTemp$ where ρ_s is the density of the soil.

$\mu=\mu_n/\mu_d$ and at every iteration, $\mu_n=\mu_n+E/A+h$ and $\mu_{d=}ds$ where d is distance or at every iteration, d=d+V and s is time in seconds or at every iteration, s=s+1.

The use of E/A is to determine the change in mass (kg) of the water per unit of area (m^2), since the change in weight is dependant on it while the change in distance and time is not dependant on area. The user input variables are ρ_s, M, and L.

While those two variables are recycled back into the original equation to iterate the function, $E_1=E_1+E$ at every iteration while t/L=t/L-V at every iteration where t is the total area as previously mentioned. This recalculation continues until t/L=0. Then, the final erosion of one day is calculated from $E_{1.}$ $E_2=E_1+E_1(3599)V_0R_t$ where V_0 is V at its initial values and R_t is the time in hours of rain. This equation accounts for the water from previous rain shielding the soil from the current time step's rain.

To simulate the effects of erosion over many days with deforestation, A is increased by decreasing y, the previously mentioned variable that is the number of trees at each day by a fixed user input amount, in which the erosion iterations run.

Data and Analysis

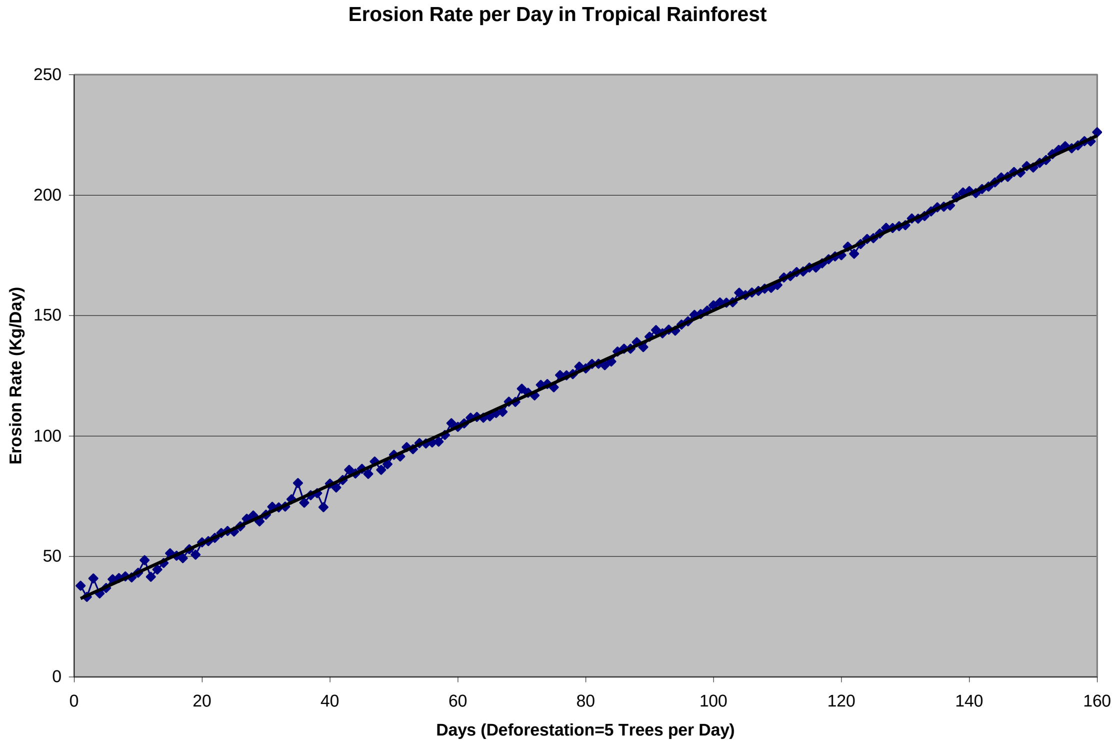

Figure 2: Erosion Rate in Tropical Rainforest. 795 Trees, 1,000,000 meters^2, and representative tree is white oak. K=0.35, M=0.35, Rain Probability=0.55 Rain Intensity=1 cm/hr.

