Chaotic Attractor in Tumor Growth and Decay

A Differential Equation Model

Wen-sau Yim & Michael Harney

12/10/2011

Abstract

Cancer growth and decay can be modeled as a system of chaotic nonlinear differential equations. The system is based on a reaction-diffusion cancer growth model, expressed by:

$$\frac{\partial n}{\partial t} = d_n \nabla^2 - \rho \nabla \cdot (n \nabla f)$$

$$\frac{\partial f}{\partial t} = \alpha \eta (m - f)$$

$$\frac{\partial m}{\partial t} = d_m \nabla^2 m + \kappa n - \sigma m$$

$$\frac{\partial c}{\partial t} = d_c \nabla^2 c + \nu f - \omega n - \phi c$$

The above model will be simplified by Ivancevicet. al (eqs. 12 – 15) so as to allow useful parameters to be applied for simulation and then recreated in C/Java, to directly compare with its predecessor project created by Ivancevicet. al (simulated in Wolfram CDF and Mathematica).

Varying input parameters of glucose and oxygen levels will be examined to determine chaos behavior. Ultimately, the result is the development of a bifurcation graph to characterize chaotic behavior. Based on this result, the model parameters will be optimized in order to minimize chaos in the system.

Methodology

From the reaction-diffusion cancer growth model above, a simplified non-dimensional non-spatial derivative model was found.This model was further modified by adding four additional parameters (α, β, γ, δ) which represents tumor cell volume, glucose level, number of tumor cells, and diffusion saturation level from the surface, respectively. The starting point of the generation C/Java simulations is expressed by:

$$\dot{n} = 0$$

$$\dot{f} = \alpha \eta (m - f)$$

$$\dot{m} = \beta \kappa n + f(\gamma - c) - m$$

$$\dot{c} = \nu f m - \omega n - \delta \phi c$$

Calculations can be found in Appendix A. For a unit dimensionless time change, the following was solved for f_{new}, $\mathrm{m_{new}}$, and $\mathrm{c_{new}}$:

$$f_{new} = (1 - \alpha\eta) f_{old} + \alpha \eta m_{old}$$

$$m_{new} = \beta \kappa n + f_{old}(\gamma - c)$$

$$c_{new} = (1 - \delta\phi)c_{old} + \nu f_{old} m_{old} - \omega n$$

The methodology in finding a non-chaotic system is by analyzing the set of constant parameters that describe the system. The constants hypothesized to exhibit chaotic behavior include: $\alpha, \beta, \gamma, \delta, \eta, \beta, \kappa, \gamma, \nu, \omega, \delta, \phi$, where α = tumor cell volume, β = glucose level, γ = number of tumor cells, δ = diffusion saturation, and $\eta, \beta, \kappa, \gamma, \nu, \omega, \delta, \phi$ are non-dimensional constants.

Documentation

Translated into a high-level programming pseudo-code:

```
// Set initial conditions
n = 50; // tumor cell density in the simulation
m = 0; // matrix-degradative enzyme concentration
f = 0; // matrix-metalloproteinases concentration
c = 0; // Oxygen concentration

// Start the simulation loop for N loops
Start Loop
       f[i+1] = (1-alpha * eta)*f[i] + alpha * eta* m[i];
       m[i+1] = beta * kappa * n + f[i]*(gamma - c[i])
       c[i+1] = (1 - delta * phi)*c[i] + nu * f[i] * m[i] - omega * n
End Loop
```

Sample C and Java source codes are provided in Appendix B.

Software

Three programming software packages were utilized:

- The C simulation was written and compiled using Pelles C, version 6.50 (http://www.smorgasbordet.com/pellesc/).
- The Java simulation was written and compiled with Java SE Version 6 Update 27 and Java SE Development Kit (JDK) 6, provided by Oracle (http://www.oracle.com/technetwork/java/javase/downloads/index.html).
- Wolfram Alpha CDF (http://www.wolframalpha.com/cdf-experiment/) and Wolfram Alpha Mathematica

Summary

Case 1: Gamma = Delta = 0

As it has been found from our C and Java simulation that decreasing Gamma and Delta increases bifurcations for an increase in tumor cell density, we start the CDF simulation with Gamma and Delta set to 0 and we use the m,f phase plot results in Figure 1 for comparison.

