8-11-2016

Incorporating Exponential Functions into an Optimal Control Model for a Chronic Wound

Peter Agaba
Western Kentucky University, peter.agaba003@topper.wku.edu

INCORPORATING EXPONENTIAL FUNCTIONS INTO AN OPTIMAL CONTROL MODEL FOR A CHRONIC WOUND

A Capstone Experience/Thesis Project

Presented in Partial Fulfillment of the Requirements for

the Degree Bachelor of Mathematics with

Honors College Graduate Distinction at Western Kentucky University

By:

Peter Agaba
*****

Western Kentucky University
2016

CE/T Committee:

Dr. Schugart, Advisor

Dr. Richmond

Dr. Dunkum

Approved by

____
Advisor
Department of Mathematics

ABSTRACT

A chronic wound is a wound that does not heal in an orderly manner and on time. In this project, we simulate different ways of minimizing the time of therapy using exponential functions. The analysis in this research project focuses on treating chronic wounds using both mathematical and biological models. These models primarily focus on the amount of oxygen supplied to the wound using both hyperbaric and topical oxygen therapies. This amount should be optimal since too much oxygen is toxic to the body, and can potentially lead to death. The goal is to minimize the time spent in therapies since longer periods make treatments costlier. In this project, we incorporate exponential functions into several existing models of wound healing.

Keywords: Chronic wound, Gaussian function, negative exponential, topical and hyperbaric oxygen therapy, and optimal oxygen

Dedicated to

My mum, my family, Yohan Taremwa, and the Western Kentucky University Mathematics Department.

ACKNOWLEDGEMENTS

I would like to thank my advisor, Dr. Schugart for his guidance throughout my thesis project. Not only did he agree to work with me, he was also available whenever I needed his advice. Without his knowledge, guidance, and support, I would not have been able to successfully complete this thesis.

I would also like to thank Dr. Richmond and Dr. Dunkum for agreeing to serve in my thesis committee, and for giving me great insights whenever I went to see them with any question.

I am grateful to Dr. Kessler, Dr. Claus Ernst, and Dr. F. Atici for their support, not just when I was working on this thesis, but throughout my career at WKU's math department. Their support played a major role in my being able to complete my degree and graduate from Western Kentucky University.

I am also grateful to Dr. Lacretia Dye, WKU's math department, and Honors College for their support, mentorship and encouragement.

I would like to express my sincere gratitude to Dr. Brian Meredith and Ms. Toni Dye of the enrollment management office for making it possible for me to graduate from WKU. They provided me with scholarships that paid for all my tuition for my last year of undergraduate study at WKU.

Finally, I would like to thank my family and friends for supporting me in many ways. I particularly would like to thank Mr. and Mrs. Kimeu for giving me insightful ideas, and helping me edit my academic papers, including this thesis. My family also encouraged me, believed in me and provided any support whenever needed.

VITA

June 10, 1992 ....Born- Mbarara, Uganda

2012 ....Ntare School Mbarara, Uganda

2013 – 2016 .... Western Kentucky University Bowling Green, Kentucky

PRESENTATIONS

2015 ....45th Reach Week Conference Western Kentucky University
Bowling Green, Kentucky

2015 ....35th Math Annual Symposium Western Kentucky
University Bowling Green, Kentucky

2015 ....8th NIMBioS Conference University of Tennessee
Knoxville, Tennessee

2016 ....46th Reach Week Conference Western Kentucky University
Bowling Green, Kentucky

TABLE OF CONTENTS

LIST OF TABLES

Table

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Biology: Explanation of a Chronic Wound and Skin

A chronic wound is a wound which does not heal in an orderly set of stages and in a usual pattern of healing. Wounds that do not heal in a period of one month are normally considered chronic wounds. Chronic wounds normally develop from acute wounds, (wounds which heal in less than a month). The most common chronic wounds are ulcers. There are several causes of chronic wounds and these are due to lack of necessary components of wound healing which include a good supply of blood, oxygen, and nutrients. Also lack of a clean and infection-free environment may be another cause of chronic wounds (“Chronic Wounds,” 2016).