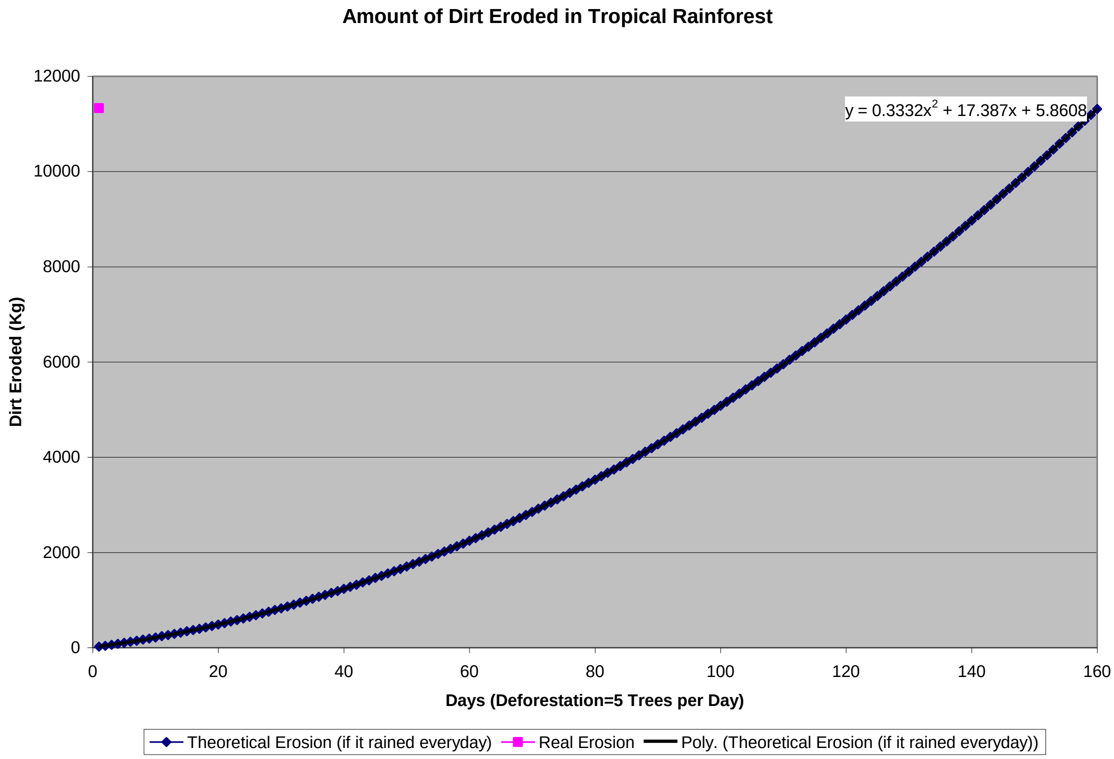

Figure 3: Erosion in Tropical Rainforest. 795 Trees, 1,000,000 meters^2, and representative tree is white oak. K=0.35, M=0.35, Rain Probability=0.55 Rain Intensity=1 cm/hr.

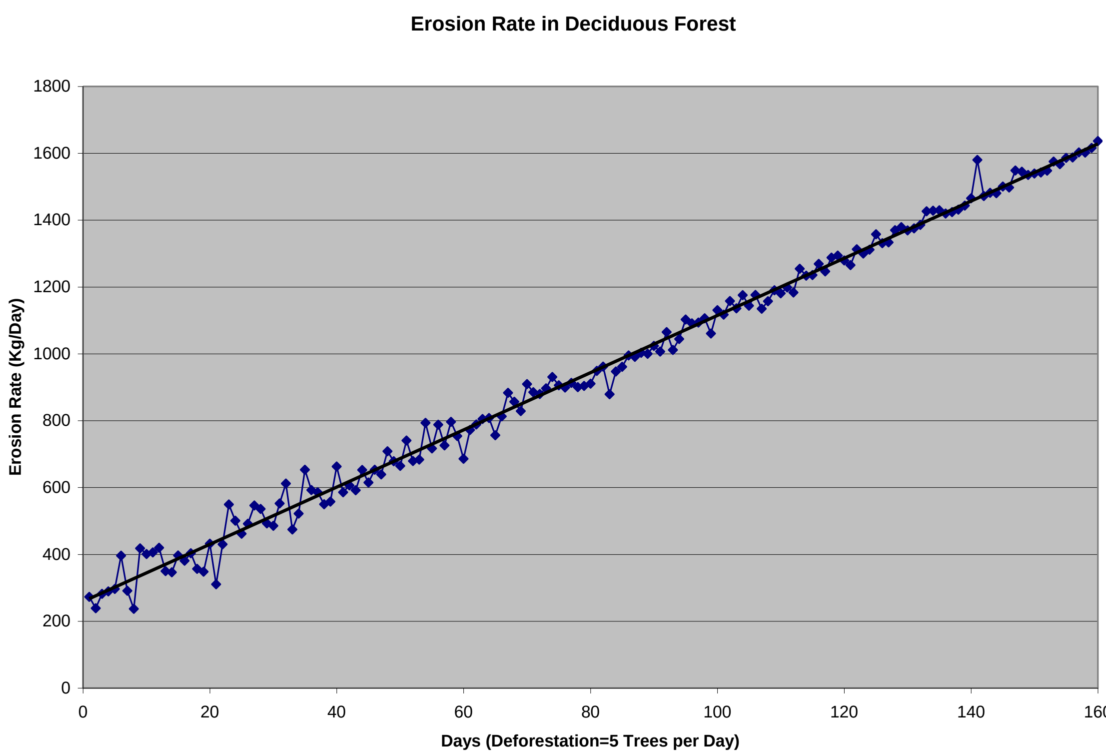

Figure 4: Erosion Rate in Deciduous Forest. 795 Trees, 1,000,000 meters^2, and

representative tree is white oak. K=0.35, M=0.35, Rain Probability=0.142857 Rain Intensity=2 cm/hr.