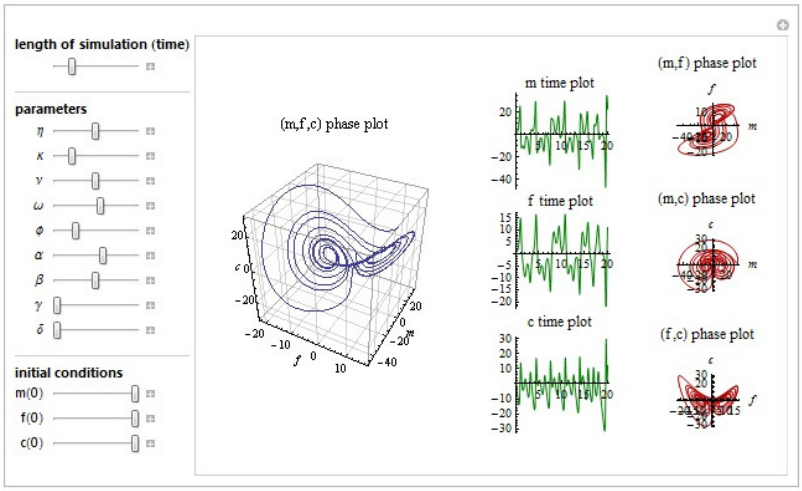

Figure 1: Case 1, Gamma = Delta = 0

Case 2: Gamma = Delta = 0, decreased oxygen

By decreasing oxygen (variable c), we find that the average m,f phase plot has a lower trajectory density than in Figure 1.This would be expected – decreasing oxygen slows growth and moves trajectories away from the attractor. Note that the two trajectory points in Figure 1 have decreased to one trajectory in Figure 2. Also, the (f,c) phase plot shows reduced activity, which is to be expected as *c* is the Oxygen concentration variable.

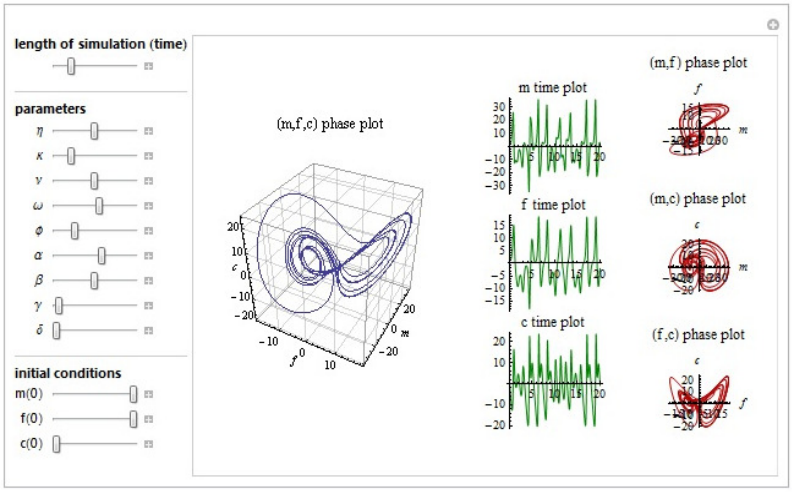

Figure 2: Case 2, Gamma = Delta = 0 with decreased oxygen levels

Case 3: Gamma = Delta = 0, decreased oxygen, decreased glucose

Starting with Case 2 and decreasing glucose (variable beta), we find that the second trajectory point returns in the m,f phase plot of Figure 3, showing an increase in attractor stability.