When a wound is formed on the skin, it normally goes through three main stages of wound healing and these are: inflammation, proliferation, and maturation (also known as remodeling). The first process known as inflammation is a natural response to trauma when a wound forms on the skin. It begins with homeostasis where blood vessels constrict and are sealed thus allowing platelets to create substances responsible for blood clotting. Once homeostasis is achieved, blood vessels then dilate again to allow the flow of nutrients and white blood cells that fight germs to the infection. At the end of this inflammation process, the skin experiences swelling, pain, heat, and redness (Broderick, 2009). The second process in wound healing process is proliferation. This is when the wound begins to rebuild and healthy granulation tissue is formed. The formation of granulation tissue needs sufficient oxygen and body nutrients. The new tissue is composed of extracellular matrix and collagen which are responsible for the development of network of blood vessels through a process called angiogenesis. Also, the body transforms damaged mesenchymal cells into fibroblasts which acts as a link to help in moving cells around the affected area. This normally happens three days after the injury formation when there is always secretion of liquids and collagen. This helps in strengthening the wound. During the process of proliferation, the wound grows stronger due to fibroblasts that help in development of new tissue that help in quickens the wound healing process (Brown *et al*., 2001). The last process in wound healing process is maturation (remodeling). Maturation occurs when the wound has closed up and this can take up to two years. In this phase of wound healing, the dermal tissues are repaired to increase the tensile strength of the tissues. At this stage, non-functional fibroblasts are replaced by new ones that function. The activities of cells abate and as a result the number of blood vessels in the wound reduces. The scar begins to form on the skin but it is still advisable to continue the treatment since at this stage only 80% of the affected part of the skin will have normalized (Brown *et al*., 2001)

There are different types of chronic wounds which result from different causes. Infectious wounds are due to bacteria, fungi or virus. Ischemic wounds come as a result of insufficient blood supply limiting the amount of nutrients and oxygen that is needed for the wound to heal. Radiation poisoning wounds are caused by too much exposure to radiation which weakens the immune system. Surgical wounds come as a result of incisions performed during surgeries. The other common chronic wounds are ulcers which can be classified as below.

- Arterial ulcers: These can occur from hypertension, atherosclerosis (plugging) and thrombosis (clotting), where the reduced blood supply leads to an ischemic state.

- Venous ulcers: These account for more than half of ulcer cases, especially in the lower limbs (mainly the legs) as associated with deep vein thrombosis, varicose veins and venous hypertension. Venous ulcers can lead to stasis, where the blood fails to circulate normally.

- Diabetic ulcers: These are a common complication in uncontrolled diabetes mellitus, resulting in impaired immune function, ischemia (due to poor blood circulation) and neuropathy (nerve damage), which eventually lead to breakage of skin and ulceration.

- Pressure ulcers: The constant pressure and friction resulting from body weight over a localized area for prolonged duration can lead to breakage of skin and ulceration (also known as bed sores); especially on the back and on the ankles and feet ("Chronic Wounds," 2016).

Chronic wounds can be identified through their symptoms and signs. These are bad odor, pus drainage, dead tissue, inflammation (fever, pain, redness, hotness, and swelling), and decrease in hair growth, vomiting, abdominal pain, blistering, skin thickening, itching, and weak pulse sensation of the body.

CHAPTER 2

Introduction: Mathematical Modeling

Over the past 20 years, different mathematical models have been developed for the treatment of chronic wounds. These mathematical models will focus on how much oxygen will be supplied to the wound. This amount should be the right amount since too much oxygen is toxic to the body and much oxygen can potentially kill patients. It will also focus on how to modify the model so to capture the significance of the length of the therapy. The goal is to minimize the time used in therapy since the longer period makes it costly.

With the use of Matlab, code can be written which solves a system of differential equations and integral functions (objective functional). The graphs that Matlab is able to generate can be analyzed for the results. The plots depict how the level of bacteria changes with neutrophils increase and at different times of administering oxygen to the therapy. The aforementioned process is called *optimal control* and is modeled by the following equations:

Equation for Bacteria

$$\frac{db}{dt} = k_b b(1-b) - b\,\frac{k_{nr}n + \delta}{\lambda_{rb}b + 1} * \frac{w}{w + k_w} - \lambda_b \tag{1}$$

Bacteria proliferation for logistic growth

Oxidative killing of bacteria

Natural death of bacteria

Equation for Neutrophils

$$\frac{dn}{dt} = k_p e^{-\lambda_p t}(1-n) + \frac{k_{ni}bn(1-n)g_{nw}(w)}{\lambda_{ni}n + 1} - \frac{\lambda_n n}{1 + eb} \tag{2}$$

Activation of neutrophils

Recruitment of neutrophils

Death of neutrophils

Supplemental Oxygen equation therapy scaled by Gaussian factor

$$\frac{dw}{dt} = \beta + \gamma * \sum_{i=1}^{P} e^{-\delta(t-\tau_i)^2}) * u(t) - \lambda_w w - \lambda_{bw}bw - \lambda_{nw}nw \tag{3}$$

Amount of oxygen from surrounding blood vessels

External input of oxygen scaled by gamma

Decay of oxygen

Uptake of oxygen by bacteria and inflammatory cells

Optimal Control Framework

Optimal control theory is used in making decisions regarding minimization or maximization. Given the variables, we can apply different techniques and test different variables to a control function and be able to come up with an optimal solution. The main goal is to minimize or maximize the objective function. This can be interpreted using Pontryagin's Maximum Principle given as Theorem 1.1 in Lenhart and Workman (2007). Pontryagin's Maximum Principle provides a set of necessary conditions that need to be satisfied for an optimal solution.