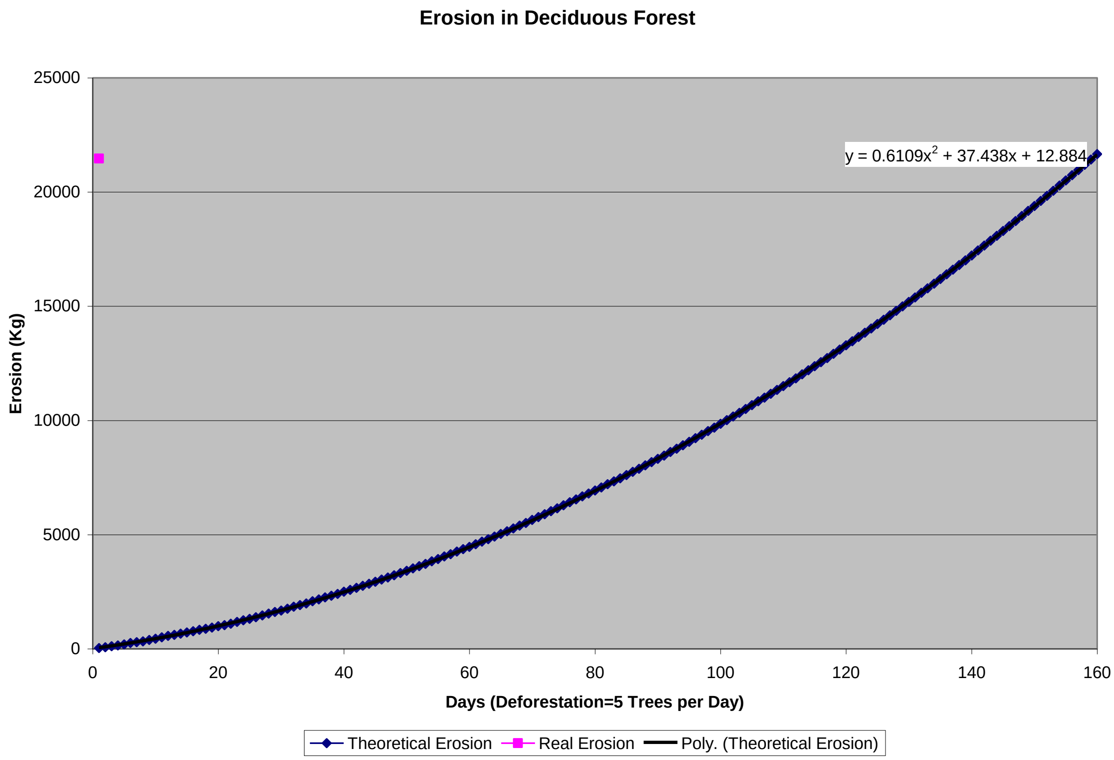

Figure 5: Erosion in Deciduous Forest. 795 Trees, 1,000,000 meters^2, and representative tree is white oak. K=0.35, M=0.35, Rain Probability=0.142857 Rain Intensity=2 cm/hr.

