HARVEY MUDD
COLLEGE

Mathematics Clinic

Final Report for
Los Alamos National Laboratory

Modeling Vascular Tumor Growth

May 1, 2006

Team Members

Cris Cecka
Alan Davidson
Tiffany Head
Dana Mohamed (Project Manager)
Liam Robinson

Advisor

Prof. Lisette dePillis

Liaison

Dr. Yi Jiang

Abstract

We have enhanced a computational tumor model developed at the Los Alamos National Laboratory (LANL). We have modified portions of the model in order to accommodate the inclusion of *vasculature* (a blood-vessel system). Other modifications were made which allow chemotherapy treatments which are not cell cycle specific to be simulated. We have worked to increase the efficiency of the model code, reducing its computational memory requirements are less severe and allowing simulations to run more efficiently.

These improvements in accuracy and efficiency have allowed us to study the effects of chemotherapy treatments on tumor spheroid growth. We report the results of simulated experiments using two different dose levels, both within normal treatment levels for cyclophosphamide, which is a commonly-used non-cell cycle specific chemotherapy drug.

Contents

Acknowledgments

We would like to acknowledge the assistance and contributions of the following people:

Professor Lisette dePillis, Harvey Mudd College
Professor Michael Raugh, Harvey Mudd College
Dr. Yi Jiang, Los Alamos National Laboratory
Dr. James “Mac” Hyman, Los Alamos National Laboratory
Amy Bauer, Los Alamos National Laboratory
Claire Connelly, Harvey Mudd College
Barbara Schade, Harvey Mudd College

Chapter 1

Introduction

Cancer is the second leading cause of death among Americans and is responsible for one of every four deaths in the United States (ACS [2006]). Chemotherapy is one of the most commonly used methods of cancer treatment, but most forms of chemotherapy are extremely toxic and take a heavy toll on the health of the patient. This is because many standard forms of chemotherapy target and kill cells in the process of division, an approach that destroys healthy, non-cancerous cells as well as cancerous cells. Cancer patients undergoing chemotherapy suffer from hair-loss and compromised immune systems because both involve rapidly dividing cells. Still, the full effects of chemotherapy are not completely understood. By creating a better model of tumors, tumor growth, and the effects of chemotherapy upon such tumors, new knowledge about tumors may be uncovered to improve treatments for patients with cancer.

Our Mathematics Clinic team has been provided with a computer program developed by Dr. Yi Jiang and her team at Los Alamos National Laboratory (LANL) that simulates 3-dimensional spheroid tumor growth. LANL was established in Los Alamos, New Mexico in 1943 to conduct research for the Manhattan Project. LANL conducts research in many fields, including basic scientific research projects such as ours.

We have modified Dr. Jiang's computational model to include support for a blood-vessel structure, or *vasculature*. Our modified model simulates vascular tumor growth dynamics. Chemotherapy treatments using cyclophosphamide, a non-cell cycle specific anti-cancer drug, have also been simulated. Section 3.3 of this paper contains a summary of how the simulated cyclophosphamide chemotherapy treatment has been implemented.

The original LANL model simulates avascular tumor spheroid growth

in a laboratory setting. It is a multistage model that captures tumor behavior at the molecular level through chemical diffusion, at the subcellular level through protein expression, and at the cellular level through cellular division, growth, and death. This model does not contain vasculature. Numerous changes were required to include vasculature. Some of the changes were minor, while others were extensive. We also streamlined the existing code to reduce execution time.

In this paper, we detail the work completed over two academic semesters. In Chapter 2, we provide a description of the original avascular growth code provided by LANL. The first section describes the model of protein expression used to simulate a cell's passage through the cell cycle. The second section describes the Monte Carlo approach used to model cell movement and growth. The chapter concludes with a discussion of how diffusion of chemicals to and from the cells is modeled. Chapter 3 enumerates general code improvements as well as changes made to include vasculature and chemotherapy treatments. The results of these improvements are described in Chapter 4. Chapter 5 describes continuing and future work. Our conclusions are given in Chapter 6.

In order to better understand the nature of our project, and to clarify the relationship of this project with other research in the field, we have obtained and read many papers by researchers in the field. As a result, we offer our consideration of the importance of the project in terms of other past and current research.

The growth of tumors can be divided into three stages. According to Alarcon et al. [2005], these stages are described as avascular growth, angiogenesis, and vascular growth. During the avascular stage, there is no blood supply to the tumor, and the tumor reaches a maximum size limited by the amount of oxygen and nutrients the tumor can receive through its surface, Then, some of the tumor cells produce substances known as tumor angiogenic factors (TAFs). When TAFs diffuse to the surrounding vasculature of noncancerous tissue, angiogenesis occurs. In this stage, the vasculature grows toward and into the tumor. Once blood vessels have reached the tumor, the third stage begins: the tumor receives a vast amount of nutrients and can grow larger than was possible during the avascular growth stage. Furthermore, the vasculature now serves as an avenue for metastasis. Once a tumor enters the vascular growth stage, its potential lethality greatly increases (Alarcon et al. [2005]).

Cancer modeling has seen contributions scientists of many backgrounds. For this project, we focused primarily upon research dealing with vascularized tumors and less on papers about avascular tumor growth and an-

giogenesis. Tumor modelling has greatly increased in sophistication over recent decades. Initial models considered avascular tumors that grow exponentially, which is only a good approximation for initial tumor growth. Some current tumor models make use of cellular automata (CAs) which describe cell interactions in space and time. See, for example, de Pillis and Mallet [2006], Qi et al. [1993], and Ferreira et al. [2003].

Cancer modeling has advanced in two distinct ways. Some projects have aimed to produce a model that describes all three stages of tumor growth, including intricate details of the linkage between stages. One example of this is a model created by Alarcon et al. [2005]. Other projects have studied individual stages in great depth and considered variations in the tumor environment, as done by Alarcon et al. [2004]. Some models have taken into account extremely complex fluid dynamics in blood flow modeling, e.g. McDougall et al. [2004]. Our project has aimed to address the effectiveness and efficiency of chemotherapy, an area that can benefit greatly from further study.

Chapter 2

Model Overview

We summarize the model provided by our liaison at LANL, and highlight areas where we have extended or improved the program. In this paper, computational elements of the code are denoted by a distinctive font (e.g., CELLS) while real, biological elements will remain in plain text (e.g., cells). The model used a three-level approach to modeling spheroid development. On the subcellular level, chemical concentrations within and between cells are adjusted in time through chemical diffusion. The chemical concentrations within each cell can cause the cell to change its biological state. CELLs can be in one of three biological states: proliferating, quiescent, or necrotic. These states are governed by a cellular model. At the cellular level, the cell cycle is modeled though a simulated protein regulatory network. At the extracellular level, cells grow, shrink, and divide, as well as interact with one another through a Monte Carlo algorithm. Together, these three levels of simulation capture the fundamental processes that govern tumor development. All three levels interact within the framework of a 3 dimensional space, referred to as the computational GRID. The GRID comprises a series of SITEs, which are single computational units of space. A computational CELLs can grow in the GRID and will occupy a number of SITEs.

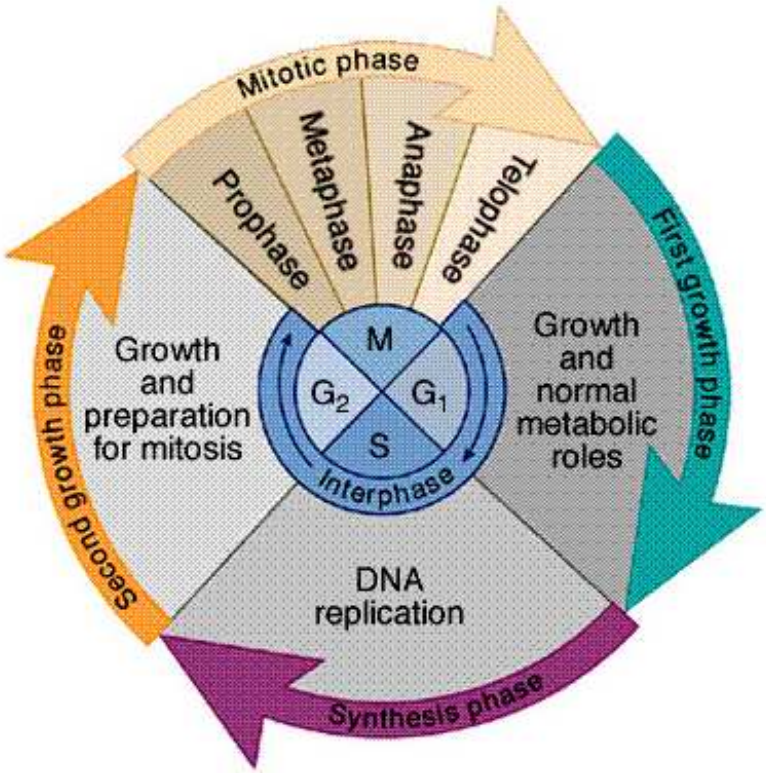

Figure 2.1: The four main phases of the cell cycle (Pennsylvania State University [2005]).

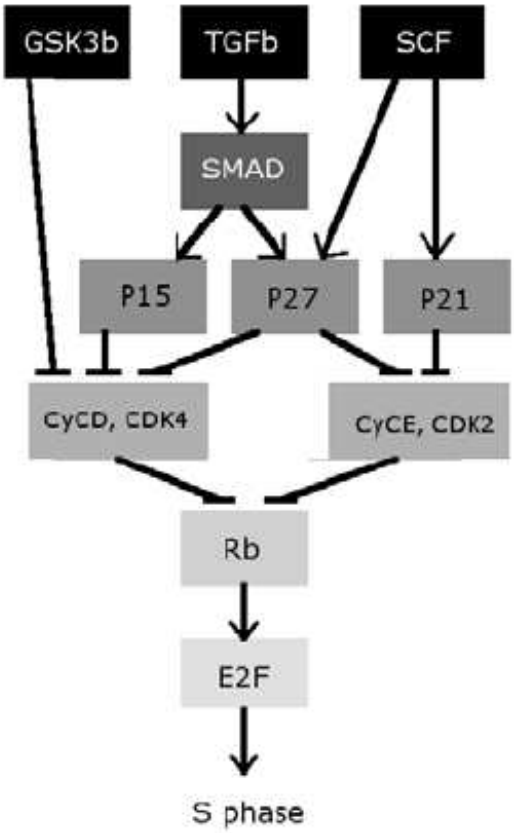

Figure 2.2: The protein regulatory network used in the simulation. Proteins are either be expressed or not expressed. A protein's state of expression may change depending on other proteins' expression state. The pointed arrows in the figure signify a stimulatory link while the flat arrows signify an inhibitory link. For example, the expression of SMAD stimulates the epxression of P15 and the expression of P15 inhibits the expression of CyCD and CDK4 (Jiang et al. [2005]). See Section 2.2.2 for further explaination. (Jiang et al. [2005]).

2.1 Chemical Diffusion: Subcellular Model

2.1.1 Introduction

A standard non-homogeneous diffusion equation is used to model the diffusion of chemicals throughout the computational domain:

$$\frac{\partial u(\vec{x},t)}{\partial t} = \kappa \nabla^2 u(\vec{x},t) + a(\vec{x}) \tag{2.1}$$

where t is time, $\vec{x}$ is the position in the domain, $u(\vec{x},t)$ is the concentration of a chemical at $\vec{x}$ and time t, κ is a diffusion constant, and $a(\vec{x})$ is a source/sink term. A numerical PDE solver was designed at LANL for this code. In Chapter 3, we present the numerical solver we implemented to allow for general boundary conditions.

2.1.2 Chemical Diffusion Details

The subcellular model currently considers five chemicals: oxygen, nutrients (such as glucose), waste products, growth factors, and inhibition factors. These five concentrations of chemicals in a grid site are allowed to change through the diffusion equation given above. The diffusion constant κ determines how quickly or slowly the chemicals diffuse, allowing smaller, more neutral chemicals to diffuse faster than larger chemicals. The $a(\vec{x})$ term allows chemicals to be used or produced within the cells themselves. For example, this occurs when a cell consumes oxygen and produces waste, which then diffuse to and from the cells.

After cells have been allowed to move, grow, and divide as determined by the protein network and Monte Carlo algorithm, the diffusion algorithm is applied to all of the cell sites. Using finite difference approximations, the simulation obtains a linear system of equations which is then solved using a custom iterative solver. Each CELL determines its own concentration after this diffusion process by averaging chemical concentrations over all SITEs it occupies. These average concentrations of chemicals are used later in the CELL reaction within the cellular model (see Figure 2.2).

2.2 Protein Expression: The Cellular Model

2.2.1 Introduction

This section is included in order to explain how the cell cycle is simulated in the program. We have not modified the behavior of this code, although

we have optimized its use of memory as explained further in Chapter 3.

CELLs progress according to the cell cycle shown in Figure 2.1 wherein they divide through mitosis in the M phase, move through the first growth stage G1 and a recombinant DNA phase S, and then prepare to divide again in phase G2. Once a cell develops past the G2 phase, it undergoes mitosis, spawning a daughter cell, and returns to the beginning of the cell cycle in phase G1. However, not all cells complete the entire cell cycle. Some become quiescent, essentially getting "stuck" in the cell cycle. It has been found that 85% of quiescent cells are stuck in the transition between the G1 and S phase [Jiang et al., 2005]. As a simplifying assumption, cells within the model can only become quiescent when they attempt to transition from G1 to S.

In order to model the cell cycle, the model's CELL contains a simplified protein regulatory network based on the Kyoto Encyclopedia of Genes and Genomes [Kyoto, 2005]. The protein regulatory network is outlined in Figure 2.2. This network is controlled by the CELL's environment and will govern its progression through the cell cycle. The combination of this protein network, its interaction with the growth and inhibition factors in the environment surrounding the CELL, and probabilistic transitioning between cell phases provides a surprisingly accurate computational model of the biological cell cycle.

2.2.2 Protein Expression Details

The protein regulatory network, which handles the phase progression of the cell cycle, only considers quiescent and proliferating cells because they are the only ones that can move through the cell cycle. A cell that has become necrotic is *dead* for the rest of the simulation and will not actively reenter the cell cycle. For a *live* cell, if the concentrations of oxygen, nutrients, and waste in the environment of the cell are above or below the necessary thresholds for normal cell development, a proliferating cell becomes quiescent and a quiescent cell becomes necrotic. If the chemicals are within given thresholds, the algorithm then calculates the cell metabolic rates and a "factor level" that describes the ratio of growth factor to inhibition factor the cell is receiving. Cellular metabolic rates are used for a number of purposes. In order for normal cell division and growth to occur, a cell must have the correct metabolic rate. In relation to the chemical diffusion model, metabolic rates define the production and consumption rates of chemicals: they are used to define the source and sink terms $\vec{a}(\vec{x})$ for each chemical within the diffusion equation described in Section 2.1.