Theorem 1.1. *For the given control $\vec{u} = (u_1, \ldots., u_m)^\top$ belonging to the admissible control set U and related trajectory $\vec{x} = (x_1, \ldots., x_n)^\top$ that satisfies*

$\frac{d\vec{x}}{dt_i} = g_i(\vec{x}, \vec{u}, t)$ *(state equation)*

$\vec{x_i}$ *(a) =* c_i *(initial conditions)*

but with free end conditions, to minimize the performance criterion

$J = \phi(\vec{x}, t)|_a^b + \int_a^b f(\vec{x}, \vec{u}, t)dt$

it is necessary that a vector $\vec{\lambda} = \vec{\lambda}(t)$ exist such that

$\frac{d\vec{\lambda}i}{dt_i} = -\frac{\partial H}{\partial \vec{x}_i}$ *(adjoint equations)*

$\vec{\lambda}_i(b)$= $\phi x_i[\vec{x}(b), b]$ *(adjoint final conditions)* *where the Hamiltonian*

$H(t,\vec{x},u) = f(t,\vec{x},u) + \lambda^{\intercal} * \vec{g}(t,\vec{x},u),$ *= integrand + RHS of DE*

for all t, a ≤ t ≤ b, and all $\vec{u} \in U$ *, satisfies*

$H[\vec{\lambda}(t),\vec{x}^{*}(t),\vec{u}] \geq H[\vec{\lambda}(t),\vec{x}^{*}(t),\vec{u}^{*}],$ where $u*$ stands for optimal state of U.

Adjoint equations that are used in the equation above are like Lagrange multipliers because they add constraints to the variables being optimized (Daulton, 2013).

CHAPTER 3

Non-Linear Control Problem

Our objective functional for non-linear control is given by the equation below:

$$J(u) = \int_{t_0}^{t_1} [b(t) + (\textstyle\sum_{i=1}^{\tau} e^{-\delta(t-\tau_i)^2}) \frac{c}{2} u^2(t)]\, dt) \qquad (4)$$

where $0 \leq u \leq M_2$.

This models hyperbaric oxygen therapy. We consider a nonlinear function for the control u because it is unlikely that a body processes oxygen in a linear way (Daulton, 2013). We use equation (6) to see if there is a change in the length of therapies and to compare with the results that were obtained by Daulton. In equation (6), we use a sum of Gaussian functions to better simulate hyperbaric oxygen therapy.

When using equation (6) combined with differential equations (2) — (4) that were obtained from Daulton, we can form the Hamiltonian:

$$H{=}e^{-\delta t}(b{+}\frac{c}{2}u^2){+}\leftthreetimes_1\, (k_b b(1-b)\text{-}b\frac{k_{nr}n+\,\delta}{\leftthreetimes_{rb}b+1}\frac{w}{w+k_w} -\leftthreetimes_b\, b) \qquad (5)$$

$$+ \leftthreetimes_2\, (k_p e^{-\leftthreetimes_p t}(1-n) + \frac{k_{ni}bn(1-n)\big(g_{nw}(w)\big)}{\leftthreetimes_{ni}n+1} - \frac{\leftthreetimes_b n}{1+eb})$$

$$+\leftthreetimes_3\, (\beta + \gamma u(t) -\leftthreetimes_w\, w -\leftthreetimes_{bw}\, bw -\leftthreetimes_{nw}\, nw).$$

Following the Theorem 1.1 stated above, we get the adjoint equations below:

$$\lambda'_1 = -\frac{\partial H}{\partial b} \tag{6}$$

$$= -[\,1 + \lambda_1 (k_b - 2k_b b - \lambda_b) + \frac{(k_{nr}n + \delta)b\lambda_{rb - (\lambda_{rb}b+1)(k_{nr}n+\delta)}}{(\lambda_{rb}b+1)^2}\frac{w}{w+k_w} - \lambda_b\, b)$$

$$+ \lambda_2 \left(\frac{k_{ni}bn(1-n)\big(g_{nw}(w)\big)}{\lambda_{ni}n+1} + \frac{\lambda_b ne}{(1+eb)^2}\right) \quad + \lambda_3 (-\lambda_{bw}\, w)$$

$$\lambda'_2 = -\frac{\partial H}{\partial n} \tag{7}$$

$$= \qquad -\Big[\lambda_1 \left(\frac{-bk_nr}{\lambda_{rb}b+1}\frac{w}{w+k_w}\right) + \lambda_2 \Big(\frac{g_{nw}(w)[(\lambda_{ni}n+1)(k_{ni}b-2k_{ni}bn)-k_{ni}bn(1-n)\lambda_{ni}]}{(\lambda_{ni}n+1)^2} - \frac{\lambda_n}{1+eb} -$$