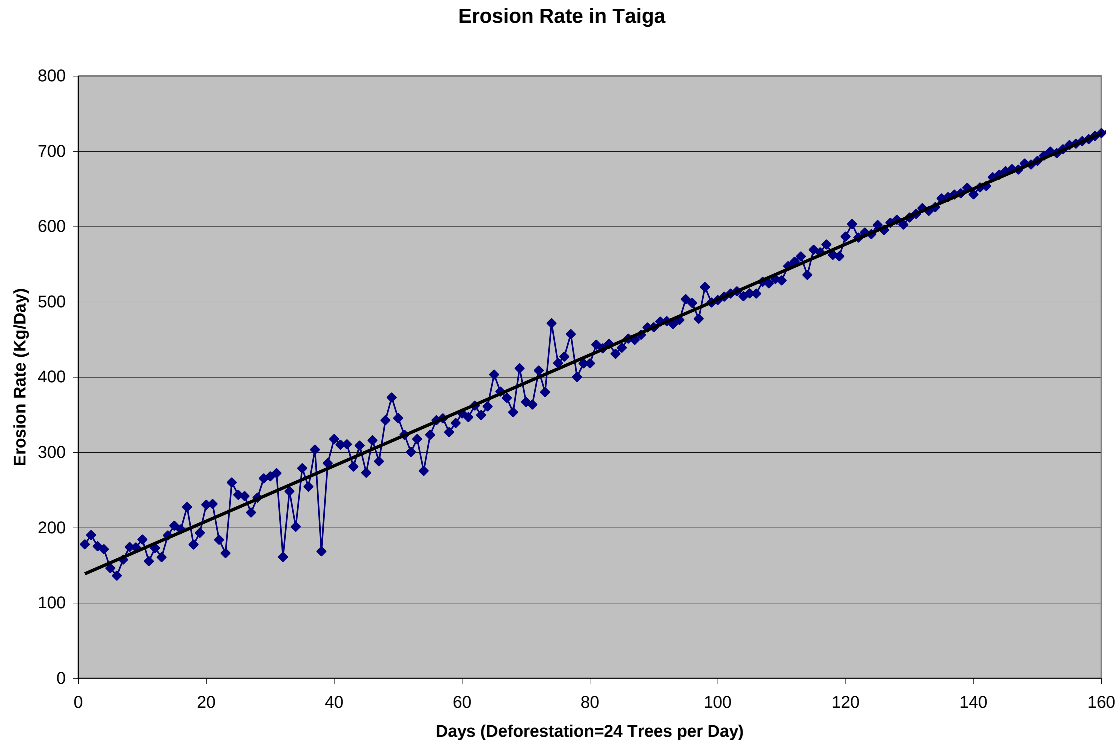

Figure 6: Erosion Rate in Taiga. Trees 3875, 1,000,000 meters^2, and representative tree is Norway spruce. K=0.35, M=0.35, Rain Probability=0.1 Rain Intensity=1 cm/hr.

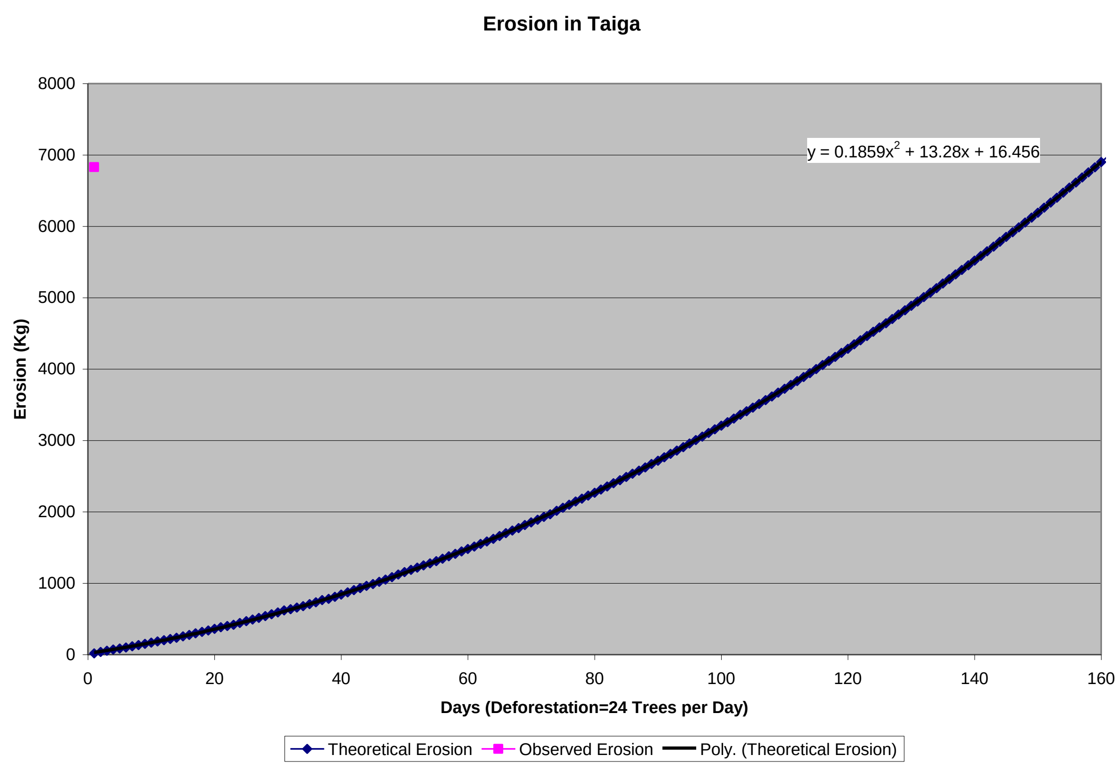

Figure 7: Erosion in Taiga. Trees 3875, 1,000,000 meters^2, and representative tree is Norway spruce. K=0.35, M=0.35, Rain Probability=0.1 Rain Intensity=1 cm/hr.