The protein regulatory network used to model protein interdependencies is a system of levels in which proteins are set to either ON or OFF and are either stimulatory or inhibitory for proteins in the next level. In the protein regulatory network, if a protein is stimulatory, the next level protein will be turned ON if that original protein is ON, and OFF otherwise. If any number of proteins pointing to another protein are inhibitory, the next level will be turned OFF if any of the inhibitory proteins are ON, and on only if all of the inhibitory proteins are OFF (see Figure2.2).

The factor level is calculated as

$$Factor\ level = \left(1 + \exp\left(-\alpha\left(\frac{gF - ihF}{initGF} - \theta\right)\right)\right)^{-1},$$

where gF and ihF are current local concentrations of growth and inhibitory factors, respectively. Both are outputs of the extracellular chemical equations. $initGF$ is the concentration of growth factors in the medium surrounding the aggregate, while θ is a factor level threshold, and α is a free parameter (Jiang et al. [2005]). The factor level describes a transition probability that is used in conjunction with the protein regulatory network to determine the expression of each protein. A protein may be either ON or OFF. In order for a protein to make a transition between these states, the protein network must first allow for the transition. This is accomplished when all proteins which inhibit the expression of the protein of interest are in the OFF state and at least one protein which stimulates its expression is turned ON. When these conditions are met, the factor level is used as a probability for transition of the protein expression state.

Note that a higher growth factor concentration and a lower inhibitory factor concentration result in a higher factor level, which in turn results in a higher probability that a protein in the network will actually make a state transition provided that the regulatory network conditions are satisfied.

In the code, two conditions must therefore be met in order for a certain protein to turn ON - either a stimulatory link must be ON or all inhibitory links must be OFF, and a random number generated by the code is less than the factor level previously calculated, effectively making the factor level the probability of transition.

If the cell makes it through the entire protein regulatory network and meets a target level of growth, it will move onto the S phase. Once in the S phase, if the factor level is high enough and the cell is "on track" to continue growing at the desired rate, it will move into the G2 phase. Once it is within the G2 phase, it will become quiescent if it has not grown enough within a specified interval of time. If at any point, the cell does not move onto the

next phase or the necessary protein links are not turned on or off, a cell will become quiescent.

2.3 Monte Carlo Algorithm: Extracellular Model

2.3.1 Introduction

This section describes the algorithm used to model cellular growth over the GRID. We have not changed this algorithm, though we have extended it to handle the intercellular interactions of the CELL types that we incorporated, such as the VASCULARCELL. We include the description to provide a more complete explication of the simulation.

Monte Carlo algorithms describe a broad range of numerical methods that use function of random numbers and whose accuracies are bounded statistically. Monte Carlo algorithms are used in many diverse fields, from multi-variable integration to high-energy physics and game simulation. In contrast to numerical discretization of partial differential equations modeling the underlying physic of a system, Monte Carlo methods simulate the physical system directly. Thus, there is no need to develop equations describing the evolution of the system. Instead, Monte Carlo algorithms proceed through random sampling of a probability density function to direct the behavior of the system.

Our Monte Carlo algorithm provides cells with a mechanism for cell growth and movement. This algorithm, dubbed the *Metropolis* algorithm, is a stochastic algorithm. It proposes a random change in a system and then determines whether to accept or reject this change based on the system's response. The change in "energy" of the system is used as to determine the probability of acceptance of a proposed change. Here, "energy" can be any intrinsic function of the system which one desires to minimize. In our cellular model, energy is determined from the cells' volumes and predefined coupling energies with bordering cells. Natural systems tend toward lower energy, but are subject to stochastic variation, allowing temporary and random fluctuations. The Monte Carlo algorithm incorporates these realities of nature.

In general, a physical system is modeled with the Metropolis algorithm by first finding the current system's energy (determined from the intrinsic energies of the bodies being simulated and the energy of interactions between these bodies) and compares this to the energy of the system after a small change. If this change causes the total energy of the system

to decrease, the change is immediately accepted. Otherwise, if the change results in a higher energy, the modification is accepted with a probability determined from the Boltzmann factor, a probability density function describing many physical systems. In Chapter 3, we describe our changes to the Monte Carlo algorithm to include support for the vasculature.

2.3.2 The Monte Carlo Algorithms

We explain how the Monte Carlo program allows for CELLs to move and grow. First a random SITE is chosen within the GRID. The neighbors of this SITE are tested to determine whether it is contained entirely within a CELL or on a CELL border. If none of this SITE's neighbors are owned by another CELL, the SITE is known to be internal and another random SITE is chosen until a border SITE is found. Once a border SITE is found, a random neighbor SITE owned by another CELL is chosen. This results in an original grid SITE and a neighbor SITE, owned by a different CELL. The energy of the original configuration is calculated as a function of the volume of each CELL and its target volume, as well as on a matrix J describing the coupling energy between two CELLs:

$$E_s = \sum_i \gamma_i (V_i - V_{target,i})^2 + \sum_{j \in \{neighbors\}} J[s][j] \tag{2.2}$$

In Equation 2.2 , s is the SITE for which the energy is being calculated, i iterates over the two cells of interest at the border, γ is some constant prefactor, and $J[s][j]$ describes the interaction energy between the type of cell at s and the type of cell at j.

After this calculation, the algorithm assigns the original SITE's ownership to that of the neighboring cell. The same energy calculation is made on this modified configuration. If the energy is lower than that of the original configuration, the change is immediately accepted. If the energy is higher, the Boltzmann factor is used:

$$factor = e^{-\Delta E / k_B T} \tag{2.3}$$

where ΔE is the change in energy between the initial configuration and the final configuration after the proposed change, k_B is the Boltzmann constant relating temperature to energy, and T is the temperature of the system.

The result, $factor$, is the Boltzmann factor and represents a probability of acceptance. A random number is then generated from the uniform

distribution between 0 and 1, and the change is accepted if this random number is lower than the Boltzmann factor. Note that the Boltzmann factor depends on the two energies as well as the temperature, so a high change in energy would result in a lower Boltzmann factor and, therefore, a lower probability of acceptance. A lower temperature results in a larger Boltzmann factor and therefore a higher probability of acceptance. If the change is rejected, the ownership of the SITE in question reverts back to the original cell, and if the change is accepted, ownership of the SITE is unchanged.

Stochastic energy fluctuations are allowed in Monte Carlo algorithms as is allowed by statistical mechanics and observed in all finite temperature systems.

Chapter 3

New Developments: Streamlining, Vascularizing, and Adding Chemotherapy

The team has extensively revised the avascular tumor model described in Chapter 2. Support for simple vascular systems have been completely integrated into the code. We have created three different vasculature geometries, and the code can be extended to support any three-dimensional vasculature. Additionally, these vasculatures may be inserted after the simulation has been running for a pre-specified amount of time. This automatically creates new Dirichlet boundary conditions at SITES designated to be part of the vasculature.

3.1 General Code Improvements

Before implementing model changes, it was necessary to streamline the preexisting avascular code in order to improve the run-time, make more efficient use of memory, and fix computational errors. The team changed the CELL identification numbers to pointers, created new global constants, streamlined memory usage, and corrected miscalculations.

3.1.1 CELL IDs to CELL Pointers

We changed how CELLs are mapped to particular GRID SITEs. The original model generated a list of CELLs, each one with a unique identification number, or ID. Each Site on the GRID held the ID of the CELL in which it

was contained. There were disadvantages to using this method. In order to find the CELL corresponding to a given GRID SITE, the entire list of CELLs needed to be searched. This was inefficient because the simulation would search through many unrelated CELLs every time one of them needed to be accessed. Additionally, extra memory was required to handle a variable-length array such as this list of CELLs as well as more time since the CELLs were being constructed on the stack, copied into the array, then deleted from the stack. The team modified the entire code so that each Site in the GRID stores a pointer to the CELL to which it belongs. This method requires no additional memory overhead or additional CPU time. Also, this change makes the code easier to understand without removing functionality. For one, "magic numbers" or special CELL IDs were removed from the code. For example, the C++ code below is a common operation which finds a cell from a GRID SITE by extracting the CELL ID, searches the CELL array for the ID to determine the CELL Index, and, if not in the array, sets this index to zero, which is understood to be a dead cell.

```
int ID = grid(i,j,k).ID;
int cellIndex = getCellIndex(grid(i,j,k).ID);
if (ID > 0 && cellIndex < 0) cellIndex = 0;
```

By changing the data structure so that the grid contains CELL pointers, this code translates to the following

```
Cell* cell = grid(i,j,k).getCell();
```

which has the same functionality since one may easily determine whether the cell is necrotic, ask for its chemical concentrations, and modify it through this handle.

Such modifications make the code easier to read and cause the simulation to run slightly faster when run on a single computer. However, these changes may need to be undone to parallelize the code so it can run concurrently on multiple computers. When these modifications were made, ease of future parallelization had not yet become one of the goals of our project.

3.1.2 Constant Definitions

We modified the code to eliminate constants that were not used without being predefined, thus enhancing the readability and usability of the model code. Whenever these constants need to be changed, it is now possible to do so only once at the beginning of the code. Previously, it was necessary to search through every line to find and modify all instances of the constant.

3.1.3 Boolean Storage

A limitation of the original code is the amount of memory it uses. For large GRID sizes, the amount of memory is larger than the amount of RAM. When this occurs, the operating system will write the excess data to the swap space on the hard drive. When swapped data are needed, they are moved back into RAM, and another chunk of the data is moved to the hard drive. However, accessing the hard drive is several orders of magnitude slower than any other operation of the program. When this occurs, the program consequently spends the vast majority of its run-time simply reading and writing to the hard disk. This is known as "thrashing," and should be avoided if possible. In its original form, the simulation would thrash on any GRID size larger than about 220 sites on any computer with less than a gigabyte of RAM. This is the reason that the program would take several weeks to run on large GRID sizes, but slightly smaller simulations would finish in a matter of hours.

The team worked to alleviate this problem by optimizing the code to use less memory. The space required to represent protein expression in CELLs has been reduced. In any given cell, a protein can either be expressed or not expressed. In the CELL class, each protein was originally stored as an integer, which requires 8 bytes of space. If the integer was a 1, the protein was expressed in that cell. If it was a 0, the protein was not expressed. As an example, the following code relates the expression of the TGFB protein and the levels of growth and inhibition factor to the SMAD protein:

```
if (TGFB == 1 && drand48() < factorLevel) {
  SMAD = 1;
} else {
  SMAD = 0;
}
```

To conserve memory, the code now stores all proteins as booleans, which each require only one byte of space. The above code has been changed to the following:

```
if (TGFB && drand48() < factorLevel) {
  SMAD = true;
} else {
  SMAD = false;
}
```

With our modification, the CELL code is now 87% more memory efficient than the original code. Moreover, the code is now syntactically easier to read. If more memory efficiency is necessary, it is possible to store each protein in a single bit, resulting in another 87% savings in memory. However, this will result in code that is slightly (though insignificantly) slower and syntactically more convoluted. These and other changes to optimize the program's memory usage enables the simulation to run more quickly on larger data sets.

3.1.4 Incorrect Code

The team also made a small but important change in the code so that the simulation matches what is described in [Jiang et al., 2005]. In the CELL class, there is a function called `calculateMetabolicRates`, which calculates the rates of production/consumption of each of the five chemicals used in the simulation. Originally, this function calculated the rate of waste production as the average of the rate of oxygen consumption and the rate of glucose consumption, as shown here:

```
double perw = (perO2+pern)/2;
```

However, in the *Parameters* section on page 9 of [Jiang et al., 2005], it is stated that the rate of waste production should be 1.5 times the rate of glucose (nutrient) consumption. Moreover, it makes little sense to define the waste production rate as the average: this will break if the model is extended to include other metabolic pathways that consume oxygen. The code has been changed to be consistent with the paper. It now reads as follows:

```
double perw = 1.5 * pern;
```

3.2 Addition of Vasculature

Our main task this year was to incorporate vasculature into the model. This was a significant step towards accurate tumor modeling because we can then simulate the tumor's response to chemotherapy interacting with the tumor through the vasculature. In order to add vasculature, we needed to extend the CELL class and create a new VASCULARCELL class. It was also necessary to improve the diffusion solver so that it supports boundary conditions at the edges of the vasculature. Lastly, we needed to create a way to

insert the vasculature into the tumor after it had been growing avascularly for several timesteps. (Recall that the initial growth of a biological tumor is typically avascular).

The incorporation of vasculature required modifications to multiple segments of the code. The simplest change involved inclusion of new global constants to describe the concentrations of chemicals in the vasculature. These are defined in the definitions file at run-time. We also implemented a new class of tissue and a new PDE solver. The current implementation of the vasculature inserts itself into the GRID after a delay in which the tumor grows avascularly. The following sections describe how these changes were made. The Appendix includes further examples of altered source code.

3.2.1 TISSUE Class Addition

The preexisting CELL class supported necrotic, quiescent, and proliferating cells. In order to create the vasculature, the team created a class of objects similar to the CELL class. In order to create these behaviors, a superclass, TISSUE, was created, and CELL was changed to inherit from TISSUE. The TISSUE class fulfills all of the generic functions of a cell on the GRID, such as supporting functions involving the center of mass of a cell, the storage of chemical concentrations, and so forth. The CELL class handles behavior particular to CELLs. It implements the `react()` function that governs protein expression, as well as the functions to make cells quiescent, necrotic, and proliferating.

We created the VASCULARCELL class, which also inherits from the TISSUE class. This has allowed us to keep the vasculature as a kind of TISSUE that can occupy SITES on the GRID, yet change some aspects of the class to give the desired functionality. For instance, protein synthesis does not occur in the bloodstream, so the protein expression portion of the model was modified to not change the proteins expressed in the vasculature. The functions which change the chemical concentrations in the TISSUE have also been overwritten to not change the concentrations within the vasculature, which are intended to remain constant.

3.2.2 The Actual Creation of the Vasculature

In order to make the vasculature geometries as versatile as possible, we insert the vasculature into the simulation using the VASCULARMAKER class.