$$k_p e^{-\lambda_p t}\Big) + \lambda_3 (-\lambda_{nw}\, w)\Big]$$

$$\lambda'_3 = -\frac{\partial H}{\partial w} \tag{10}$$

$$= -\left[\lambda_1 \left(\frac{-b(k_{nr}n+\delta)}{(\lambda_{rb}b+1)}\frac{k_w}{(w+k_w)^2}\right) + \lambda_2 \left(\frac{k_{ni}bn(1-n)(g'_{nw}(w))}{\lambda_{ni}n+1}\right) + \lambda_3 (-\lambda_w - \lambda_{bw}\, b - \lambda_{nw}\, n)\right]$$

where $g'_{nw}(w) = f(x) = \begin{cases} 6w^2 - 6w \; for\; 0 \le w < 1, \\ \quad 0 \qquad for\;\; w \ge 1, \end{cases}$

and the final time values are:

$\lambda_1 (T) = 0,\ \lambda_2 (T) = 0,\ \lambda_3 (T) = 0.$

Since $\frac{\partial H}{\partial u} = (cu + \gamma\,\lambda_3)$, the optimality conditions follow as given below (Daulton, 2013):

$$u^*(t) = \begin{cases} 0 & implies\ (cu + \gamma\,\lambda_3) \ \ge 0\ at\ t, \\ 0 < \frac{-\gamma\lambda_3}{c} < M_{2,} & implies\ (cu + \gamma\,\lambda_3) = 0\ at\ t, \\ M_2 & implies\ \ (cu + \gamma\,\lambda_3) \le 0\ at\ t. \end{cases} \tag{12}$$

Determining the maximum value for delta used in the summation of Gaussian

This is the plot to show determination of delta used in Gaussian, which is indicated as delta2 in our Matlab code.

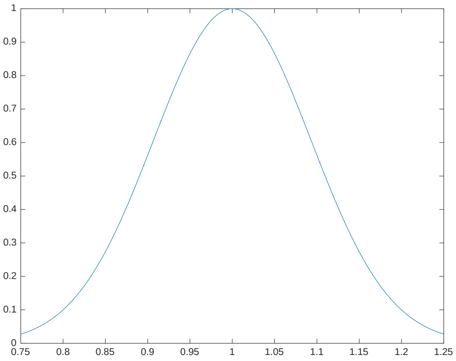

Figure 1: The maximum delta value for this graph tells us how long we should administer therapy in a day.

Looking at the objective function where we incorporate a summation of Gaussian factor $(\int_{t_0}^{t_1}[b(t) + (\sum_{i=1}^{P} e^{-\delta(t-\tau_i)^2})\frac{c}{2}u^2(t)]\,dt)$ in an optimal model for wound healing, the following results were obtained as indicated in Figure 3 below:

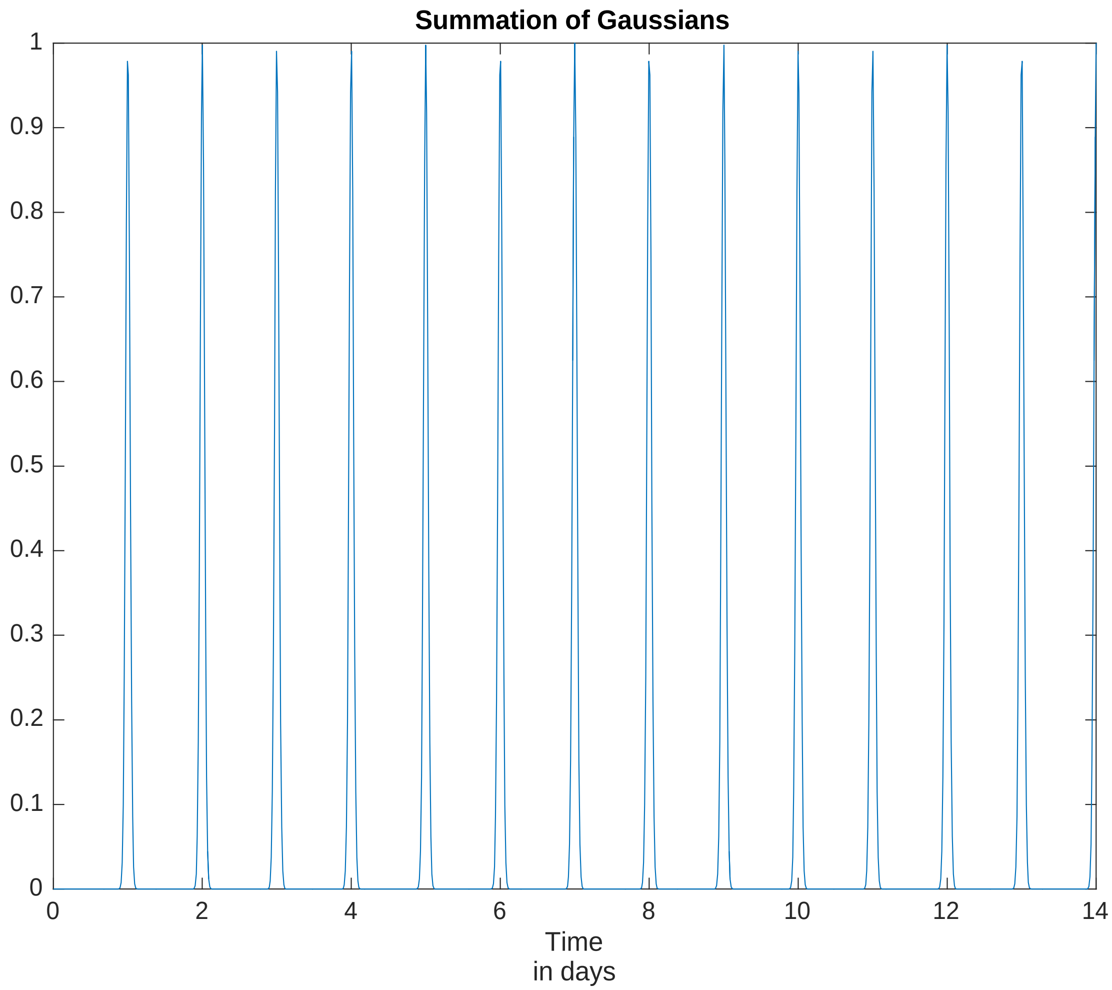

Figure 2: Summation of Gaussians for determining how long the therapy should be done (each day) for 14 days.

Proof for Nonlinear Existence

In order to obtain the solution to the above problem, the following theorem (also by Lukes) is helpful in making the arguments about the solution (Daulton, 2013).

Theorem: *Let L be the integrand of the objective functional,* $\vec{g}$ *be the right-hand side of the differential equations, U be a closed subset of* E^n*, the space of the n tuples x = (x-*$_{1}$*,…….*x_n*) of real numbers. Let* $\mathcal{F}'$ *be the class of all (*x_0*, u) such that u is a Lebesgue – integrable function on the interval [*t_0*,* t_1*] with the values in U and the solution of the differential equations satisfying the end conditions e* $\in$ *S. Let S be a given subset of* E^{2n+2} *and J(*x_0*, u) =* $\phi_j(t_0, t_1, x(t_0), x(t_1) = \phi(e)$ *for j=2,…..,k and e denotes a (2n+2)-tuple of the end points. For each (t, x)* $\in E^{n+1}$*, let* $\tilde{F}$ *(t, x) =* $\{\tilde{z}: z = g(t,x,u), zn+1 \geq L(t, x, u), u \in U\}$.

Suppose that $\vec{g}$ *is continuous, there exists positive constants* C_1*,* C_2 *such that*

(a). $|\vec{g}(t, x, u)| \leq C1(1 + |x| + |u|)$*,*

(b).$|\ (\ g(t), x', u) - g(t, x, u)| \leq C2|x'\text{-}x|(1+|u|)$ *for all* $t \in E1, x,\ x'^{(t)} \in E^n$*, and* $u \in U$*,*

L is continuous, and that:

1. $\mathcal{F}'$ *is not empty;*
2. *U is closed;*
3. *S is compact and* ϕ *is continuous on S;*
4. $\tilde{F}$ *(t, x) is convex for each (t, x)* $\in E^{n+1}$*;*
5. *L(t, x, u)* $\geq$ *h(u), where h is continuous and* $|u|^{-1}h(u) \rightarrow +\infty$ *as* $|u| \rightarrow \infty$*,* $u \in U$*.*

   Then there exist (x_0^**,*u^**) minimizing J(*x_0*,u) on* $\mathcal{F}'$*.*

From the above theorems we are going to check if all the conditions from Daulton's thesis are met (Dalton, 2013):

a. The set of the control and state variables is non-empty.
b. The control set U is closed and convex.
c. The RHS of the state variable is bounded by the linear function in both state and control variables.
d. The integrand of the objective function is convex on U.
e. There exists constants c_1,c_2>0, and $\beta > 1$ such that the integrand L (t, x, u) satisfies L(t, x, u) ≥$c_1|u|^{\beta}$ - c_2 .