- Figures 3,5,7 were the plots of the definite integrals from 0 to x of figures 2,4,6 multiplied by the rain probability, so it is expected for the best fit line of the 3,5,7 to be quadratic if 2,4,6 have linear best fit lines, which they do.
- The observed erosion has less than 5% variance when compared to theoretical erosion. That control shows that the random number generator is functional.
- Similarly, this data is also reasonable since the increase in days with constant deforestation results in an overall increase in erosion and erosion rates.
- The rainfall was lowest in the taiga; as a result, it had lower erosion rates and lower overall erosion, which corresponds to real world statistics (Relative Erosion Rates [RER]).
- Coniferous trees take up less root space than deciduous trees.
- The rainfall was highest in the deciduous forest, implying that rain intensity is the most important variable in deciding erosion and erosion rates since the only difference in deciduous and tropical rainforest was the different rain probabilities and rain intensities. Even with less than half the rain probability of the tropical rain forest, it produced an erosion rate 8 times as large with double the rain intensity.
- The maximum erosion of bare rainforest is 32 ton/hectare/year and the deciduous forest maximum is 30-40 ton/hectare/year. Assuming the maximum rate from data and multiplying to account for years, rain probability, and hectares, the data shows 9.5 ton/hectare/year and 5.0 ton/hectare/year respectively. This implies that more factors are involved in long-term erosion.
- The minimum erosion rate of deciduous forest is 0.5-0.7 ton/hectare/year and tropical rainforests have 1 kg/hectare/year. Assuming the minimum rate from data and multiplying to account for years, rain probability, and hectares, the data shows 1.3 ton/hectare/year and 0.8833 ton/hectare/year. This implies that more factors are involved in halting erosion than mentioned.
- The time it takes for half the dirt to erode over the 160 day period of deforestation is around 105-110 days, implying the inputs are more or less accurate.

Conclusion:

- Rain intensity is most important to erosion.
- This model produces reasonable numbers that will fit most data towards the center of the graph (partial deforestation) since the predicted minimum was greater than the actual minimum and the predicted maximum is less than the actual maximum.

Future Work

- Snow makes up a majority of the precipitation of the taiga and a large percentage of the precipitation of the deciduous forest. Snow causes erosion when it melts and not when it hits the surface of the land.
- Extend the runtime to beyond total deforestation because soil degradation is worst in the years following clear cutting.
- Include soil nutrient leaching which can accelerate deforestation rates.
- Include the topological factors on the forest floor and the effect of leaves in slowing rain and thus lowering displacement and erosion. A cell based model is a possible alternative where each cell is given a height and it would remove all the generalizations.
- Introduce random rainfall intensity and duration; currently those are fixed user inputs. This is more realistic and can better model seasonal or monsoonal changes especially in tropical regions.