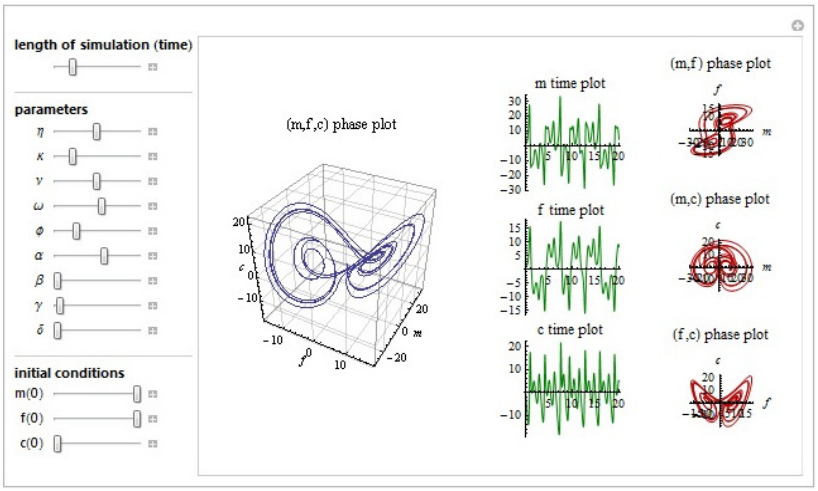

Figure 3: Case 3, Gamma = Delta = 0 with decreased oxygen and glucose

Case 4: Gamma = Delta = 0, decreased oxygen, increased glucose

Starting with Case 3 and increasing glucose significantly, we find that the lower left trajectory point in Figure 4 of the m,f plot is decreasing in size and density. This may indicate that the attractor is becoming less stable. The f,c phase plot also shows reduced activity, similar to when Oxygen concentration was initially decreased in Case 2.

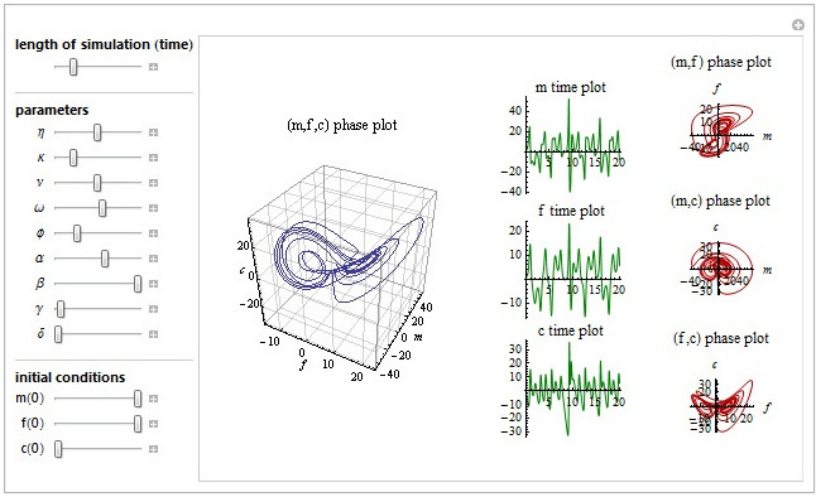

Figure 4: Case 4, Gamma = Delta = 0 with decreased oxygen and increased glucose

Case 5: Glucose maximized

For comparison to the above simulations where Gamma and Delta were set equal to 0, we restore Gamma, Delta and the Oxygen concentration to significant values and we also increase glucose. The results show a robust attractor with two well defined trajectory points in the m,f plot of Figure 5. This is typical of a textbook definition of tumor growth – high Oxygen and glucose concentration yield strong tumor growth.