This class can easily be extended to include more vasculature geometries than the three we currently provide.

To create a new vasculature geometry, one must create a new function in the VascularMaker class, using the `makeSiteVascular()` function to actually add specific Sites in the GRID to the vasculature. One must then edit `layoutVasculature()` to call this new function and create a vasculature with the new geometry. The function `layoutVasculature()` creates all of the different vasculature geometries, and chooses the correct one based on the value of the global integer `VascType`, specified in the definitions file passed into the simulation at run-time. If the value of `VascType` is not a valid form of vasculature, no vasculature is inserted. The class currently supports the creation of a single linear blood vessel, a hexagonal mesh, a rectangular mesh, or no vasculature. These vasculatures are further specified using the global parameters `VascRadius` and `VascLayers`. `VascRadius` specifies the radius, in SITES, of each blood vessel. `VascLayers` specifies the number of planar meshes (either hexagonal or rectangular) inserted into the GRID; these meshes are evenly spaced through out the entire GRID.

3.2.3 Time-Delayed Addition of the Vasculature

Our liaison at LANL requested that we structure the code so that the vasculature could be inserted after the tumor growth simulation has begun. That is, the model should grow an avascular tumor for a given amount of time before the vasculature is created. Separate code simulating the angiogenesis stage of tumor growth is currently unders development at LANL. It is anticipated that this angiogenesis code and the code simulating vascularized growth that we have developed will ultimately be combined to allow for modeling multistage tumor growth.

To facilitate the delayed appearance of vasculature in the tumor, we added a new parameter to the definitions file, called `VascDelay`. It specifies the number of days that the tumor is allowed to grow before insertion of the vasculature . When vasculature is inserted, the simulation iterates through each SITE in the GRID where vasculature will be inserted and, if it is already occupied by a cell, shrinks that cell so that it no longer occupies the SITE. The vasculature is then inserted into the SITE.

A potential problem with this method is that it is theoretically possible to cut a CELL into two separate parts by inserting the vasculature through the middle of the CELL. However, the team believes that this situation is sufficiently unlikely to occur, and if it does, sufficiently unlikely to cause problems with the simulation. This may occur if a CELL grows in a barbell

shape and the vasculature is inserted in the middle of the cell, but we have not seen any such unusually shaped CELLs.

If the vasculature has a planar geometry, it may slice a CELL into halves, but such a geometry would be biologically unfounded (though using such a geometry may still cause this CELL slicing to occur). We suspect that even if such a CELL is split, this would not make a significant difference: It would affect the diffusion of chemicals because the concentrations of chemicals are averaged over all SITEs within a CELL, but if the CELL is close enough to the vasculature for it to be cut into two parts the concentrations of the chemicals within it should all be quite close to the concentrations in the vasculature itself, and therefore the concentrations should not be significantly affected.

There is one more case in which this "tunneling" method of inserting the vasculature may have an unusual impact on the simulation. If tumor CELLs are significantly larger than the vasculature, it is possible to "impale" a CELL so that the vasculature goes in one side of it and out the other. However, this will not actually impact the CELL adversely, because it is still a single piece. Although this situation is not realistic because it can result in toroidal CELLs, it should not adversely affect the simulation.

3.2.4 Diffusion Model Modifications

The introduction of the vasculature required a restructuring of the original diffusion model, as described below. Since blood flow occurs at a much faster rate than the diffusion modeled in the simulation, changes in the concentrations of chemicals in the vasculature due to diffusion are overwhelmed by changes due to the circulation of the blood itself throughout the rest of the body. The human heart pumps approximately 6 quarts of blood per minute. This is equivalent to cycling the body's entire blood supply once. Therefore in 45 minutes (the real time being modeled within one diffusion step) the blood in a human body would have been cycled almost 45 times through the vasculature. Consequently, the team decided that the best model would be for the vasculature to maintain constant concentrations of the 5 chemicals internally.

These constant concentrations are ideally implemented as a Dirichlet boundary condition in the diffusion equation solver. However, the incorporation of such boundary values in a vasculature of arbitrary geometry into the diffusion equation is difficult because changing the geometry of the vasculature changes the boundary values of the PDE problem.

Initially, our liaison suggested we approximate the boundary condi-

tions by solving the diffusion equation without the boundaries, and then resetting the concentrations of the chemicals on the boundary itself. Unfortunately, this method did not produce the desired result. After further consultation, we then rewrote the entire diffusion solver to handle Dirichlet boundary conditions in arbitrary geometries. This new solver gives the correct behavior. The two methods we tried are described below.

The First Method

The team initially attempted to perform an approximation to constant chemical concentrations, instead of actually treating them as boundary conditions. At the end of each diffusion calculation, the concentrations of chemicals within the vasculature were reset to their original values. This computational technique allows the model to avoid solving for the computationally expensive non-symmetric boundary values that would otherwise arise. This approximation had already been used by our liaison to attempt to keep the concentration of chemicals in the medium surrounding the tumor constant, and it proved useful in that instance. Although this approximation gave slightly different results than a similar model in which the concentrations in the vasculature were treated as actual boundary values, this method made the equations solvable by the original PDE solver that was already implemented in the code, and could therefore be seamlessly integrated into the simulation.

Unfortunately, this approximation did not work as well as expected. The team implemented this method and experimented with different frequencies of resetting the concentrations, but the rates of production and consumption of chemicals in cells and the rates of diffusion of the chemicals overwhelmed this resetting, and the concentrations of the chemicals within the vasculature were all but governed by the concentrations of the chemicals in the surrounding TISSUE. Due to this setback, the team decided to completely re-implement the PDE solver to support arbitrary Dirichlet conditions.

The New Diffusion Solver

We created a new diffusion solver that uses a backward-Euler finite difference approximation to iteratively solve the diffusion equation using the Gauss-Seidel method. We outline the foundation of the technique below.

Derivation of the Algorithm The general partial differential equation governing diffusion is given by

$$\frac{\partial u(t,\vec{x})}{\partial t} = \kappa\nabla^2 u(t,\vec{x}) + a(\vec{x}) \tag{3.1}$$

where $u(t,\vec{x})$ is concentration at location $\vec{x}$ and time t, κ is a diffusion constant, and $a(\vec{x})$ is a source/sink term.

In order to numerically integrate this equation in time, one must first discretize the system in time and space. Thus, the spacial domain becomes a grid of size $X \times Y \times Z$ and u shall be redefined over this domain:

$$u(t,\vec{x}) \mapsto u^t_{x,y,z} \qquad\qquad a(\vec{x}) \mapsto a_{x,y,z} \tag{3.2}$$

where $0 < x < X$, $0 < y < Y$, $0 < z < Z$ are integer indices and X, Y, Z are the dimensions of the grid. Writing t as a superscript is only out of convenience.

Recall the Second Central Difference Approximation is given by

$$\frac{\partial^2}{\partial x^2}u^t_{x,y,z} = \frac{u^t_{x+1,y,z} - 2u^t_{x,y,z} + u^t_{x-1,y,z}}{\triangle x^2} + \mathcal{O}(\triangle x^2) \tag{3.3}$$

Substituting this approximation into the PDE yields

$$\frac{\partial u(t,\vec{x})}{\partial t} = \kappa\left(\frac{u_{x+1,y,z} - 2u_{x,y,z} + u_{x-1,y,z}}{\triangle x^2} + \cdots\right) + a_{x,y,z} \tag{3.4}$$

and by using a cubic grid where $\triangle x = \triangle y = \triangle z$, one obtains

$$\frac{\partial u(t,\vec{x})}{\partial t} \approx a_{x,y,z} - \frac{\kappa}{\triangle x^2}\left(6u^t_{x,y,z} - \sum_{x',y',z'} u^t_{x',y',z'}\right) \tag{3.5}$$

where x', y', z' are taken over the 6 adjacent neighbors of grid site x, y, z.

To achieve stability, the Backward Euler scheme may be applied to yield

$$u^{t+\triangle t}_{x,y,z} - \triangle t\left(a_{x,y,z} - \frac{\kappa}{\triangle x^2}\left(6u^{t+\triangle t}_{x,y,z} - \sum_{x',y',z'} u^{t+\triangle t}_{x',y',z'}\right)\right) = u^t_{x,y,z} \tag{3.6}$$

The right hand side of Equation 3.6 can clearly be calculated explicitly. Define

$$\Phi^t_{x,y,z} \triangleq u^t_{x,y,z} + \triangle t a_{x,y,z} \tag{3.7}$$

Then,

$$u_{x,y,z}^{t+\triangle t} + \frac{\kappa \triangle t}{\triangle x^2}\left(6u_{x,y,z}^{t+\triangle t} - \sum_{x',y',z'} u_{x',y',z'}^{t+\triangle t}\right) = \Phi_{x,y,z}^{t} \tag{3.8}$$

and therefore

$$\left(1 + 6\frac{\kappa \triangle t}{\triangle x^2}\right) u_{x,y,z}^{t+\triangle t} - \frac{\kappa \triangle t}{\triangle x^2} \sum_{x',y',z'} u_{x',y',z'}^{t+\triangle t} = \Phi_{x,y,z}^{t} \tag{3.9}$$

which is simply a matrix equation with diagonal terms $\left(1 + 6\frac{\kappa \triangle t}{\triangle x^2}\right)$ and 6 off-diagonal terms $\left(-\frac{\kappa \triangle t}{\triangle x^2}\right)$.

The team chose to implement the Gauss-Seidel iterative method to solve the above sparse linear system. For a general linear system $Ax = b$, or $\sum_{j=1}^{n} a_{ij}x_j = b_i$ (where the matrix components a_{ij} should not be confused with the forcing term a_{xyz} of Equation 3.2), the Gauss-Seidel iteration is given by

$$x_i^{k+1} = \frac{1}{a_{ii}}\left(b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{k+1} - \sum_{j=i+1}^{n} x_j^k\right) \tag{3.10}$$

where x_i^k represents the kth iterate of the ith component of the vector x. In our case, this iteration translates into

$$u_{x,y,z} = \frac{1}{\left(1 + 6\frac{\kappa \triangle t}{\triangle x^2}\right)}\left(\Phi_{x,y,z} + \frac{\kappa \triangle t}{\triangle x^2} \sum_{x',y',z'} u_{x',y',z'}\right) \tag{3.11}$$

where we are updating the iterate $u_{x,y,z}$ immediately while iterating through the grid.

Boundary Conditions In the Gauss-Seidel iteration, Dirichlet boundary conditions may be enforced by setting

$$u_{x,y,z}^{BC} = u^0 \tag{3.12}$$

where u^0 is a boundary value and $u_{x,y,z}^{BC}$ is a boundary site. In terms of the matrix equation, this is equivalent to the row corresponding to site (x, y, z) consisting of only a one on the diagonal. Thus, by setting all boundary sites $u_{x,y,z}^{BC}$ to the boundary value before the Gauss-Seidel iteration and skipping

the update from Equation 3.11 of the Gauss-Seidel iteration for just these boundary sites, this constraint will be conserved.

Thus, we are able to treat arbitrary boundary geometries by applying this strategy on a site-by-site basis. That is, non-boundary sites will obey Equation 3.11, while boundary sites obey Equation 3.12.

Convergence It is clear that our linear system is strictly diagonally dominant since

$$|a_{ii}| = 1 + \frac{3\kappa\triangle t}{\triangle x^2} \; > \; \sum_{j\neq i} |a_{ji}| = 6 \cdot \frac{\kappa\triangle t}{2\triangle x^2} = \frac{3\kappa\triangle t}{\triangle x^2} \tag{3.13}$$

Therefore, the spectral radius of our linear system must be less than 1 so the Gauss-Seidel iteration must converge for any initial condition $u^{(0)}$. (Saad [1996])

Furthermore, the backward Euler method is unconditionally stable with error linear in step size. Therefore, the backward Euler method guarantees convergence to the exact solution as $\triangle t \rightarrow 0$ and the Gauss-Seidel iteration gaurantees convergence to the backward Euler method's approximate solution for any given $\triangle t$.

In implementation, the Gauss-Seidel method requires a break condition. We use a vector residual condition given by

$$||u^{t+\triangle t} - u^t|| \quad < \quad \epsilon \tag{3.14}$$

where $||\cdot||$ is the Euclidean norm and ϵ is a given tolerance level. Since the number of components of u (the dimension of u) scales with the total grid size XYZ, we choose $\epsilon \propto \frac{1}{XYZ}$ to achieve error which is proportional only to the step size $\triangle t$, which orginates from the backwards Euler method.

3.2.5 The Monte Carlo Model

Finally, the Monte Carlo code needed to be extended to support vasculature. The J-matrix (described in Section 2.3) which is used by the model for energy calculations to determine the growth of cells was extended by one row and one column to contain relational constants for the vasculature. To calculate these values, an example configuration of cells and cites next to vasculature was considered. From this configuration, we calculated the energy, which includes the unknown J-matrix energy for surface binding energy between the vasculature and other cell type. We then solved this equation for the previously unknown J-matrix entry. The energy equation

contains two terms: a volume component, and a configuration component. The configuration component in this calculation contains the unknown J-matrix energy, and must be the same order of magnitude as the volume component. So, a J-matrix value is determined which ensures that this is the case. This process was repeated for each unknown J-matrix entry.[1]

Additionally, the team generalized the code to support J-matrices of arbitrary size. This will allow additional CELL types to be quickly and easily added in the future. Moreover, in our current implementation, vasculature cells cannot become any other cell type and do not change in size or multiply. This has been added to the model so that the Monte Carlo algorithm will not change the vasculature in any way.

3.3 Chemotherapy

One of the most commonly used chemotherapy drugs is cyclophosphamide. Cyclophosphamide is a non-cell cycle specific drug, which means that cylophosphamide treatments can be modeled without modifying the subcellular component of the model. Cyclophosphamide induces apoptosis in rapidly dividing cells by alkylating their DNA. This prevents DNA replication and ultimately results in cell death. In order to model cyclophosphamide treatments, we made several approximations. Due to the structure of the model, these approximations were necessary, but may be relaxed if certain changes are made to the model in the future.

3.3.1 Limitations of the model

Several aspects of the model limit the accuracy of possible chemotherapy modeling. First, the model contains no mechanism by which necrotic cells can be removed. Because chemical diffusion and growth dynamics are affected by the presence or absence of cells, this can negatively affect the accuracy of chemotherapy simulation using this model. Secondly, the toxicity responses of patients to particular chemotherapy dose levels are highly variable (Gurney [2002]) and so a model which does not consider variable toxicity effects will have limited use. Currently, there is no metric within this model for measurement of patient toxicity, because the model was designed for in vitro tumor spheroids.