Proof:

Following Luke's theorem stated above, we can prove the existence of solutions on a given bounded interval of coefficients. Following step (d), we also know that u is convex since of the derivative of u function is linear and is closed since its domain is closed; that is $0 \leq u \leq P * M_2$. Also using the same argument from Daulton's thesis (2013), the RHS of the state system of the equations is bounded by a linear function in the state and the control because we know that bacteria and neutrophils are bounded by the carrying capacities b_0 and n_0 respectively. Also considering that the amount of oxygen is bounded by $\{w_{int}, \frac{\beta+\gamma*PM_2}{\lambda_w}\}$, where P is the summation of Gaussians and M_2 is the maximum amount of oxygen input (Daulton, 2013). Let $\alpha = \beta + \gamma * PM_2$. Then $\frac{dw}{dt} = \alpha - \lambda_w * w$ is maximized where u= M_2 and at this point we are not considering the amount of oxygen used by bacteria and neutrophils, thus we equate n=b=0. Solving the differential equation step by step we have : $\frac{dw}{dt} = \alpha - \lambda_w * w$

1).$\int \frac{dw}{dt} = \alpha - \lambda_w * w = \int \frac{dw}{\alpha - \lambda_w * w} = dt$

2). Let u = $\alpha - \lambda_w * w\ , them\ du = -\lambda_w dw$

3). In| $\alpha - \lambda_w * w$| = $-\lambda_w t$ + C, $\alpha - \lambda_w * w = e^{-\lambda wt + c}$

4). $w = \frac{\alpha - C * e^{-\lambda wt}}{\lambda_w}$, at the initial stage, we have *w(0)* = $\frac{\alpha - C}{\lambda_w}$

5). So C= $\alpha - \lambda_w * w_{int}$. So we claim that $\overline{w}(t) = \frac{\alpha\left(1 - e^{-\lambda wt}\right) + \lambda_w * w_{int} e^{-\lambda wt}}{\lambda w}$ bounds the oxygen function.

6). We also need to show that $\overline{w}(t)$ is decreasing when $\overline{w}_{\text{int}} > \frac{\alpha}{\lambda_\omega}$ and is increasing when $w_{\text{int}} < \frac{\alpha}{\lambda_\omega}$. If $\overline{w}(t)$ is decreasing, then w(0) = w_{int}.

7). If $\overline{w}(t)$ is increasing, then the maximum value $\lim_{t \to \infty} w(t) = \frac{\alpha}{\lambda_w} = \frac{\beta + \gamma * M_2}{\lambda_w}$, and w'(t) = $\frac{d\overline{w}}{dt} = (\alpha - \lambda_w * w_{int})e^{-\lambda wt}$.

8). Thus we conclude that w is bounded above by $\overline{w}$ and the maximum amount of oxygen is given by M=max{b_0, n_0, w_{int}, $\frac{\beta + \gamma * PM_2}{\lambda_w}$} which bounds the the state and control variables.

9). The integrand of the objective function is convex on u because $(b + \sum_{\tau_i = 0}^{P} e^{-\delta(\tau - t\boldsymbol{i})^2} * \frac{c}{2} u^2)$ is convex function as defined in Bartle and Sherbert (Bartle et al,2000):

"*Let I be an open interval and suppose that f: I →R has a second derivative on I. Then f is convex function on I iff f" (x)*$\geq 0\ for\ all\ x \in I$."

Thus the second derivative of$(b + \sum_{\tau_i=0}^{P} e^{-\delta(\tau-t)^2} * \frac{c}{2}u^2)$= C $\sum_{\tau_i=0}^{P} e^{-\delta(\tau-t)^2}$> 0. Let C_1 = C $\sum_{\tau_i=0}^{P} e^{-\delta(\tau-t)^2}$> 0, C_2 > 0 and β=2. Thus $(b + \sum_{\tau_i=0}^{P} e^{-\delta(\tau-t)^2} * \frac{c}{2}u^2)\ \geq$ $C_1|u|^2 - C_2$.

Non- Linear Solution

Using the Hamiltonian given as equation 7, we test different initial conditions and parameters to see if our objective functional meets the convergence criterion. We make different simulations to see which one drives bacteria to zero. Also, we also make sure oxygen goes to zero because it is not being used by bacteria and neutrophils, there is a likelihood of oxygen toxicity in the body thus finding our solutions to our biological problem.