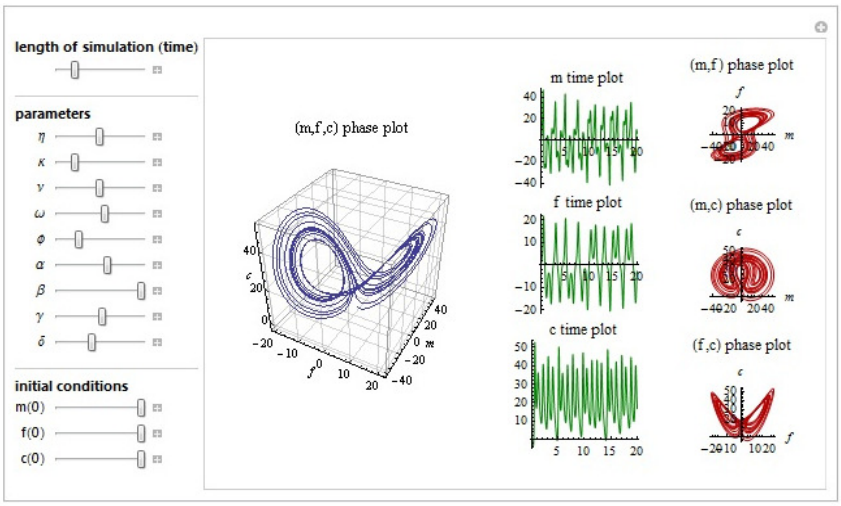

Figure 5: Case 5, maximized glucose

Case 6: Glucose minimized

From Case 5 settings, we decrease glucose and find that the trajectory points in the m,f plot of Figure 5 shows some reduction (as expected from a standard cell growth model) but not as significant a reduction as shown in Case 3, where Gamma and Delta where set to 0 and glucose was increased.

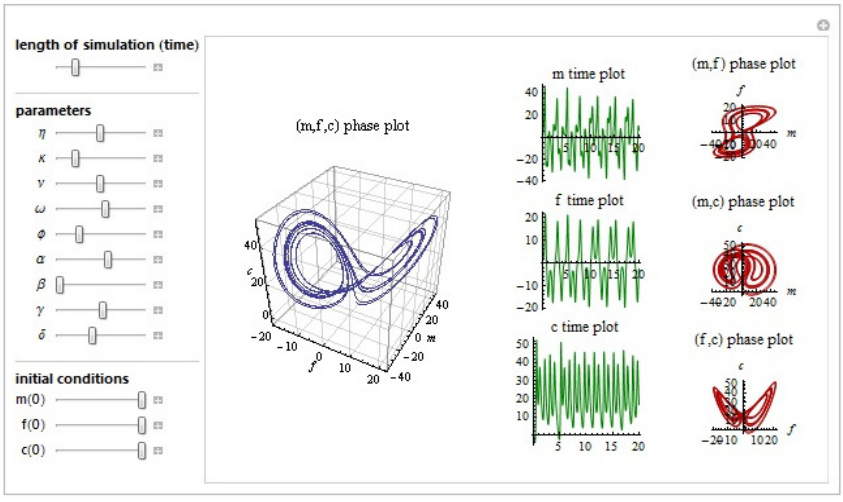

Figure 6: Case 6, minimized glucose

Validation

To validate the C/Java/CDF model used in this study, the 3D chaotic attractor generated by Ivancevic et. al. was recreated. The original attractor taken in Figure 1 from Ivancevic is shown in Figure 7.

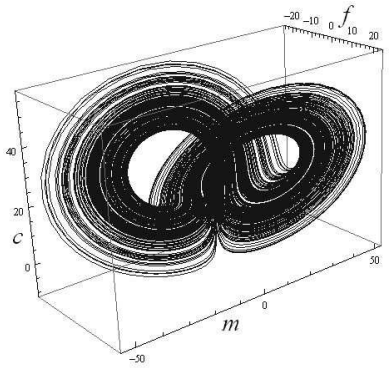

Figure 7: 3D Lorenz-like chaotic attractor from Ivancevic

By inserting the original conditions in the C/Java/CDF model, the following 3D model was generated, illustrated in Figure 8.