Source Code:

```
package program2;
/*
 * Main.java
 *
 * Created on August 17, 2006, 4:34 PM
 *
 * To change this template, choose Tools | Template Manager
 * and open the template in the editor.
 */
//import com.sun.org.apache.xerces.internal.impl.xpath.regex.Match;
import java.io.*;
import javax.swing.JOptionPane;
import java.awt.*;
import javax.swing.*;

public class program2 extends JPanel{
    public static void main( String args[] ){
        program2 draw = new program2();
        draw.setOpaque(true);
        JFrame f = new JFrame();
        f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        f.setContentPane(draw);
        f.setSize(1,1);
        f.setLocation(0, 0);
        f.setVisible(true);
        int x, y, timestep1 ;
        double result1, result2, z, r, d, result3, w, result4, result5, result6, p0, p1, p2, result7,
wi, ro, sf, m0, dbh, c0, m1, c1, af, slopeg, result4e, result5e, result6e, ef, denm, denv,
slopelen, sm, rp, viscos, v, sbd;
        String xVal, yVal, zVal, rVal, wVal, p0Val, p1Val, p2Val, wiVal, roVal, sfVal,
m0Val, dbhVal, c0Val, m1Val, c1Val, slopegVal, slopelenVal, smVal, rpVal,
r_initialVal, sbdVal;
        double r_initial;
        String sCont = JOptionPane.showInputDialog("[Enter] to use defaults. Input any
value to specify defaults.");
        if(sCont.equals("")) {}{
        xVal = JOptionPane.showInputDialog(
                "Enter Number of Trees:" );
        yVal = JOptionPane.showInputDialog(
                "Enter Deforestation (Number of trees being destroyed):" );
        zVal = JOptionPane.showInputDialog(
                "Total area (in m^2):" );
        r_initialVal = JOptionPane.showInputDialog(
                "How much in hours does it rain when it rains?:" );
        wVal = JOptionPane.showInputDialog(
```

```
        "How much per hour per unit of area does it rain? (in centimeters):");
    p0Val = JOptionPane.showInputDialog(
        "What percentage of the volume is made of large raindrops?:");
    p1Val = JOptionPane.showInputDialog(
        "What percentage of the volume is made of normal raindrops?:");
    p2Val = JOptionPane.showInputDialog(
        "What percentage of the volume is made of small raindrops?:");
    wiVal = JOptionPane.showInputDialog(
        "What is the speed of the wind in kph?:");
    roVal = JOptionPane.showInputDialog(
        "What is the ratio of the root diameter to the crown diameter?:");
    sfVal = JOptionPane.showInputDialog(
        "What is the soil factor?:" );
    m0Val = JOptionPane.showInputDialog(
        "What is the slope of the linear correlation function between diameter at breast
height in cm and crown spread in m^2?:");
    dbhVal = JOptionPane.showInputDialog(
        "What is the maximum diameter at breast height in cm?:");
    c0Val = JOptionPane.showInputDialog(
        "What is constant value of the linear correlation function?:");
    m1Val = JOptionPane.showInputDialog(
        "If tree maturity is measured from 0 to 1, m1=Maximum Maturity-Minimum
Maturity?:");
    c1Val = JOptionPane.showInputDialog(
        "c1=Minimum Maturity:");
    slopegVal = JOptionPane.showInputDialog(
        "What is the average angle of the ground to the horizontal in radians?:");
    slopelenVal = JOptionPane.showInputDialog(
        "What is the average length of the slope in consideration in m?:");
    smVal = JOptionPane.showInputDialog(
        "What is soil moisture constant, where 1 is fully moist?:");
    rpVal = JOptionPane.showInputDialog(
        "What is the probablility that it will rain?:");
    sbdVal = JOptionPane.showInputDialog(
        "What is the density of the soil?");}


    if (p2Val.equals("")) p2Val = "10";
    if (xVal.equals("")) xVal = "3875";
    if (yVal.equals("")) yVal = "24";
    if (zVal.equals("")) zVal = "1e6";
    if (r_initialVal.equals("")) r_initialVal = "1";
    if (wVal.equals("")) wVal = "1.5";
    if (p0Val.equals("")) p0Val = "10";
    if (p1Val.equals("")) p1Val = "80";
    if (wiVal.equals("")) wiVal = "15";
```

```
if (roVal.equals("")) roVal = "3";
if (sfVal.equals("")) sfVal = "0.35";
if (m0Val.equals("")) m0Val = "0.7746";
if (dbhVal.equals("")) dbhVal = "43.18";
if (c0Val.equals("")) c0Val = "4.7894";
if (m1Val.equals("")) m1Val = "0.3";
if (c1Val.equals("")) c1Val = "0.7";
if (slopegVal.equals("")) slopegVal = "0.0872664625997164788461845384244431";
if (slopelenVal.equals("")) slopelenVal = "1e3";
if (smVal.equals("")) smVal = "0.35";
if (rpVal.equals("")) rpVal = "0.1";
if (sbdVal.equals("")) sbdVal = "1.2";


timestep1 = 1;
double pi = 3.14159265358979323;
double viscosn = 0.001;
double viscosd = 1 * timestep1;
viscos = viscosn / viscosd;