Important to note in the code are the parameters $k_{nr}, \delta, \lambda_{bw,}\ \lambda_{nw}$ which are added to our differential equations 2, 3, and 4. They work as Lagrange multipliers which add constraints to the equation . They are meant to kill the bacteria faster so as to increase the likelihood of convergence of the results (Daulton, 2013). Also, parameters A, B, C, and D are numbers either divided or multiplied by $k_{nr}, \delta, \lambda_{bw,}\ \lambda_{nw}$ respectively . We make the following choice A$\gg$B due to the oxidative killing of bacteria by the presence of neutrophils in the wound (Daulton, 2013). It is important to note is that, to keep the same ratio of $\frac{\lambda_{bw,}}{\lambda_{nw}}$, we chose C = D. We use parameters parameters in Daulton's thesis which come from the work of Schugart and Joyce to check for the convergence criterion with these parameters as shown in Table 1 below, but with parameters bacteria still persists in the wound as shown in the Figure 3 below (Daulton, 2013):

Parameter Values	
c	0.1
kb	14.256
knr	2*A
δ	3.84*B
λ_{rb}	3.73
kw	0.75
λ_b	0.14256
kp	0.52
λ_p	3.04
λ_{ni}	80
λ_n	0.1728
γ	1
λ_w	1.0656
β	0.7992
λ_{bw}	12.6593/C
λ_{nw}	25.5744/D
e	100
kni	10.28

Table 1: The above are parameters used in our code to test for test for convergence criterion where the the values of A, B, C, D = 80,8, 3, 3 respectively unless stated otherwise.

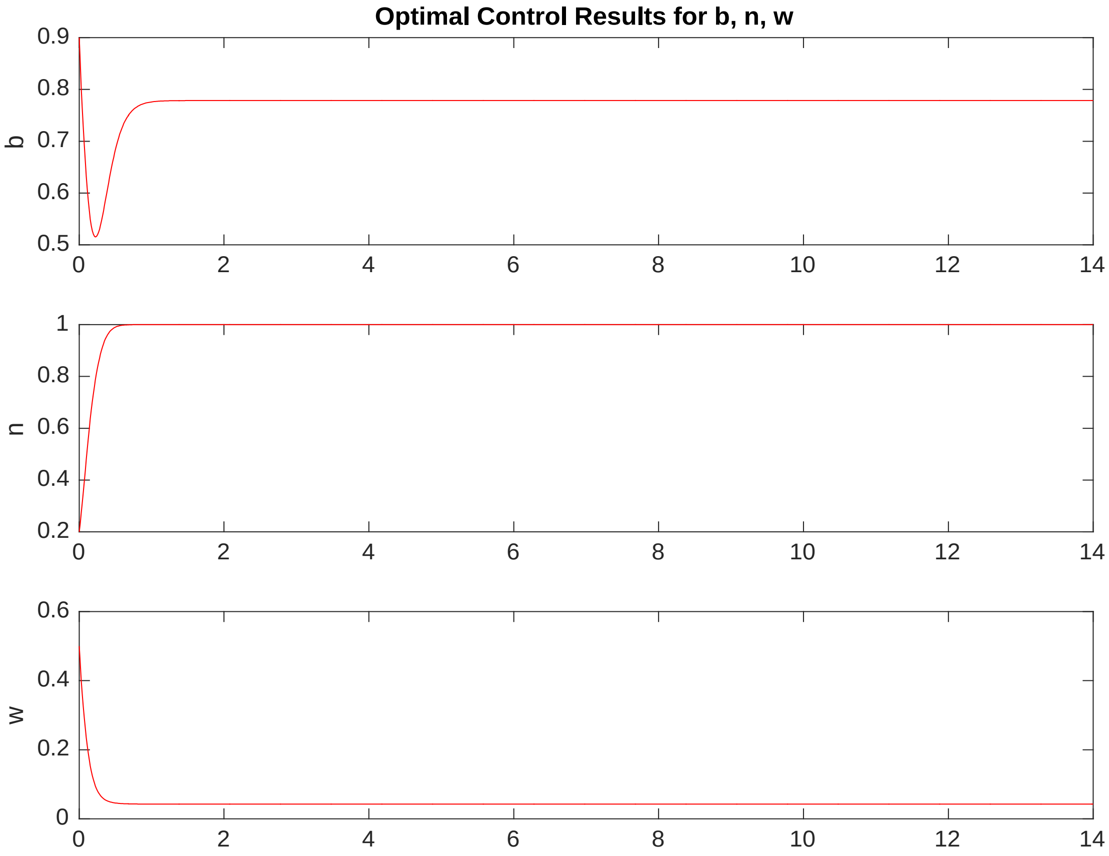

Figure 3: This is the figure for b, n, and w when there is no oxygen therapy (u-input). We can notice from the figure that bacteria persist in the wound.

We are going to run simulations with different parameters to see which ones converge using the following results in the table from Daulton's thesis. The main goal for using Daulton's results is to see if there is a change when we incorporate the summation of Gaussians in our objective functional. The following table from Daulton's thesis shows different initial conditions that are tested for *b, n, w, A, B, C,* and *D* where *b* stands for bacteria, *n* neutrophil level, and *w* oxygen from surrounding blood vessels as indicated earlier on page 4.