[1] The team would like to thank Amy Bauer at LANL for her explanation of this method.

3.3.2 Cyclophosphamide Treatment Implementation

The reduction in tumor size in response to a dose of cyclophosphamide is governed by

$$N = N_o * exp(-KD_o), \tag{3.15}$$

where N_o is the number of non-necrotic cells present in the tumor prior to the treatment being administered, N is the number of non-necrotic cells present in the tumor after the treatment, K is a constant representing the tumor-killing efficacy of the particular drug (for cyclophosphamide, $K \approx 5$), and D_o is the amount the drug administered, in grams (Osteen et al. [2001]). A typical initial dose of cyclophosphamide is 1.6 grams (Gurney [2002]). In practice, further dose amounts will typically be adjusted depending on the patient's response to the initial and subsequent doses, but as mentioned above, this is not within the scope of the model. So, D_o will be held constant at 1.6 grams.

A number of computational simplifications are necessary in order to incorporate this into the model. It is not clear how long it takes for the tumor cells to be killed by a particular chemotherapy. As a simplifying assumption, if a chemotherapy dose is to kill X tumor cells, and there are T time steps between doses, we assign each cell a probability of death such that the expected number of cells killed during each time step is X/T. During each time step, we calculate the total amount of cyclophosphamide that is found in non-necrotic cells, $D_{(nn)}$. Then, we assign to each cell i a probability P_i of death:

$$P_i = D_i(1 - exp(-8))/(TD_{(nn)}). \tag{3.16}$$

The coefficient $D_i/D_{(nn)}$ assures that the probability of a cell dying is dependant upon the amount of cyclophosphamide it contains. Because the sum of these coefficients over all cells is identically 1, the expected number of cells killed is correct.

3.3.3 Implementation

We implemented a stochastic model to simulate the chemotherapy. Chemotherapy, and in particular cyclophosphamide, has been added as a new chemical in the model, and it diffuses from the vasculature and medium into the tumor like all other chemicals. We have added the global constants ChemoPeriod and ChemoDuration to describe doses of chemotherapy: ChemoPeriod is the time from the beginning of one dose to the beginning of the next, while ChemoDuration is the time from the beginning of a

dose to the end of that dose. The first dose starts when the vasculature is inserted, and doses begin at regular intervals after that.

The First Implementation

In our original implementation of the chemotherapy model, each CELL has a probability of being affected by cyclophosphamide based on the global constant ChemoDeathFactor. This constant holds the value of $(1 - e^{-kD_0})$, which is the fraction of the tumor that will die each dose. At each step, every proliferating cell has a probability of dying from chemotherapy. This probability is proportional to the relative amount of chemotherapy in the cell, the ChemoDeathFactor, and inversely proportional to the length of the dose. If a cell is killed, all of the chemotherapy in it is "used up" and the concentrations of cyclophosphamide within the cell are set to 0. The SITEs in the CELL then become part of the medium again, which is analogous to the cell being swept away by the bloodstream or otherwise absorbed into the body.

This model reflects the empirical behavior of actual tumors (Osteen et al. [2001]). However, it leads to some counterintuitive results: the fraction of the tumor that dies is equal to the ChemoDeathFactor, but the fraction of it that survives is slightly more than the remaining amount. This occurs because the cells that are not lysed are still dividing. As a simple example, it is possible for half the cells to die, half of them to divide, and the total number of proliferating cells in the tumor to remain the same. Unfortunately, this behavior is realistic and is the reason most patients require multiple doses of chemotherapy.

The Second Implementation

Although this model exhibited the desired properties of chemotherapy, it did not exhibit this behavior independently. Instead, one of the parameters given to the model was the toxicity of the cyclophosphamide in the form of the expected porpotion of cells that were to die each step. What we actually desired was a simulation which exhibited this as an emergent phenomenon.

Apoptosis Consequently, we created another stochastic model of cyclophosphamide. In this new model, each CELL has a probability P_i of undergoing apoptosis defined as follows:

$$P_i = 1 - e^{-C/C_0} \tag{3.17}$$

where C is the concentration of cyclophosphamide within the CELL, and C_0 is a reference concentration (See CHEMOREFCONC in the parameters input file). Note that when $C = C_0$, the cell has approximately a 63.2% chance of entering apoptosis.

Additionally, we have implemented a model of cell apoptosis. When a CELL is affected by cyclophosphamide, it becomes an apoptotic cell, which has been implemented as a new CELL type. When a CELL becomes apototic, it will remain apoptotic for a time specified by AACTIVATIONTIMER from the parameters file before undergoing phagocytosis, when the cell's constituent SITEs turn back to medium. Apoptotic cells have their own sets of metabolic rates separate from other CELL types. At this point, we have set these metabolic values to those of queiscent cells, though this is likely an overestimate of an actual apototic cell's metabolic rates. These values are also easily changable from within the input parmeters file passed to the program at execution.

Furthermore, CELLs now metabolize cyclophosphamide even when they are not apoptotic. However, due to time constraints, we could not obtain clinical data concerning rates of cyclophosphamide metabolism. Currently, we have assigned values for these rates which we believe are of the correct order of magnitude based on the values given for growth factor metabolism.

Chemotherapy Schedules Schedules of chemotherapy treatments involve a number of factors such as the amount of chemotherapy chemicals administered in a single dose and the amount of time between doses. Both of these factors have been included as parameters in the input file. CHEMOCONC specifies the maxmimum concentration of chemotherapy concentration within a dose period. CHEMOPERIOD specifies the time from the start of one dose to the start of another. A chemotherapy dose is composed of a chemical profile goverened by

$$C(t) = C_0 bte^{(-bt+1)} \tag{3.18}$$

where $C(t)$ is the concentration of chemotherapy chemicals at time t, C_0 is the maximum value of $C(t)$, and b is a time scale parameter calculated from CHEMODURATION such that $C(\text{CHEMODURATION}) = .05C_0$. This yeilds a chemical profile similar to that which is found in AUC curves (Bardelmeijer et al. [2000]).

3.4 Matlab Imaging

The imaging code used to view the experimental results was created using Matlab (Mathworks [2006]). The functions were inspired by code given to us by Los Alamos, though most of the current imaging routines have been written mainly from scratch. The following is a description of each type of image available to the user as well as how the tool might be used. A more detailed description of how to run each file is included in the Appendix along with the code.

The first plot type is a simple 2-D slice of the tumor at one instant in time, made using `TumorSlice.m`. The user specifies in which plane the slice is taken (x-y, y-z or x-z) and at which depth. The user can also specify which chemical (or cell types) to image. The output for this m-file is two images. The first is a color map of concentrations or cell types.[2] It shows in a simple and direct way the chemical concentrations or cell types in the tumor as a function of location. The second image is a height map of the concentrations or cell types over the 2-D slice. This plot has been especially useful for debugging any changes to the diffusion code but can also be used to view the effect on chemical concentrations with the addition of the vasculature.

The second plot type is a movie which shows a series of 2-D slices passing through the tumor. These are made using `TumorPassthrough.m`. The user specifies the axis which he or she wishes to travel through as well as which chemical (or cell types) to image. This is the best way to get a complete view of the tumor at one instant in time. This movie, as well as others, can be saved in MPG format using the MATLAB plug-in `MPGWRITE`. Once saved, it is easy for the user to scroll through the tumor using any MPG viewer. This image type is also useful when determining the effects of chemotherapy on the tumor. One may look at the layers in the tumor just above and just below the vasculature in order to see its effects.

The third plot type is a movie that shows a series of 2-D slices as the tumor grows over time. These are made using `TumorGrowth.m`. The user specifies in which plane the slices are to be taken (x-y, y-z or x-z) and at which depth. The user can also specify which chemical (or cell types) to image. This image type has to open up a new file to output each frame in the movie, and consequentially it takes a fair amount of time to completely render the movie (several minutes for large movies). However, this is the only way to see how the cells move and divide over time.

[2] For examples, see Figure 4.1 and Figure 4.2.

The last plot type is a plot of chemical concentrations along an x, y or z line in the grid and is created using `A2dchemplot.m`. This plot shows a vertical cross-section of the height map generated with the file `TumorSlice.m`. The user specifies which line of sites to look at by giving the depths of the other two axes. The user also specifies which chemical should be plotted. This plot is the best way, other than simply looking at the numbers, to display chemical concentrations in the grid.

Chapter 4

Experimental Results

We first conducted a run with no vasculature insertion. This provides us with base-level results for comparison with vasculature insertion and chemotherapy results. We conducted experiments on three different types of vasculature in order to demonstrate the adaptability of our section of the model. The three types of vasculature are: a single line through the tumor and the grid, a hexagonal lattice throughout the entire grid, and a square lattice throughout the entire grid. For each of these types of vasculature, we have implemented two distinct experiments: one in which the vasculature was in place before the tumor was grown and the second in which the vasculature was tunneled into the tumor at a delay of three days. The experiments without the delay resemble data collected from histoculture growth within a laboratory, and the experiments with the delay resemble data collected from in vivo tumor growth. This delay would approximate angiogenesis if we were able to grow the tumor to a detectable growth size.

In the experiments with the single line vasculature (Figure 4.3(C),4.3(D)), we see an interesting difference between the delayed vasculature insertion and the non-delayed vasculature insertion. It appears that the tumor is larger in the non-delayed vasculature insertion case, which is what we would expect as a longer time of growth with vasculature would result in more nutrients and therefore quicker growth of the tumor as well as a larger number of healthy, proliferating cells. In the experiments with the square grid structure (Figure 4.3(E),4.3(F)) and the hexagonal grid structure (Figure 4.3(G), 4.3(H)), we do not observe as noticeable a difference between the growth with delay and the growth without delay. However, we do observe more proliferating cells in the non-delayed vasculature as expected. It should also be noted that the oxygen levels for each of these

experiments are plotted in Figure 4.4.

In Figure 4.4(C), the number of quiescent cells vs. time for the line vasculature without delay shows an interesting plateau during the beginning of the second day but then rebounds to a steep increase. The dip in quiescent cells in the vasculature with delay (Figure 4.5(C)) occurs at the beginning of day 3 as the vasculature is added, which is as we would expect. The increase in nutrients to the cells with the added vasculature would slow the rate of proliferating cells becoming quiescent cells.

The experiments with the square grid structure depict another interesting plateau in quiescent cells growth (Figure 4.6(C)), from early in the simulation through to the middle of the second day. Once that point is hit, the growth of quiescent cells grows similarly to that with line vasculature without delay. The square grid vasculature with delay growth curves (Figure 4.7(C)) look similar to the line vasculatre with delay growth curves.

Interestingly enough, the hexagonal grid structure with no delay shows no quiescent cells until somewhat into the second day at which point it resumes similar growth as the other vasculatures, as seen in Figure 4.8(C). The growth curves for the vasculature with delay (Figure 4.9(C)) resemble the square grid vasculature with delay.

In Tables 4.1-4.3 one can note the small amount of variation between the growth of tumors for delayed or non-delayed vasculature and a non-vascularized tumor. We would not expect a large difference between these numbers as a small tumor would receive a large amount of nutrients from the medium so that the added nutrients from the vasculature only make a small impact on the growth of the cells.

In the runs with chemotherapy, we ran two experiments that considered different levels of chemotherapy being administered to the tumor. Each of these experiments was conducted on delayed line vasculature. The high dose is 2.0 grams of the chemotherapeutic agent and the low dose is 1.2 grams of the chemotherapeutic agent [1]. The chemotherapeutic agent is added to the vasculature at the same time as the vasculature is added. Once the chemotherapeutic agent is added, a majority of the proliferating cells become apoptotic and soon dissolve into the medium. Shortly thereafter, tumor growth resumes as we would expect. Note the significant differences in total and proliferating cell counts, displayed in Figures 4.10 and 4.11.

It is apparent from Figure 4.10 as well as Figures 4.11 and 4.12 that the

[1] For cyclophosphamide, an average dose as calculated by the body-surface area method is 1.6 grams (Gurney [2002])

amount of chemotherapy dose makes a significant difference in the growth of the tumor. The growth curve depicted in Figure 4.11(C) shows a smaller dip in the proliferating cells than in Figure 4.12(C) with a high dose. The tumor volume and radius depicted in Figure 4.11(A,B) and Figure 4.12(A,B) also show significant differences in the size and extremities of the minimum and maximum but similar patterns of growth.

Tables 4.4 and 4.5 denote cell counts of tumors for the non-chemotherapy runs at several different time steps. Tables 4.6 and 4.7 show counts for the high and low doses of chemotherapy, respectively. Note that the number of proliferating cells at 64 time steps with no chemotherapy ranges from 6196 to 6652 cells and is 3121 with a low dose of chemotherapy and 1204 with a high dose of chemotherapy. Although the chemotherapy runs are only one experiment at each dose, the variation between the chemotherapy vs. non-chemotherapy runs is large enough to be deemed significant. Also note that the quiescent cell counts have an even larger disparity with non-chemotherapy runs ranging from 6331 cells to 6557 cells and a low dose chemotherapy count is 1533 and a high dose chemotherapy count is 256.

It is important to keep in mind that these results only reflect the effect on the tumor and not on the patient, and that these levels of chemotherapy are not necessarily possible with many patients. Many drugs are known to have a wide coefficient of variation, and one must couple this knowledge with the differences in the response of the tumor in the high dose and the low dose of chemotherapy. With this coupled knowledge, it is clearly necessary to add more features to the model to allow for incorporation of the demonstrated coefficient of variation.