x = Integer.parseInt( xVal );
y = Integer.parseInt( yVal );
z = Double.parseDouble( zVal );
r_initial = Double.parseDouble( r_initialVal );
w = Double.parseDouble( wVal );
p0 = Double.parseDouble( p0Val );
p1 = Double.parseDouble( p1Val );
p2 = Double.parseDouble( p2Val );
wi = Double.parseDouble( wiVal );
ro = Double.parseDouble( roVal );
sf = Double.parseDouble( sfVal );
m0 = Double.parseDouble( m0Val );
dbh = Double.parseDouble( dbhVal );
c0 = Double.parseDouble( c0Val );
m1 = Double.parseDouble( m1Val );
c1 = Double.parseDouble( c1Val );
slopeg = Double.parseDouble( slopegVal );
slopelen = Double.parseDouble ( slopelenVal );
sm = Double.parseDouble( smVal );
rp = Double.parseDouble( rpVal );
sbd = Double.parseDouble( sbdVal );
if (Math.random() <= rp)
   r=r_initial;
else
   r=0;
double den = 1;
```

```
        denm = r * w / 3600;
        denv = r * w / 3600;
        double dTotaleftrue = 0;
        int iDay = 0;




         //water has viscosity of 0.001 Pa*sec at 20C

        result1 = x * (m0 * dbh * (m1 * Math.random() + c1) - c0) * ro * ro;
        af = (m1 * Math.random() + c1);
        if(y > 0) {double dTotalef = 0;
           while (result1 >= 0) {

                iDay ++;
                //System.out.println("Day: " + iDay);

           // if (timestep1 > 0) {
           // while (slopelen > 0)
//                JOptionPane.showMessageDialog( null,
//                        "The amount of dirt held down (in m^2) is " + result1 );

                result2 = z - result1;
//                JOptionPane.showMessageDialog( null,
//                        "The erodeable area (in m^2) is " + result2 );

                d = result2 * w;
                result3 = d * r;
                // JOptionPane.showMessageDialog( null,
                //          "The amount of dirt removed in grams is" + result3 );
                result4 = result3 * p0/100 * 4/3 * pi * (0.508/2) * (0.508/2) * (0.508/2) *
((32.18688) * (32.18688) + (wi) * (wi)) / 2 * 5 / 18;
                result5 = result3 * p1/100 * 4/3 * pi * (0.2032/2) * (0.2032/2) * (0.2032/2) *
((22.530816) * (22.530816) + (wi) * (wi)) / 2 * 5 / 18;
                result6 = result3 * p2/100 * 4/3 * pi * (0.02032/2) * (0.02032/2) * (0.02032/2)
* ((2.5749504) * (2.5749504) + (wi) * (wi)) / 2 * 5 / 18;
                result4e = Math.sin( Math.atan( 32.185888 / wi ) - slopeg) * result4;
                result5e = Math.sin( Math.atan( 22.530816 / wi ) - slopeg) * result5;
                result6e = Math.sin( Math.atan( 2.5749504 / wi ) - slopeg) * result6;
                result7 = sf * (result4e + result5e + result6e);
                //JOptionPane.showMessageDialog( null,
                //        "The real amount of dirt removed in grams is " + result7);
                float fDirtEroded = 0;
                double dTempSlopeLen = slopelen / 2;
                double L = result2 / (2 * slopelen);
```

```
            double dTemp = (w / 3600 * .01);
            if (r==0){
               dTotaleftrue = 0;
               dTempSlopeLen = 0;
               System.out.println("Day:\t" + iDay + "\tEroded Today:\t 0");
//                  JOptionPane.showMessageDialog( null,
//                      "The dirt eroded in kg " + dTotaleftrue);
            }
            else {
            while (dTempSlopeLen > 0){
            v = den * 9.8 * Math.pow(dTemp,2) * Math.sin( slopeg ) * 1000 / (3 * viscos );
            //System.out.println("v=" + v +"den=" + den +"  dTemp="+ dTemp +"
slopeg="+ slopeg + "  viscosn=" + viscosn +"viscosd=" + viscosd +"timestep1=" +
timestep1);
            ef = (v / dTemp) * (3 * viscos) * L  * v * sm * sf;
            //System.out.println("ef=" + ef);
            viscosn += (ef / result2 + dTemp);
            timestep1 += 1;
            viscosd += (v * timestep1 + timestep1 * viscosd - viscosd + dTemp);
            viscos = viscosn / viscosd;
            denm += (ef / result2 * 1000 + dTemp);
            denv += (ef / result2 * 1000 * (1 / sbd) + dTemp);
            den = denm / denv;
            dTempSlopeLen -= v;

            dTotalef += ef;
            }

            dTotaleftrue = dTotalef * r * 3599 * (9.8 * Math.pow(dTemp,2) * Math.sin(
slopeg ) * 1000 / (3 * 0.001 )) / (slopelen) + dTotalef;
            System.out.println("Day:\t" + iDay + "\tEroded Today:\t" + dTotaleftrue);

//              JOptionPane.showMessageDialog( null,
//                      "The dirt eroded in kg " + dTotaleftrue);

            }
         // }
            if(x > 0) {
               x-=y;
            } else {
               x=-1;
            }
            result1 = x * (m0 * dbh * af - c0) * ro * ro;
         timestep1 = 1;
         pi = 3.14159265358979323;