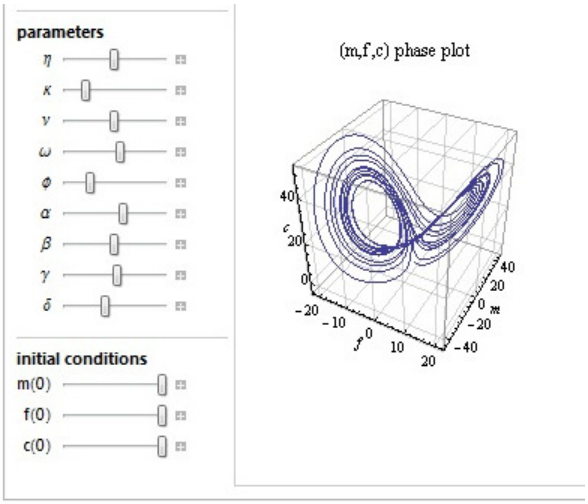

Figure 8: Validation (m,f,c) Phase Plot

By manipulating the values of the constants of gamma and delta (in the C/Java source code found in Appendix B), which represent the number of tumor cells and the diffusion from the surface respectively, the system was found to produce increasing bifurcations as is shown in Figure 9, which shows bifurcation levels for matrix-degradative enzyme concentration (MDE), matrix–metalloproteinases concentration, and Oxygen concentration (which hypothetically changes due to rapid and then slower growth rates). The system modeled with the C and Java simulation produces results that approach the plots in Figure 1 of the CDF time domain simulations.

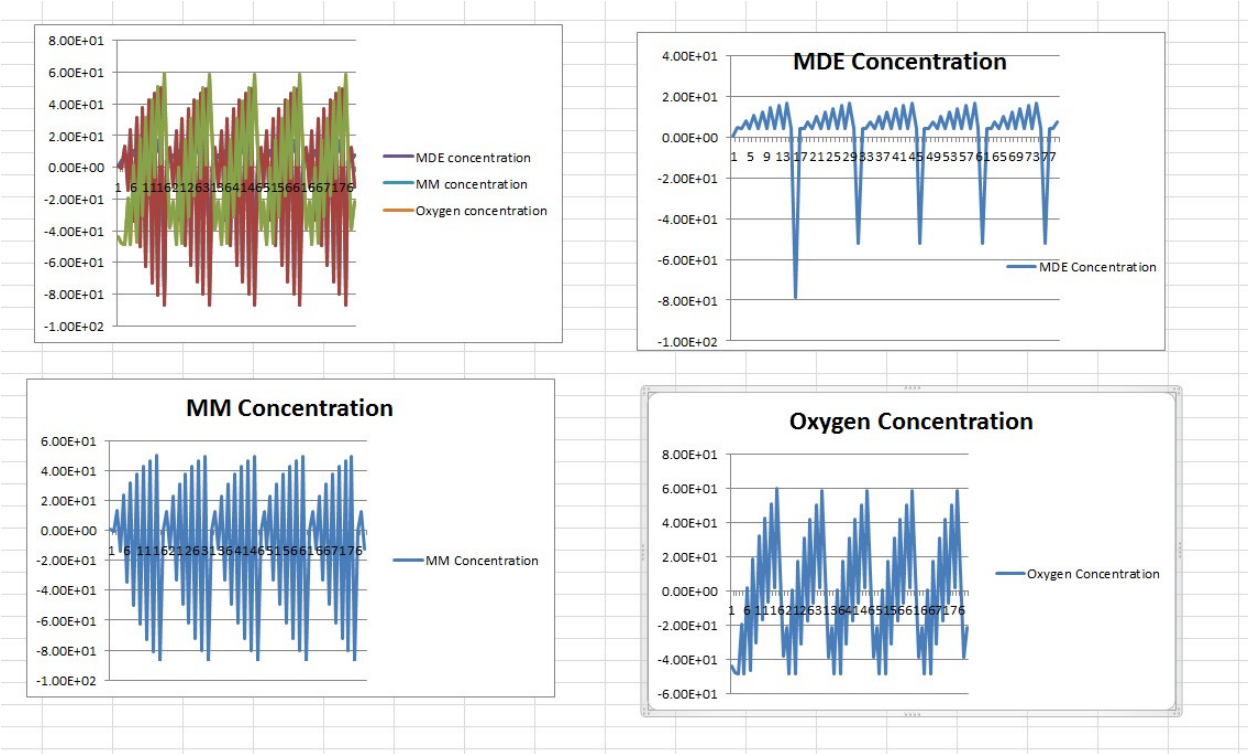

Figure 9 – C/Java Simulation Results with gamma = 0.265 (low), delta = 0.4 (low) and beta = 0.05 (low)