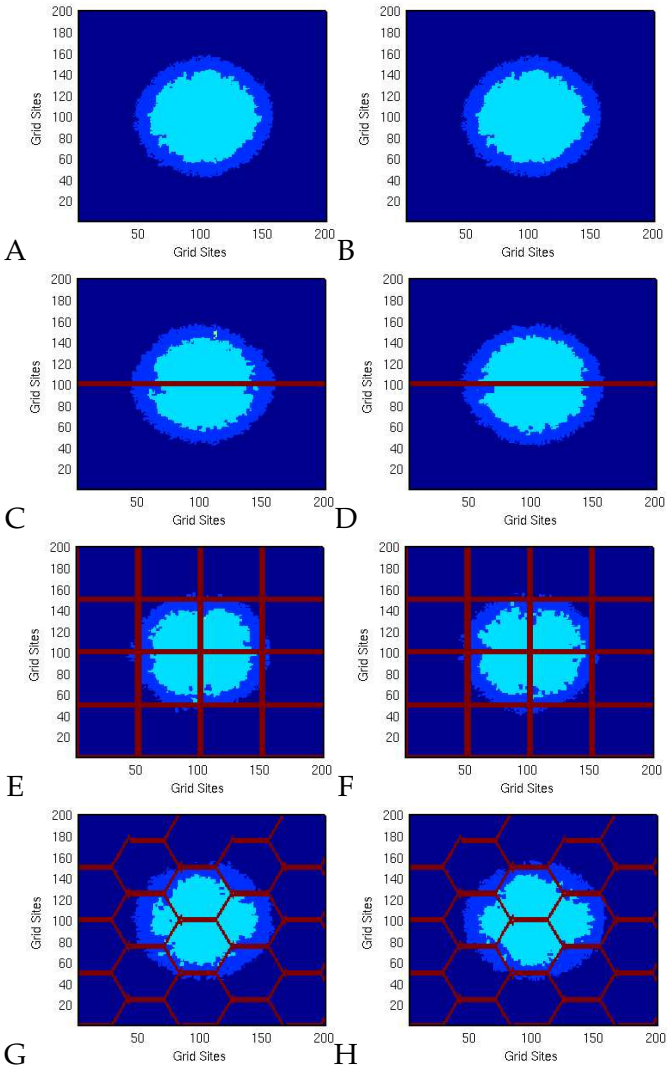

Figure 4.1: Cell types of tumor after 64 Monte Carlo steps (5 days 8 hours) grown with no vasculature (A,B), line vasculature (C,D), square grid vasculature (E,F), hexagonal grid vasculature (G,H). In the delayed vasculature (B,D,F,H), the line of vasculature was added through the center along the x axis at the beginning of day 3. Images are an x-y cross section at a z elevation of 100 GRID SITEs. The length of one SITE is equivalent to 4.2E-4 cm or 4.2 μm. Dark blue represents medium, blue represents proliferating cells, light blue represents quiescent cells and red represents the vasculature.

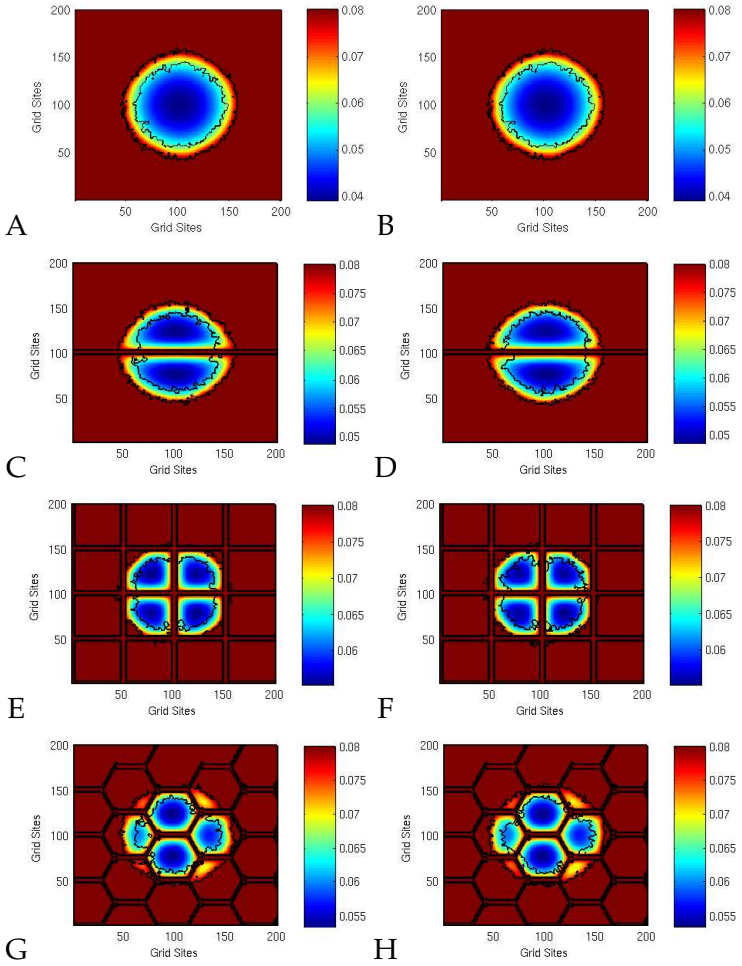

Figure 4.2: Oxygen levels of tumor after 64 Monte Carlo steps (5 days 8 hours) grown with no vasculature (A,B), line vasculature (C,D), square grid vasculature (E,F), hexagonal grid vasculature (G,H). In the delayed vasculature graphs, (B,D,F,H), the vasculature was added through the center along the x axis at the beginning of day 3. Images are an x-y cross section at a z elevation of 100 GRID SITES. The length of one SITE is equivalent to 4.2E-4 cm or 4.2 μm. The color bar at the right of each image describes oxygen levels, with high concentrations in red and low concentrations in blue.

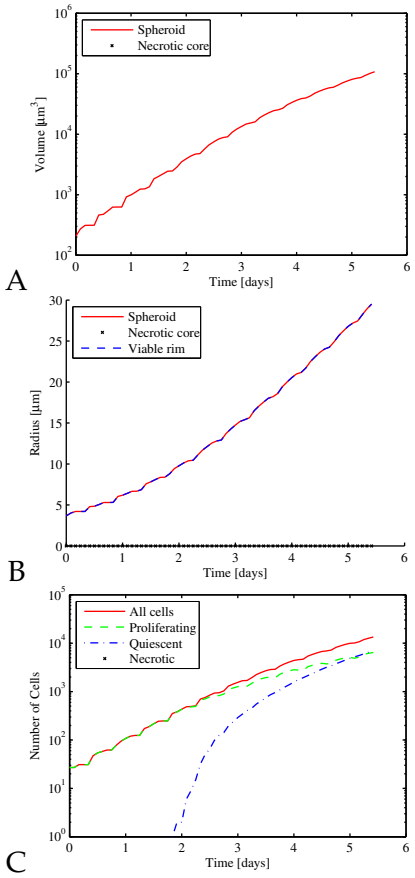

Figure 4.3: Growth curves for no vasculature. (A) Log plot of tumor volume (cm^3) vs. time (days) of tumor grown. (B) Tumor radius (cm) vs. time (days). (C) Log plot of number of cells vs. time (days) of a tumor grown. Red represents total number of cells, green represents number of proliferating cells, blue represents the number of quiescent cells and there are no necrotic cells

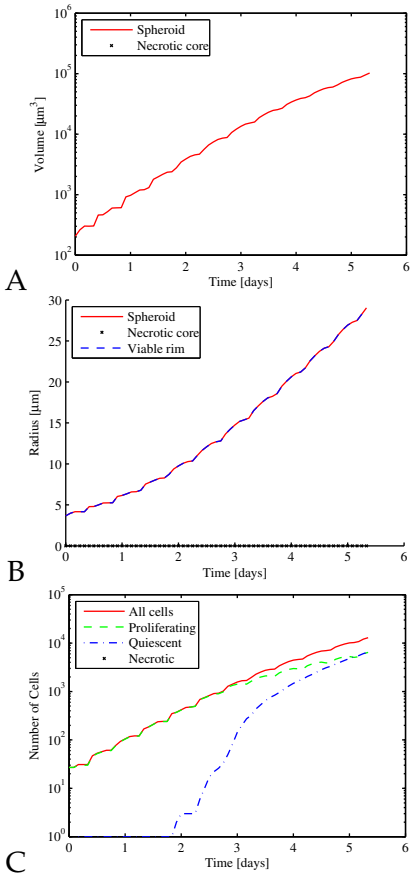

Figure 4.4: Growth curves for line vasculature without delay. (A) Log plot of tumor volume (cm^3) vs. time (days) of tumor grown. (B) Tumor radius (cm) vs. time (days). (C) Log plot of number of cells vs. time (days) of a tumor grown with a line of vasculature through the center along the x axis at the beginning. Red represents total number of cells, green represents number of proliferating cells, blue represents the number of quiescent cells and there are no necrotic cells.

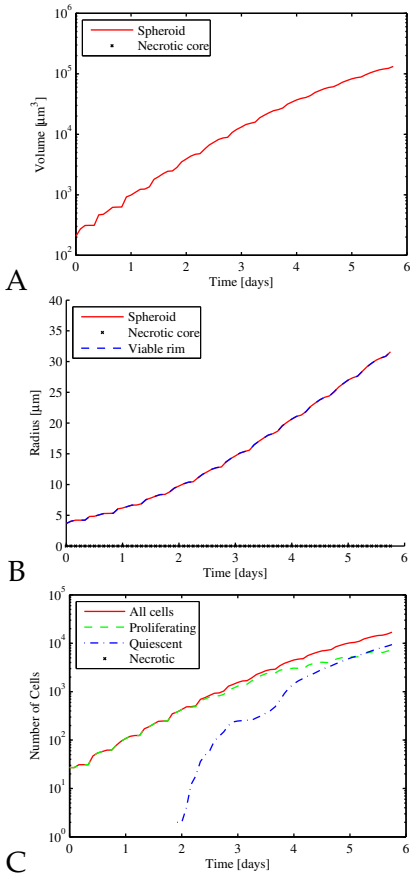

Figure 4.5: Growth curves for line vasculature with delay. (A) Log plot of tumor volume (cm^3) vs. time (days) of tumor grown. (B) Tumor radius (cm) vs. time (days). (C) Log plot of number of cells vs. time (days) of a tumor grown with a line of vasculature added through the center along the x axis at the beginning of day 3. Red represents total number of cells, green represents number of proliferating cells, blue represents the number of quiescent cells and there are no necrotic cells.

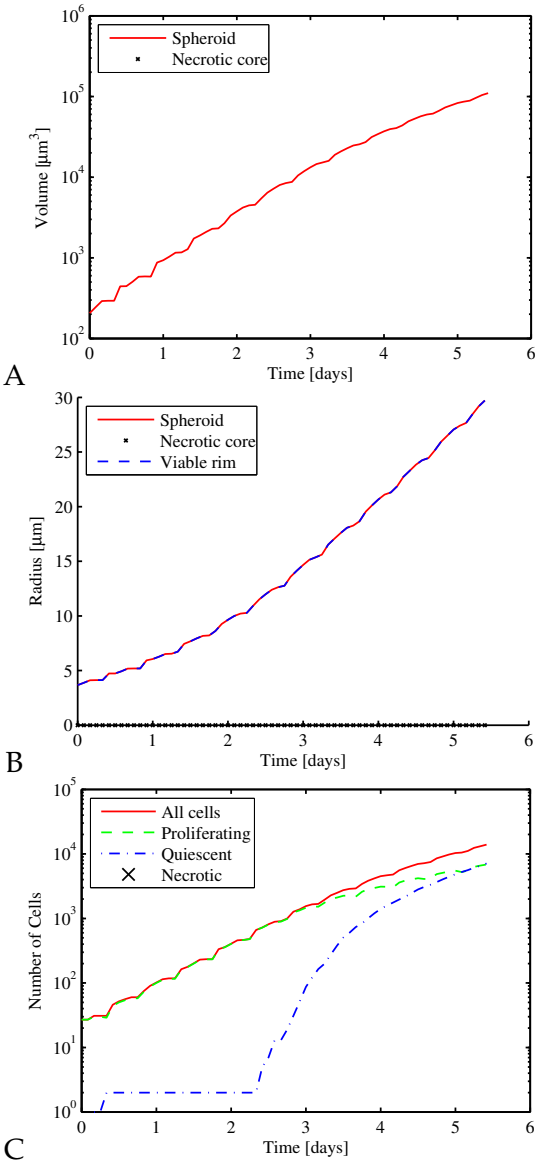

Figure 4.6: Growth curves for square grid vasculature without delay. (A) Log plot of tumor volume (cm^3) vs. time (days) of tumor grown. (B) Tumor radius (cm) vs. time (days). (C) Log plot of number of cells vs. time (days) of a tumor grown with a square vasculature through the entire grid at the beginning. Red represents total number of cells, green represents number of proliferating cells, blue represents the number of quiescent cells and there are no necrotic cells.

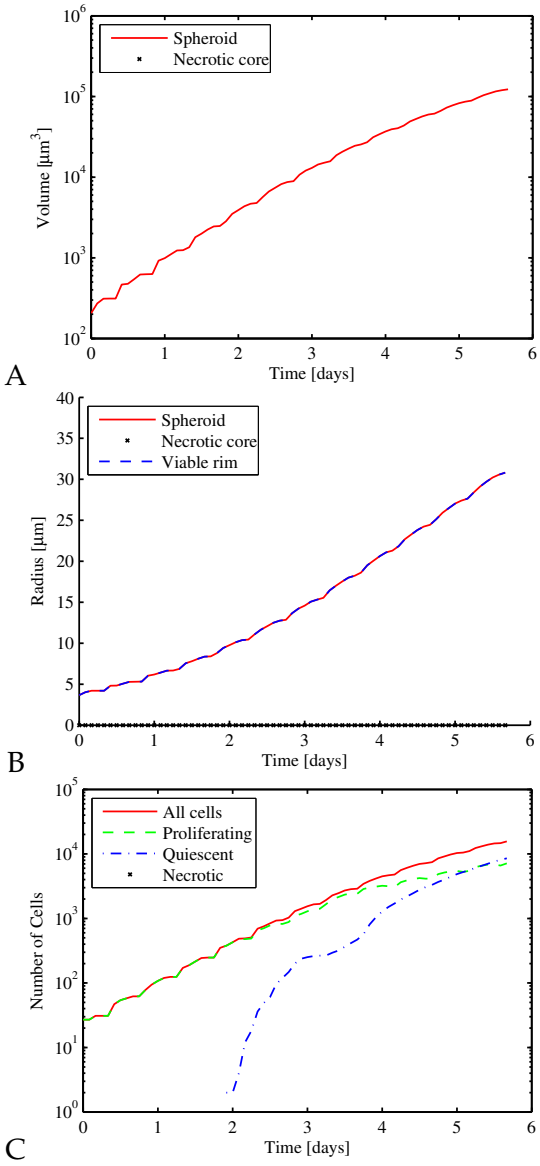

Figure 4.7: Growth curves for square grid vasculature with delay. (A) Log plot of tumor volume (cm^3) vs. time (days) of tumor grown. (B) Tumor radius (cm) vs. time (days). (C) Log plot of number of cells vs. time (days) of a tumor grown with square vasculature added through the entire grid at the beginning of day 3. Red represents total number of cells, green represents number of proliferating cells, blue represents the number of quiescent cells and there are no necrotic cells.