         viscosn = 0.001;
```

```
            viscosd = 1 * timestep1;
            viscos = viscosn / viscosd;
            den = 1;
            denm = r * w / 3600;
            denv = r * w / 3600;
            dTotaleftrue = 0;
            dTotalef = 0;
        if (Math.random() < rp)
            r=r_initial;
            else
            r=0;

            }
        } else {


           JOptionPane.showMessageDialog( null,
                   "The amount of dirt held down (in m^2) is " + result1 );

              result2 = z - result1;
              JOptionPane.showMessageDialog( null,
                   "The erodeable area (in m^2) is " + result2 );

              d = result2 * w;
              result3 = d * r;
              // JOptionPane.showMessageDialog( null,
              //         "The amount of dirt removed in grams is" + result3 );
              result4 = result3 * p0/100 * 4/3 * pi * (0.508/2) * (0.508/2) * (0.508/2) *
((32.18688) * (32.18688) + (wi) * (wi)) / 2 * 5 / 18;
              result5 = result3 * p1/100 * 4/3 * pi * (0.2032/2) * (0.2032/2) * (0.2032/2) *
((22.530816) * (22.530816) + (wi) * (wi)) / 2 * 5 / 18;
              result6 = result3 * p2/100 * 4/3 * pi * (0.02032/2) * (0.02032/2) * (0.02032/2)
* ((2.5749504) * (2.5749504) + (wi) * (wi)) / 2 * 5 / 18;
              result4e = Math.sin( Math.atan( 32.185888 / wi ) - slopeg) * result4;
              result5e = Math.sin( Math.atan( 22.530816 / wi ) - slopeg) * result5;
              result6e = Math.sin( Math.atan( 2.5749504 / wi ) - slopeg) * result6;
              result7 = sf * (result4e + result5e + result6e);
              //JOptionPane.showMessageDialog( null,
              //       "The real amount of dirt removed in grams is " + result7);
              float fDirtEroded = 0;
              double dTotalef = 0;
              double dTempSlopeLen = slopelen / 2;
              double L = (result2 + result7 * sm) / (2 * slopelen);
              double dTemp = (w / 3600 * .01);
              if (r==0){
                 dTotaleftrue = 0;
```

```
                dTempSlopeLen = 0;
                JOptionPane.showMessageDialog( null,
                    "The dirt eroded in kg " + dTotaleftrue);
            }
            else {
            while (dTempSlopeLen > 0){
            v = den * 9.8 * Math.pow(dTemp,2) * Math.sin( slopeg ) * 1000 / (3 * viscos );
            //System.out.println("v=" + v +"den=" + den +"  dTemp="+ dTemp +"
slopeg="+ slopeg + "  viscosn=" + viscosn +"viscosd=" + viscosd +"timestep1=" +
timestep1);
            ef = (v / dTemp) * (3 * viscos) * L  * v * sm * sf;
            //System.out.println("ef=" + ef);
            viscosn += (ef / result2 + dTemp);
            timestep1 += 1;
            viscosd += (v * timestep1 + timestep1 * viscosd - viscosd + dTemp);
            viscos = viscosn / viscosd;
            denm += (ef / result2 * 1000 + dTemp);
            denv += (ef / result2 * 1000 * (1 / sbd) + dTemp);
            den = denm / denv;
            dTempSlopeLen -= v;
            dTotalef += ef;
            }

            dTotaleftrue = dTotalef * r * 3599 * (9.8 * Math.pow(dTemp,2) * Math.sin(
slopeg ) * 1000 / (3 * 0.001 )) / (slopelen) + dTotalef;
            System.out.println("dTotaleftrue=" + dTotaleftrue);
            JOptionPane.showMessageDialog( null,
                    "The dirt eroded in kg " + dTotaleftrue);

            }


        System.exit( 0 );
    }
}
}
//File fileName = "H:\\Data.txt";
//1 mature tree holds down 10507086.286565712 cm^2 of dirt.
//If it rains x cubic centimeters per centimeter^2 per hour, and it rains 1 hour, the amount
of dirt removed in grams x*result2.
//public class GraphThingyNow extends Applet {
//public void init()
//{
//  setBackground(Color.white);
//}
// public void paint(Graphics g){
```

References

1. www.ipm.iastate.edu/ipm/icm/
2. md1.csa.com/partners/viewrecord.php?requester=gs&collection=ENV&recid=4226
3. www.seafriends.org.nz/enviro/soil/
4. www.grow.arizona.edu/Grow--GrowResources.php?ResourceId=146
5. mepas.pnl.gov/mpeas/formulations/source_term/5_0/5_32/5_32.html
6. www.ext.colostate.edu/PUBS/Garden/02926.html
7. www.forestry.iastate.edu
8. www.itc.nl/ilwis/applications/application12.asp
9. www.ohioline.osu.edu/sc195/028.html
10. The Reneco Flash Issue Number 2 December 1999
11. www.worldwildlife.org/wildworld/profiles/terrestial/pa/pa1209_full.html
12. www.nps.gov/phso/sp/eawoch03.pdf
13. www.blueplanetbiomes.org
14. www.mortonarb.org
15. www.cocori.com
16. Bird, R.B., Stewart, W.E., and Lightfoot, E.N. Transport Phenomena, 2nd Edition, John Wiley and Sons Inc., New York, 2002.
17. www.weather.com (Brasalia, Rio de Janiero, New York City, Boston, Paris)