Conclusions

Bar-Yam's definition of a chaotic system is a deterministic system that is difficult to predict. Chaotic systems are often associated with the Butterfly Effect or "sensitivity to initial conditions." Both characteristics stem from the behavior of chaotic systems to evolve into unpredictable paths. This study aimed to characterize the chaotic behavior in the case of cancer tumor growth and decay. The terms "deterministic" and "characterize" may seem out-of-place when referring to chaotic systems, especially for tumor growth. Using a simplified non-dimensional non-spatial differential equation model, a stable (non-chaotic) system was found by manipulating the system parameters: , δ, β .

A bifurcating system that approaches a chaotic system was found by defining decreased values of gamma (number of tumor cells) and delta (diffusion constant): $\gamma = 0.265, \delta = 0.40$. In this modified system, a key finding was that the increased glucose in the presence of lowered oxygen levels decreased the tumor growth. Biologically speaking, this deviates from the current literature that argues that increased in glucose levels leads to increase of tumor growth, which is shown for comparison in Case 5 (Figure 5) and Case 6 (Figure 6). The decrease in the number of tumor cells and the decrease in the diffusion constant are expected to decrease tumor growth, but it would still be assumed that an increase in glucose would increase tumor growth and there is evidence from Case 1 – 4 examples that this is not the case. Our C/Java simulation also provides evidence of an increase in activity as glucose is decreased when Gamma and Delta have low values.

References

1. Ivancevic, T. T., Bottema M. J., and Jain, L. C., "A Theoretical Model of Chaotic Attractor in Tumor Growth and Metastasis," arXiv: 0807.4272 in Cornell University Library's arXiv.org.
2. Sole, R. and Goodwin, B.. Signs of Life: How Complexity Pervades Biology. Basic Books, New York, NY. 2000.
3. Bar-Yam, Y. "Concepts: Chaos." New England Complex Systems Institute. 2011. http://www.necsi.edu/guide/concepts/chaos.html
4. Guiot, C., Degiorgis, P. G., Delsanto, P. P., Gabriel, P., and Deisboeck, T.S., Does Tumor Growth Follow a "Universal Law?". http://arxiv.org/ftp/physics/papers/0303/0303050.pdf

Appendix A: Calculations for DE Model

The derivations for the C/Java model are provided below:

$$\int \frac{dn}{dt} dt = 0$$

$$\dot{f} = \frac{f_{new} - f_{old}}{t} = \alpha\eta(m - f)$$

$$\dot{m} = \frac{m_{new} - m_{old}}{t} = \beta\kappa n + f(\gamma - c) - m$$

$$\dot{c} = \frac{c_{new} - c_{old}}{t} = \nu f m - \omega n - \delta\phi c$$

For a unit time change of 1, solve for f_{new}, m_{new}, and c_{new}:

$$f_{new} = (1 - \alpha\eta) f_{old} + \alpha\eta m_{old}$$

$$m_{new} = \beta\kappa n + f_{old}(\gamma - c)$$

$$c_{new} = (1 - \delta\phi) c_{old} + \nu f_{old} m_{old} - \omega n$$

Appendix B: Sample source code

C Programming Language

```
#include <math.h>
#include <stdio.h>
#include <stdlib.h>

#define ITERATIONS      900

int main ()
{

inti;
FILE *F1;

floatn_old = 0.50;  //  tumor cell density 50
floatf_old = 0.0;  // matrix-metalloproteinases concentration
floatf_new = 0.0;
floatm_old = 0.0;  // matrix-degradative enzyme concentration
floatm_new = 0.0;
floatc_old = 0.0;  // Oxygen concentration
floatc_new = 0.0;

float alpha = 0.06; // tumor cell volume
float beta = 0.05;  // glucose level
float gamma = 0.265; // number of tumor cells 26.5
float delta = 0.40; // diffusion from the surface 40

floatdn = 0.0005;
float dm = 0.0005;
float dc = 0.5;
float rho = 0.01;
float eta = 0.50; //was 50
float kappa = 1.0;
float sigma = 0;
float nu = 0.5;
float omega = 0.57;
float phi = 0.025;

F1 = fopen("chaotumor.txt","w");

for (i = 0; i< ITERATIONS; i++)
      {

            f_new = alpha*eta*(m_old - f_old) + f_old;

            m_new = beta*kappa*n_old - f_old*c_old + gamma*f_old;

            c_new = nu*f_old*m_old - omega*n_old - delta*phi*c_old + c_old;


            f_old = f_new;
            m_old = m_new;
```

```
            c_old = c_new;

            fprintf (F1,"%f",m_new);
            fprintf (F1," %f\n",f_new);
            fprintf (F1," %f\n",c_new);

      }

fclose(F1);

}
```

Java Programming Language

```
importjava.util.Scanner;
import java.io.*;
importjava.lang.Math.*;
importjava.util.Random;
importjava.util.*;


public class ChaosTumor
{

     // initialize system parameters
static double alpha = 0.06; //tumor cell volume
static double beta = 0.05; // glucose level
static double gamma = 0.265; // number of tumor cells, 26.5
static double delta = 0.40; // diffusion from surface, 40

     // initalize system constants
static double n = 50;
static double dn = 0.0005;
static double dm = 0.0005;
static double dc = 0.5;
static double rho = 0.01;
static double eta = 50;
static double kappa = 1.0;
static double sigma = 0;
static double nu = 0.5;
static double omega = 0.57;
static double phi = 0.025;


public static void main (String[] args)
   {

     // initialize parameters
double f; // matrix-metalloproteinases concentration
double m; // matrix-degradative enzyme concentration
double c; // oxygen concentration


     // ask user input for growth, capacity, and initial population
```

```
     Scanner scan = new Scanner (System.in);
System.out.println("Chaos in Tumor Growth Dynamics Study");
System.out.println("****************************");
System.out.print("Enter MM concentration, f: ");
     f = scan.nextDouble();
System.out.print("Enter MDE concentration, m: ");
     m = scan.nextDouble();
System.out.print("Enter the oxygen concentration, c: ");
     c = scan.nextDouble();
System.out.println("****************************");


    // open file
try
     {
      // open output file
        File outFile = new File("ChaosTumor.txt");
BufferedWriter writer = new BufferedWriter(new FileWriter(outFile));

        // print out the inital condition
      writer.write("Initial conditions :"+m+" "+f+" "+c);
System.out.println("Initial population: "+m+" "+f+" "+c);
      writer.newLine();

      double [] temp = new double[3];
      for(inti=1; i<51; i++)
      {
            temp = compute(m,f,c);
            f = temp[0];
            m = temp[1];
            c = temp[2];

            // remove negative values
            if(m<0)
                  m=0;
            if(f<0)
                  f=0;
            if(c<0)
                  c=0;

            writer.write("Iteration "+i+": "+m+" "+f+" "+c);
            writer.newLine();
            System.out.println("Iteration "+i+": "+m+" "+f+" "+c);
      }

      // close output file
writer.close();
     }
catch (IOException e)
     {
System.err.println(e);
System.exit(1);
     }

     // close file
```

```
   } // end main


   /**
   * method: compute(float m, float f, float c)
   * computes an array containing [m,f,c] at t+1
   * @param: float m, float f, float c
   * @precondition: none
   * @postcondition: none
   * @return: returns m, f, and c concentrations at t+1
   **/
public static double[] compute(double m, double f, double c)
   {
      double[] array = new double[3];
array[0] = (1-alpha*eta)*f + alpha*eta*m;
      array[1] = beta*kappa*n +(gamma-c)*f;
      array[2] = (1-delta*phi)*c + nu*f*m - omega*n;
return array;

   }

}
```