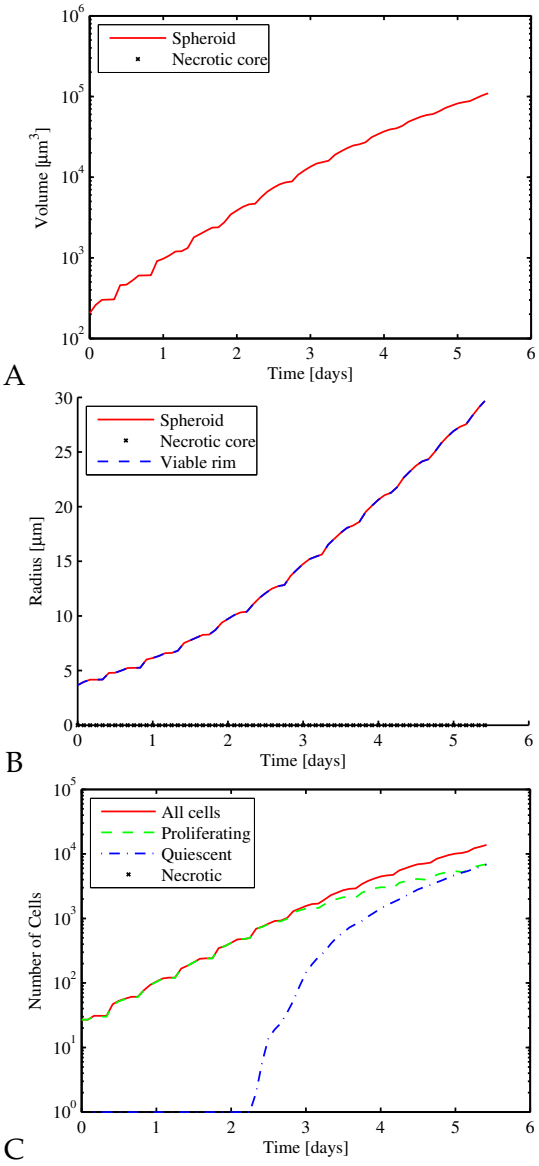

Figure 4.8: Growth curves for hexagonal grid vasculature without delay. (A) Log plot of tumor volume (cm^3) vs. time (days) of tumor grown. (B) Tumor radius (cm) vs. time (days). (C) Log plot of number of cells vs. time (days) of a tumor grown with a hexagonal grid vasculature added through the entire grid at the beginning. Red represents total number of cells, green represents number of proliferating cells, blue represents the number of quiescent cells and there are no necrotic cells.

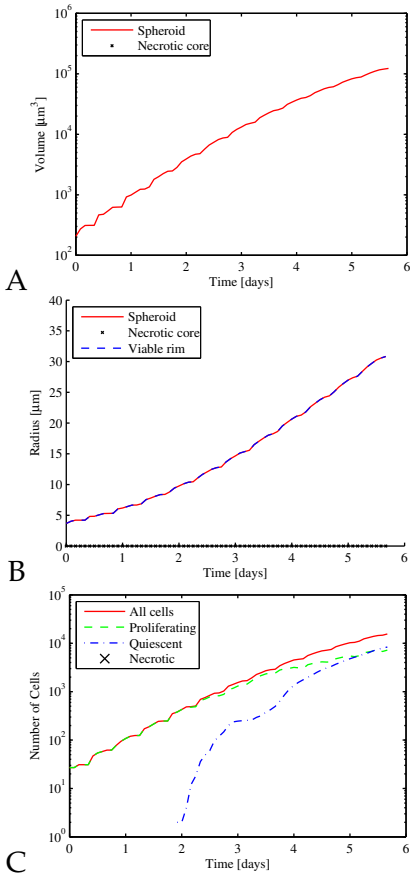

Figure 4.9: Growth curves for hexagonal grid vasculature with delay. (A) Log plot of tumor volume (cm^3) vs. time (days) of tumor grown. (B) Tumor radius (cm) vs. time (days). (C) Log plot of number of cells vs. time (days) of a tumor grown with a hexagonal grid vasculature added through the entire grid at the beginning of day 3. Red represents total number of cells, green represents number of proliferating cells, blue represents the number of quiescent cells and there are no necrotic cells.

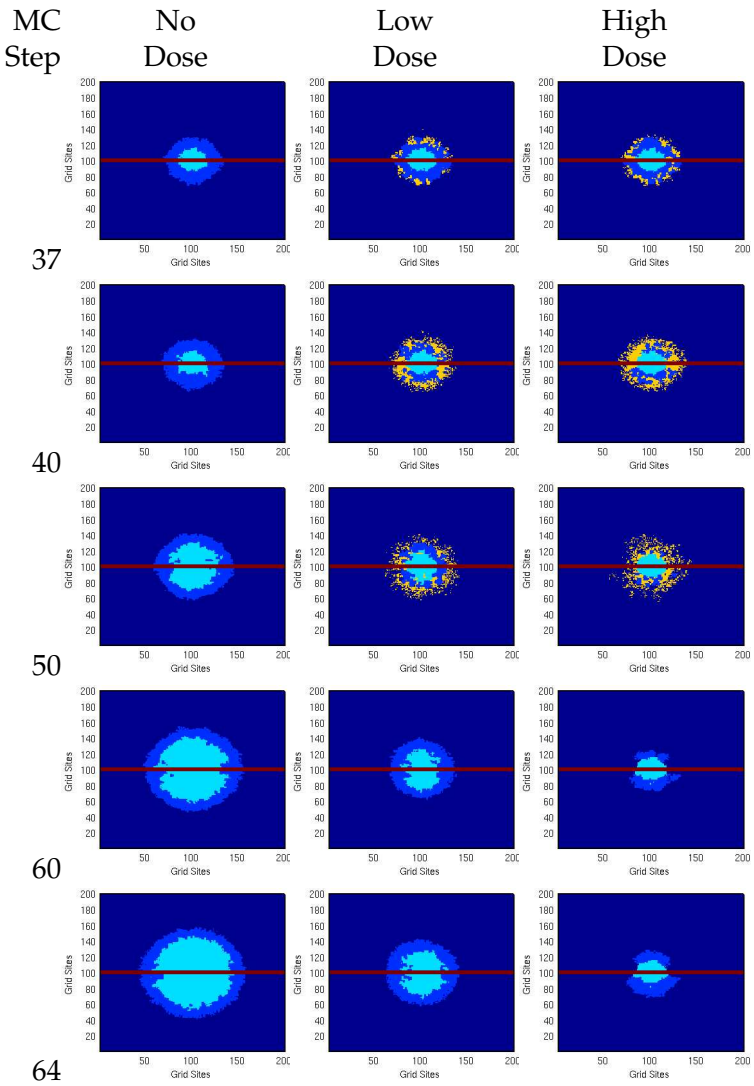

Figure 4.10: Cell types of tumor with line vasculature with delay at time steps 37 (A-C), 40 (D-F), 50 (G-I), 60 (J-L), and 64 (M-O). No chemotherapy was added for (A,D,G,J,M), a low dose of chemotherapy, 1.2g, was added for (B,E,H,K,N), and a high dose of chemotherapy, 2.0g, was added for (C,F,I,L,O). Images are an x-y cross section at a z elevation of 100 GRID SITEs. The length of one SITE is equivalent to 4.2E-4 cm or 4.2 microns. Dark blue represents medium, blue represents proliferating cells, light blue represents quiescent cells, yellow represents apoptotic cells, and red represents the vasculature.

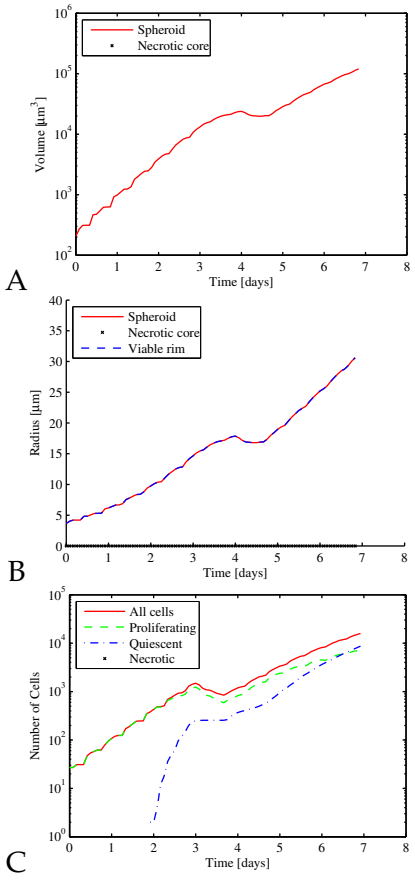

Figure 4.11: Growth curves for a delayed linear vasculature with a low dose of cyclophosphamide (1.2g). (A) Log plot of tumor volume (cm^3) vs. time (days) of tumor grown. (B) Tumor radius (cm) vs. time (days). (C) Log plot of number of cells vs. time (days) of a tumor grown with a line vasculature added through the entire grid at the beginning of day 3. The chemotherapy dose also started at the beginning of day 3. Red represents total number of cells, green represents number of proliferating cells, blue represents the number of quiescent cells and there are no necrotic cells.

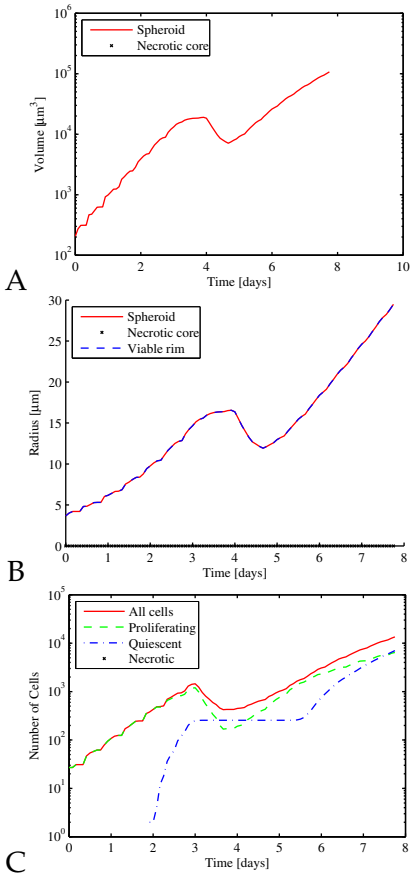

Figure 4.12: Growth curves for a delayed linear vasculature with a high dose of cyclophosphamide (2.0g). (A) Log plot of tumor volume (cm^3) vs. time (days) of tumor grown. (B) Tumor radius (cm) vs. time (days). (C) Log plot of number of cells vs. time (days) of a tumor grown with a line vasculature added through the entire grid at the beginning of day 3. The chemotherapy dose also started at the beginning of day 3. Red represents total number of cells, green represents number of proliferating cells, blue represents the number of quiescent cells and there are no necrotic cells.

MC	No Vascular Delay			Delayed Vascular			
Step	Linear	Grid	Hex	Linear	Grid	Hex	None
37	98815	166586	99179	97834	96828	97760	99019
40	126875	127714	127653	126457	126019	126654	126801
50	269653	273136	271508	273072	272866	273010	268202
60	552392	559235	551658	556439	558100	556730	543727
64	691866	701410	692846	700412	698885	697847	684429

Table 4.1: Volumes of tumors, measured in SITES, plotted at various time steps of interest. Vasculature is added during time step 36.

MC	No Vascular Delay			Delayed Vascular			
Step	Linear	Grid	Hex	Linear	Grid	Hex	None
37	57.35	57.22	57.43	57.16	56.97	57.16	57.39
40	62.34	62.48	62.47	62.27	62.20	62.31	62.33
50	80.15	80.49	80.33	80.49	80.47	80.49	80.01
60	101.80	102.21	101.75	102.04	102.15	102.07	101.26
64	109.73	110.23	109.78	110.18	110.10	110.05	109.33

Table 4.2: Diameters of tumors in millimeters, plotted at various time steps of interest. Vasculature is added during time step 36.

MC	No Vascular Delay			Delayed Vascular			
Step	Linear	Grid	Hex	Linear	Grid	Hex	None
37	1670	1658	1673	1667	1657	1664	1680
40	2282	2286	2293	2266	2275	2273	2259
50	4685	4788	4746	4805	4779	4791	4665
60	10125	10312	10127	10230	10313	10251	9936
64	12952	13138	12981	13066	13124	13065	12703

Table 4.3: Cell count of tumors, plotted at various time steps of interest. Vasculature is added during time step 36.

MC	No Vascular Delay			Delayed Vascular			
Step	Linear	Grid	Hex	Linear	Grid	Hex	None
37	1469	1537	1478	1414	1396	1413	1337
40	1845	2010	1887	1978	1980	1978	1702
50	2886	3029	2965	2928	3068	3030	2726
60	5307	5478	5394	5313	5442	5455	5078
64	6446	6585	6650	6440	6567	6652	6196

Table 4.4: Proliferating cell count of tumors, plotted at various time steps of interest. Vasculature is added during time step 36.

MC	No Vascular Delay			Delayed Vascular			
Step	Linear	Grid	Hex	Linear	Grid	Hex	None
37	201	121	195	253	261	251	343
40	437	276	406	288	295	295	557
50	1799	1759	1781	1877	1711	1761	1939
60	4818	4834	4733	4917	4871	4796	4858
64	6506	6553	6331	6626	6557	6413	6507

Table 4.5: Quiescent cell count of tumors, plotted at various time steps of interest. Vasculature is added during time step 36.

MC	Tumor	Tumor	Total	Proliferating	Quiescent
Step	Volume	Diameter	Cell Count	Cell Count	Cell Count
37	98168	57.23	1642	984	254
40	114711	60.28	1847	442	256
50	89175	55.43	1419	212	256
60	61757	49.04	1017	756	256
64	85534	54.67	1460	1204	256

Table 4.6: High chemotherapy dose information. The high dose is 2.0 grams, beginning on step 36, with concentration obeying the chemical profile produced by equation (3.18).

MC Step	Tumor Volume	Tumor Diameter	Total Cell Count	Proliferating Cell Count	Quiescent Cell Count
37	98275	57.26	1650	1125	253
40	118440	60.93	1970	814	256
50	142355	64.78	2240	886	405
60	192790	71.67	3364	2406	958
64	260225	79.21	4654	3121	1533

Table 4.7: Low chemotherapy dose information. The low dose is 1.2 grams, beginning on step 36, with concentration obeying the chemical profile produced by equation (3.18).

Chapter 5

Future Work

This Harvey Mudd Mathematics Clinic project has focued on developing one important component of the larger model being constructed by the team at LANL. As the year has progressed, the direction of the project has evolved. We have discovered through the wealth of literature on chemotherapeutic measures, myriad possibilities for future improvements upon the model. A few of these possibilities will be discussed in this section, some of which include reconsiderations of assumptions of the current model, while others involve changes only possible when considering the entire model at once. We also discuss the incorporation of additional variables into the current model. These variables could allow the model to more accurately simulate biological conditions. Lastly, different chemotherapeutic models are considered.

5.1 Modifying Basic Assumptions

One assumption to reconsider is the values of the diffusion constants of each modeled chemical. Currently the constants are set to equal values, although it seems feasible to create a variable diffusion constant that is a function of the chemical as well as of the cell type in which it is diffusing. Making the diffusion constants functions of the chemicals under consideration would involve calculations based upon properties of the chemicals, such as molecular weight and dipole moment. Also, it should be kept in mind that chemotherapeutic agents diffuse and convect at different rates depending on the permeability of the vasculature (Jang et al. [2003]) The effect of changing these values is unknown to us, although on a fundamental level, the change would be more biologically accurate, because this

variability is ignored by the current model. However, it is also possible that the source and sink terms within the model will override any differences accounted for in the change of the diffusivity constants.

Another assumption that could be reconsidered is that of simplified blood flow. Recall that the chemical concentrations were given constant values to approximate an average over time. Although the blood flow works well with the non-vascularized, non-chemotherapeutic trials, the transport of chemicals within the blood vessels is one that can be modeled and might be integral to the correctness of the transport of the chemotherapeutic agents. Also, many of the experiments run by outside researchers consist of measures of plasma blood concentration levels versus toxicity or survival levels. Modeling the blood flow with an incorporation of the plasma drug concentration curves would be an interesting course of study that might elucidate properties of transport that could not be measured within a laboratory.

5.2 Restructuring for Additional Treatment Types

With the implementation of chemotherapy, some of the basic structures of the model need to be re-evaluated for a more accurate response to chemotherapy *in vivo*. In the case of the necrotic cells, a major assumption from the original code was that upon cell death, the cell remains fixed to the grid instead of being carried away. This assumption, while perfectly valid in spheroids which contain concentric rings of cell types which do not shed, is of limited value in a vascularized chemotherapy model. Biologically, tumors may shrink as cells are lysed and carried away, although the current model has no such shrinking capability. Futhermore, in many cases, chemotherapeutic drugs induce apoptosis rather than necrosis. While "necrosis is the death of cells or tissues due to chemical or physical injury," apoptosis is "biologically programmed cell death. Necrosis leaves extensive cellular debris that need to be removed by phagocytes, whereas apoptosis does not" (Jr. et al. [2005]). Because there is quicker transport in interstitial spaces, the type of cell death (in which the induction is dependent upon type of drug as well as strength and regimen of drug) is quite important in modeling the transport of the chemotherapeutical agents. Either change would allow more accurate measures to validate our model.

An additional step in the implementation of chemotherapy would be creating a general structure in which different kinds of chemotherapeutic agents (whether of one type or multiple) could be modeled within the sys-

tem. This would not only allow the model to respond to different types of chemotherapy, but also allow chemotherapy *cocktail* treatments to be modeled. Combination chemotherapy is a very common treatment approach, so it is important to be able to simulate the results of using this method. The code we have written is structured to be flexible and to allow for additional chemotherapy types to be added if the appropriate parameters are known. New chemicals can diffuse and be consumed by cells in the same way that other chemicals are modeled. Additionally, depending on the type of chemotherapy, a more complicated cellular reaction could be modeled using components of the protein expression code.

5.3 Alternate Models

Our current chemotherapy model only handles chemotherapeutic agents which do not depend upon cells being in a particular phase of the cell cycle. An addition could be made to allow for cycle-specific drug treatments. Five main criteria have been identified to characterize a drug's pharmacokinetics. These are the route of administration, dose administered, dosing interval, plasma drug concentrations, and collection times relative to drug administration (Undevia et al. [2005]). Our model only considers the dose administered and the dosing interval. In order to account for the plasma drug concentrations, the model should utilize the common area under the curve (AUC) measurement in which plasma concentration levels are plotted against time. Also, the ratio of the dose administered to the AUC can be used to measure the body's ability to remove the drug (Undevia et al. [2005]). Interestingly, the route of administration has been shown to affect the coefficient of variation. This coefficient for oral versus intravenous administration has been shown to vary between 28% and 58%, respectively (Undevia et al. [2005], Bardelmeijer et al. [2000], El-Kareh and Secomb [2000]).

There already exist several models that simulate in various ways the drug pharmacokinetics, e.g. Parker and III [2001], Gallo et al. [2004], Mori et al. [2002], and Zhang et al. [2001]. It should be possible to incorporate some of these into our current model as well.

5.4 Further Considerations

Of the five main variables that influence drug uptake, each can be influenced by other factors, such as age, gender, ethnicity/race, etc. It has

been determined that the differences in pharmacokinetics (drug uptake, metabolism) cannot simply be extrapolated from studies conducted with adults (Groninger et al. [2004]). Furthermore, there is a possible difference in toxicity levels depending upon the age of the adult (Yancik [1997]). Race and ethnicity have also shown signs of being factors in the pharmacodynamics and pharmacokinetics of a drug (Johnson [1997]).

Chapter 6

Conclusion

Because cancer is the second leading cause of death in the United States, this disease has obvious societal impacts, including those that affect our economy and our health care system. Cancer treatments vary significantly in their levels of success. Common types of treatment, including chemotherapy and radiation therapy, destroy healthy cells along with cancerous cells. Because of this, cancer treatment is an extremely painful and risky endeavor. By improving mathematical tumor models and studying cancer treatment, it is possible to more fully understand these treatments, and, in turn, improve them.

Although a realistic cancer model may not directly lead to a cancer cure, it will elucidate our understanding of cancer and its treatments. Consequently, research involving tumor growth models has the potential to affect the quality of life for millions of people each year, if it can assist in developing better cancer treatments and permit the assessment of the effectiveness of current treatments.

This paper has presented an overview of the HMC clinic team's efforts to expand and enhance a tumor model developed at LANL. The simulation contains three models within it: a subcellular model governing the diffusion of chemicals through the tumor environment, a cellular model governing the reproduction of cells, and an extracellular Monte Carlo model governing the interaction and growth of cells.

The inclusion of a blood vessel system in a previously avascular model required many changes to the existing code. To do this correctly, the team carried out extensive research into how this particular model functions and how tumors develop. The team then made changes to the model to include vasculature. Each section of the model required some modifications. The

team made background changes to allow for blood vessel cell types to be represented, altered the diffusion calculations to account for the presence of vasculature, modified the Monte Carlo component so that the vasculature would not be encroached upon by growing cells, and determined that the protein expression step should not be affected by the vasculature. The team streamlined the code for efficiency, improving the run-time. This was achieved by converting Cell ID numbers to pointers as well as optimizing memory use and processing time.

We have added to the model the ability to simulate treatments using non-cell cycle specific chemotherapy drugs. In particular, the model now simulates treatments using cyclophosphamide, which alkylates the DNA of proliferating tumor cells. By doing so, it induces apoptosis, which results in tumor shrinkage immediately following the administration of the dose. In the experiments that we ran, higher doses of chemotherapy better served to inhibit tumor regrowth.

However, in considering the results of our experiments, it is important to keep in mind that due to hardware limitations, we were unable to run experiments where the tumor reached a detectable size before the first dose of chemotherapy was administered. Furthermore, the time allowed to elapse between chemotherapy doses was shorter than is typical in medical practice (Gurney [2002]). This adjustment was required because of hardware limitations, but can be relaxed in future experiments using this model.

We have adapted the code provided by LANL so that a vascularized tumor can be modeled. We have also laid groundwork for including chemotherapy treatment in simulations, and run preliminary chemotherapy experiments. As such, we hereby provide to LANL all of the deliverables specified in the work statement for this project.

Appendix A

Source Code

The following files are given as example source code in the simulation. The clinic team has added support for a vasculature into the CELL class and changed the cell ID's to pointers in order to improve efficiency and maintainability. The code has also been optimized for memory usage to reduce the problem of thrashing with large data sets. Thrashing occurs when the computer spends most of its time moving large amounts of data into and out of RAM, rather than actually manipulating the data and running the simulation.

`Cell.h` and `Cell.cxx` implement the CELL class. Each CELL is either proliferating, quiescent, necrotic, or part of the vasculature. Each proliferating CELL has a series of proteins that are either expressed or not expressed at each step in the simulation, and these proteins determine whether the CELL divides or becomes quiescent. Each cell also contains a pointer to its parent cell, and the concentrations of oxygen, nutrients, waste, growth factor, and inhibition factor within it.

`Site.h` implements the SITEs that make up the GRID. Each site has concentrations of the five chemicals in them (oxygen, nutrients, etc.), as well as a pointer to the cell in which it currently resides.

A.1 TumorSlice.m

```
%% Tiffany Head HMC mathematics clinic Sp2006
%% adapted from matlab scripts provided by LANL
%% This function creates 2d slices of the tumor
%% which show concentrations of chemicals or the cell types
```

```
%% be sure to unzip all .gz files before begining

function TumorSlice(prefix,time,chem,xyz,depthPer,con)

% prefix is the file prefix found on all output files

% time is the cycle number at which the tumor snapshot has been taken
%           note, this should be a 3 digit string imput

% chem is either the chemical being profiled or 'cells' which
%           will display the different cell types
%           cells will display cell types
%           gf will display growth factor
%           02 will display oxygen
%           w will display waste
%           n will dsiplay nutrients
%           if will display inhibitory factor

% xyz is either an 'x', 'y' or x depending on the desired
%           cross-section

% depthPer is the percentage in the x y or z direction which describes the
%           depth or the cross-section

% con is true if contours of cell types are to be added in the chemical
            profile plots and false if not


if (strcmp(chem,'cells')==1 )
  filename=[prefix,'Types',time];
else
  filename=[prefix,'Chem',time,'C',chem];
  if(con)
    filename2=[prefix,'Types',time];
  end
end

load(filename);

tumor=eval(filename);
```

```
len=length(tumor(1,:));
depth=round(depthPer/100*len);

if (xyz == 'z')
for(i=1:len)
  A=tumor((i-1)*len+1:len*i,depth);
  tumor2d(:,i)=A;
end
elseif (xyz == 'y')
  for(i=1:len)
    tumor2d(i,:)=tumor((i-1)*len+depth,:);
  end
else
  tumor2d=tumor((depth-1)*len+1:depth*len,:);
end

if (strcmp(chem,'cells')==0 && con)

  load(filename2);

  tumor2=eval(filename2);

  len2=length(tumor2(1,:));
depth2=round(depthPer/100*len2);

if (xyz == 'z')
for(i=1:len2)caxis([cmin cmax])
  A=tumor2((i-1)*len2+1:len2*i,depth2);
  tumorCon(:,i)=A;
end
elseif (xyz == 'y')
  for(i=1:len2)
    tumorCon(i,:)=tumor2((i-1)*len2+depth2,:);
  end
else
  tumorCon=tumor2((depth2-1)*len2+1:depth2*len2,:);
end

end
```

```
length(tumor2d);

figure;
pcolor(tumor2d); shading flat; caxis([-1 5]) ;
xlabel('Grid Sites');
ylabel('Grid Sites');
hold on;
if (strcmp(chem,'cells')==0 && con)

  colorbar;
  thing=contourc(tumorCon,5);
  left=1;
  while(left>0)
    numpts=thing(2,1);
    plot(thing(1,2:numpts+1)+.5,thing(2,2:numpts+1)+.5,'k');
    hold on;
    thing=thing(:,numpts+2:end);
    left=length(thing);
    hold on;
  end

  %figure;
  %contour(tumorCon,5);
  %hold on;

  figure;
  surf(tumor2d);
end


%figure;
%pcolor(tumor2d); shading flat; cax = caxis; colorbar;
```

A.2 TumorPassthrough.m

```
%% Tiffany Head HMC mathematics clinic Sp2006
```

```
%% adapted from matlab scripts provided by LANL
%% This function creates a serries of 2d slices of the tumor
%% which show concentrations of chemicals or the cell types in a movie
%% which passes through the tumor on the axis specified

%% be sure to unzip all .gz files before begining


function [M,moviefile]=TumorPassthrough(prefix,time,chem,xyz)

% prefix is the file prefix found on all output files

% time is the cycle number at which the tumor snapshot has been taken
%           note, this should be a 3 digit string imput

% chem is either the chemical being profiled or 'cells' which
%           will display the different cell types
%           cells will display cell types
%           gf will display growth factor
%           02 will display oxygen
%           w will display waste
%           n will dsiplay nutrients
%           if will display inhibitory factor

% xyz is either an 'x', 'y' or x depending on the desired
%           cross-section

%% This section loads the file
filename='';
chem;

if (strcmp(chem,'cells')==1 )
    filename=[prefix,'Types',time];
else
    filename=[prefix,'Chem',time,'C',chem];
end

load(filename);

tumor=eval(filename);
```

```
len=length(tumor(1,:));

% This determines the max and min vaules in the data so that the
% colors can normalized over all cross-section plots

cmax=max(max(tumor));
cmin=min(min(tumor));


% This for loop creates the slices of tumor and then captures them
% into a movie

for (j=1:len)
    depth=j;

    if (xyz == 'z')
        for(i=1:len)
            A=tumor((i-1)*len+1:len*i,depth);
            tumor2d(:,i)=A;
        end
    elseif (xyz == 'y')
            for(i=1:len)
                tumor2d(i,:)=tumor((i-1)*len+depth,:);
            end
    else
            tumor2d=tumor((depth-1)*len+1:depth*len,:);
    end

    pcolor(tumor2d); shading flat; caxis([cmin cmax]); colorbar;

    M(j) = getframe;
end

% Plays the moive

movie(M);

% Saves the movie as MPG if MPGWRITE is installed

%moviefile=['PassThrough','_',chem,'_',xyz,'_',time,'.mpg'];
```

```
%mpgwrite(M,hsv,moviefile);
```

A.3 TumorGrowthMovie.m

```
%% Tiffany Head HMC mathematics clinic Sp2006
%% adapted from matlab scripts provided by LANL
%% This function creates 2d slices of the tumor
%% which show concentrations of chemicals or the cell types
%% taken over a time interval and then shown in a movie

%% be sure to unzip all .gz files before begining

function [M,moviefile]=TumorGrowthMovie(prefix,timevec,chem,xyz,depth)


% prefix is the file prefix found on all output files

% timevec is a vector of numbers of cycles at which each movie frame
%           is taken. Example: [1,3,5,7,9] will produce a movie which
%           displays slices of the tumor after 1, 3, 5, 7 and 9
%           cycles.

% chem is either the chemical being profiled or 'cells' which
%           will display the different cell types
%           'cells' will display cell types
%           'gf' will display growth factor
%           '02' will display oxygen
%           'w' will display waste
%           'n' will dsiplay nutrients
%           'if' will display inhibitory factor

% xyz is either an 'x', 'y' or 'z' depending on the desired
%           cross-section

% depth is the percentage in the x y or z direction which describes the
%           depth or the cross-section
```

```
tlen=length(timevec);

% This loop collects all of the slices into one large matrix

for (j=1:tlen)
    if (timevec(j)>99)
        time=num2str(timevec(j));
    elseif (timevec(j)>9)
        time=['0',num2str(timevec(j))];
    else
        time=['00',num2str(timevec(j))];
    end

    if (strcmp(chem,'cells')==1 )
        filename=[prefix,'Types',time];
    else
        filename=[prefix,'Chem',time,'C',chem];
    end

    load(filename);
    (j/tlen)*100

    tumor=eval(filename);
    len=length(tumor(1,:));
    if(j==1)
        depth=round(depth/100*len);
    end

    if (xyz == 'z')
        for(i=1:len)
            A=tumor((i-1)*len+1:len*i,depth);
            tumor2d(:,i)=A;
        end
    elseif (xyz == 'y')
            for(i=1:len)
                tumor2d(i,:)=tumor((i-1)*len+depth,:);
            end
    else
            tumor2d=tumor((depth-1)*len+1:depth*len,:);
    end
```

```
    tumormov((j-1)*len+1:j*len,:)=tumor2d;
    clear(filename);
end

% This determines the max and min vaules in the data so that the
% colors can normalized over all cross-section plots

cmax=max(max(tumormov));
cmin=min(min(tumormov));

% This loop re-separates the large matrix into separate snapshots so
% and then captures them into a movie.

if(cmax==cmin)
    M=0;
    disp('No data to make movie');
    return;
else

    for (j=1:tlen)

        tumor2d=tumormov((j-1)*len+1:j*len,:);

        pcolor(tumor2d); shading flat; caxis([cmin cmax]); colorbar;

        M(j) = getframe;
    end
end

% Plays the moive

movie(M,2);

% Saves the movie as MPG if MPGWRITE is installed

st=num2str(timevec(1));
en=num2str(max(timevec));
dep=num2str(depth);
moviefile=['Growth_',chem,'_',xyz,dep,'_',st,'to',en,'.mpg'];
```

```
%disp('writing...')
%mpgwrite(M,jet,moviefile);
%disp('done')
```

Appendix B

Glossary

During the literature search, the team composed this list of commonly used biological vocabulary found in many of the research papers.

anastomosis: The surgical connection of separate or severed tubular hollow organs to form a continuous channel, as between two parts of the intestine. Source: dictionary.com

angiogenesis: The formation of blood cells from a preexisting vasculature.

annulus: A ringlike figure, part, structure, or marking, such as a growth ring on the scale of a fish. Source: dictionary.com

apoptosis: When cells that are superfluous or that could harm the individual undergo a type of ”cell suicide”. Leads to the death of cells that are no longer required in adult (Karp [2002]).

carcinogenesis: The production of cancer. Source: dictionary.com

chemotaxis: The characteristic movement or orientation of an organism or cell along a chemical concentration gradient either toward or away from the chemical stimulus. Source: dictionary.com

desmosomes: disk-shaped adhesive junctions 1 (mu)m in diam (Karp [2002]).

epithelial: Of or relating to Membranous tissue composed of one or more layers of cells separated by very little intercellular substance and forming the covering of most internal and external surfaces of the body and its organs. Source: dictionary.com

embryogenesis: The development and growth of an embryo. Source: dictionary.com

endogenous: Produced or growing from within. Source: dictionary.com

endothelial: A thin layer of flat epithelial cells that lines serous cavities, lymph vessels, and blood vessels. Source: dictionary.com

extracellular: Outside the plasma membrane (the functional barrier between the inside and outside of a cell) (Purves et al. [2001]).

extracellular matrix: plant cell wall for animals. Composed of fibrous proteins. Holds things together, lets materials pass through, etc. (Purves et al. [2001]).

metastasis: The spread of a tumor within the body (Karp [2002]).

metastatic cells: Cancer cells that are able to initiate the formation of secondary tumors.

oncogenes: Genes implicated in carcinogenesis, encode proteins that promote the loss of growth control and the conversion of a cell to a malignant state (Karp [2002]).

Bibliography

ACS. Acs webpage. 2006. URL `http://www.cancer.org`.

ACS. Cancer facts & figures 2004. 2004. URL `http://www.cdc.gov/node.do/id/0900f3ec80193c0d`.

T. Alarcon, H.M. Byrne, and P.K. Maini. A cellular automaton model for tumour growth in inhomogeneous environment. *Journal of Theoretical Biology*, 225:257–274(18), 2003.

T. Alarcon, H.M. Byrne, and P.K. Maini. Towards whole-organ modelling of tumour growth. *Progress in Biophysics and Molecular Biology*, 85:451–472, 2004.

T. Alarcon, H.M. Byrne, and P.K. Maini. A multiple scale model for tumor growth. *Multiscale Model Simul.*, 3:440–475(2), 2005.

Sharyn D. Baker, Jaap Verweij, Eric K. Rowinsky, Ross C. Donehower, Jan H.M. Schellens, Louise B. Grochow, and Alex Sparreboom. Role of body surface area in dosing of investigational anticancer agents in adults 1991-2001. 94:1883–1888(24), 2002.

Heleen A. Bardelmeijer, Olaf van Tellingen, Jan H.M. Schellens, and Jos H. Beijnen. The oral route for the administration of cytotoxic drugs: strategies to increase the efficiency and consistency of drug delivery. *Investigational New Drugs*, 18:231–241, 2000.

Elaine Y.L. Blair, Laurent P. Rivory, Stephen J Clarke, and Andrew J. McLachlan. Population pharmacokinetics of raltitrexed in patients with advanced solid tumours. *British Journal of Clinical Pharmacology*, 57:416–426(4), 2004.

H.M. Byrne, C.J. W. Breward, and C.E. Lewis. A multiphase model describing vascular tumour growth. *Bulletin of Mathematical Biology*, 65: 609–640(4), 2003.

Ian W. Campbell and Soon Song. Blood glucose levels. In *http://www.netdoctor.co.uk/health*, 2005.

C.Y. Chen and H.M. Byrne. The influence of growth-induced stress from the surrounding medium on the development of multicell spheroids. *Journal of Mathematical Biology*, 43(3):191–220, 2001.

L.G. de Pillis and D.G. Mallet. A cellular automata model of tumor-immune system interactions. *Journal of Theoretical Biology*, 239:334–350(3), 2006.

Vincent T. Devita, Samuel Hellman, and Steven A. Rosenberg. *Cancer: Principles and Practice of Oncology*. Lippincott, Williams, & Wilkins, 2004.

A.W. El-Kareh and T.W. Secomb. A mathematical model for comparison of bolus injection, continuous infusion, and liposomal delivery of doxorubicin to tumor cells. *Neoplasia*, 2:325–338(4), 2000.

S.C. Ferreira, M.L. Martins, and M.J. Vilela. Morphology transitions induced by chemotherapy in carcinomas in situ. *Physical Review*, 67, 2003.

Z.G. Forbes, B.B. Yellen, K.A. Barbee, and G. Friedman. An approach to targeted drug delivery based on uniform magnetic fields. 39:3372–3377(5), 2003.

James M. Gallo, Paolo Vicini, Amy Orlansky, Shaolan Li, Feng Zhou, Jianguo Ma, Sharon Pulfer, Michel A. Bookman, and Ping Guo. Pharmacokinetic model-predicted anticancer drug concentrations in human tumors. *Clinical Cancer Research*, 10:8048–8058, 2004.

Amy J. Galpin and William E. Evans. Therapeutic drug monitoring in cancer management. 39:2419–2430(11), 1993.

D.J. Gavaghan, J.M. Brady, C.P. Behrenbruch, R.P. Highnam, and P.K. Maini. Breast cancer: Modelling and detection. *Journal of Theoretical Medicine*, 4:3 – 20, 2002.

K Groebe and W Mueller-Klieser. On the relation between size of necrosis and diameter of tumor spheroids. *International Journal of Radiation Oncology Biology Physics*, 34 (2):395–401, 1996.

E. Groninger, J.H. Proost, and S.S.N. de Graaf. Pharmacokinetic studies in children with cancer. *Critical Reviews in Oncology/Hematology*, 52:173–197, 2004.

H. Gurney. How to calculate the dose of chemotherapy. *British Journal of Cancer*, 86:1297–1302(86), 2002.

Yuen Yi Hon and William E. Evans. Making tdm work to optimize cancer chemotherapy: a multidisciplinary team approach. *Clinical Chemistry*, 44: 388–400(2), 1998.

T.L. Jackson and H.M. Byrne. A mathematical model to study the effects of drug resistance and vasculature on the response of solid tumors to chemotherapy. *Mathematical Biosciences*, 164(1):17–38, 2000.

R.K. Jain. Delivery of molecular medicine to solid tumors: lessons from in vivo imaging of gene expression and function. *Journal of Controlled Release*, 74:7–25, 2001.

Seong Hoon Jang, M. Guillaume Wientjes, Dan Lu, and Jessie L.-S. Au. Drug delivery and transport to solid tumors. *Pharmaceutical Research*, 20: 1337–1350(9), 2003.

Y. Jiang, J. Pjesivac-Grbovic, C. Cantrell, and J. Freyer. A multiscale model for avascular tumor growth. *Biophysical Journal*, 2005.

Julie A. Johnson. Influence of race or ethnicity on pharmacokinetics of drugs. 86:(12), 1997.

Charles A. Janeway Jr., Paul Travers, Mark Walpart, and Mark J Shlomchik. *Immunobiology, 6th. Edition*. Garland Science Publishing, 2005.

Gerald Karp. *Cell and Molecular Biology, 3rd Edition*. John Wiley and Sons, Inc., 2002.

J.E. Klaunig and L.M. Kamendulis. The role of oxidative stress in carcinogenesis. *Annual Review of Pharmacology and Toxicology*, 44:239–267, 2004.

M.A. Konerding, E. Fait, and A. Gaumann. 3d microvascular architecture of pre-cancerous lesions and invasive carcinomas of the colon. 84:1354–1362(10), 2001.

Kyoto. Kyoto encyclopedia of genes and genomes. 2005. URL `http://www.kegg.com`.

Mathworks. Matlab, v. 7.1.0.183. 2006. URL `www.mathworks.com`.

S.R. McDougall, A.R.A. Anderson, and M.A.J. Chaplain. Mathematical modelling of flow through vascular networks: Implications for tumour-induced angiogenesis and chemotherapy strategies. *Bulletin of Mathematical Biology*, 64:673–702, 2004.

Howard L. McLeod. Therapeutic drug monitoring opportunities in cancer therapy. *Pharmacol. Theor.*, 74:39–54(1), 1997.

Takeshi Mori, Masafumi Ohnishi, Megumi Komiyama, Arisa Tsutsui, Hiromitsu Yabushita, and Hiroshi Okada. Prediction of cell kill kinetics of anticancer agents using the collagen el droplet embedded-culture drug sensitivity test. *Oncology Reports*, 9:301–305, 2002.

W. Mueller-Klieser. Tumor biology and experimental therapeutics. *Critical Reviews in Oncology/Hematology*, 36:123–139, 2000.

Jan Nissl. Arterial blood gases. 2004. URL `http://www.webmd.com/hw/lab\_tests/hw2343.asp`.

Robert T. Osteen, Ted S. Gansler, and Raymond E. Jr. Lenhard. *Clinical Oncology*. American Cancer Society, 2001.

Robert S. Parker and Francis J. Doyle III. Control-relevant modeling in drug delivery. *Advanced Drug Delivery Reviews*, 48:211–228, 2001.

Pennsylvania State University. Biology of cultured cells. 2005. URL `http://www.bmb.psu.edu/courses/biotc489/notes/biointeract.htm`.

William Purves, David Sadava, Gordon Orians, and H. Craig Heller. *Life*. Sinauer Associates, Inc., 2001.

A.S. Qi, X. Zheng, C.Y. Du, and B.S. An. A cellular automaton model of cancerous growth. *Journal of Theoretical Biology*, 161(1):1–12, 1993.

B. Ribba, K. Marron, and Z. Agur. A mathematical model of doxorubicin treatment efficacy on non- hodgkins lymphoma:investigation of current protocol through theoretical modeling results. *Bull Math Biol*, 2005.

Yousef Saad. *Iterative Methods for Sparse Linear Systems*. PWS Publishing Company, 1996.

M. Sarntinoranont, F. Rooney, and M. Ferrari. Interstitial stress and fluid pressure within a growing tumor. *Annals of Biomedical Engineering*, 31(3): 327–335, 2003.

J.A. Sherratt and M.A.J. Chaplain. A new mathematical model for avascular tumor growth. *Journal of Mathematical Biology*, 43(4):291–312, 2001.

Carolien H. Smorenburg, Alex Sparreboom, Marijke Botenbal, Gerrit Stoter, Kees Nooter, and Jaap Verweij. Randomized cross-over evaluation of body-surface area-based dosing versus flat-fixed dosing of paclitaxel. 21: 197–202(2), 2003.

Samir D. Undevia, Gonzalo Gomez-Abuin, and Mark J. Ratain. Pharmacokinetic variability of anticancer agents. *Nature*, pages 447–458, 2005.

Rosemary Yancik. Cancer burden in the ages. 1997.

Liping Zhang, Richard Price, Francesca Aweeka, S. Eralp Bellibas, and Lewis B. Sheiner. Making the most of sparse clinical data by using a predictive model-based analysis, illustrated with a stavudine pharmacokinetic study. *European Journal of Pharmaceutical Science*, 12:377–385(4), 2001.