Initial Condition Parameters		**J -value**	**Wound Healing**	**Time taken (days)**
(b, n, w)	(A, B, C, D)			
(0.9, 0.2, 0.5)				
	(100, 5, 2, 2)	0.7828	No	-
	(80, 8, 3, 3)	0.6999	No	-
	(70, 6, 5, 5)	0.44434	Yes	11.8
(0.7, 0.1, 0.4)				
	(100, 5, 2, 2)	0.7882	No	-
	(80, 8, 3, 3)	0.7077	No	-
	(70, 6, 5, 5)	0.4803	Yes	12.2
(0.5, 0.1, 0.5)				
	(100, 5, 2, 2)	0.7852	Not	-
	(80, 8, 3, 3)	0.7037	Not	-
	(70, 6, 5, 5)	0.4818	Yes	13

Table2: This table shows different initial parameters of *b, n,* and *w* that were tested to find the convergence criteria. Parameters A, B, C, and D are numbers either divided or multiplied by $k_{nr}, \delta, \lambda_{bw,}\ \lambda_{nw}$ respectively. They vary for each initial condition to find out which ones would make bacteria go to zero.

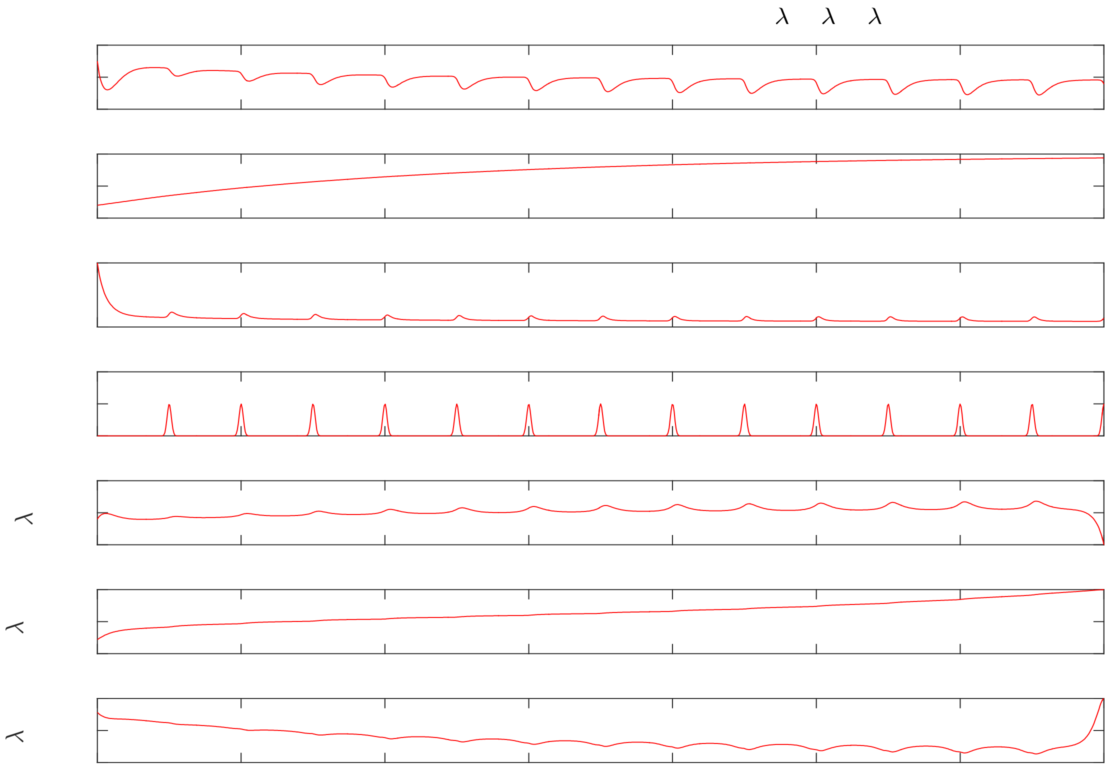

Figure 4: Results for (b, n, w) = (0.9, 0.2, 0.5) and (A, B, C, D) = (100, 5, 2, 2).

It is easily seen that adding Gaussian factors in the objective functional changes the shape of u results. The number of bacteria in the wound does not go to zero. The peaks of the curve u show how long the therapy is done and this goes on for one day to fourteen days. This is more biologically applicable because it shows how much and how long oxygen therapy should be administered, keeping in mind that too much oxygen for long periods could cause oxygen toxicity in the body. But the bacteria persist in the wound for the 14 days. It is worth noticing that every time oxygen u is administered, there is a bump in bacteria results where bacteria goes down during the therapy.

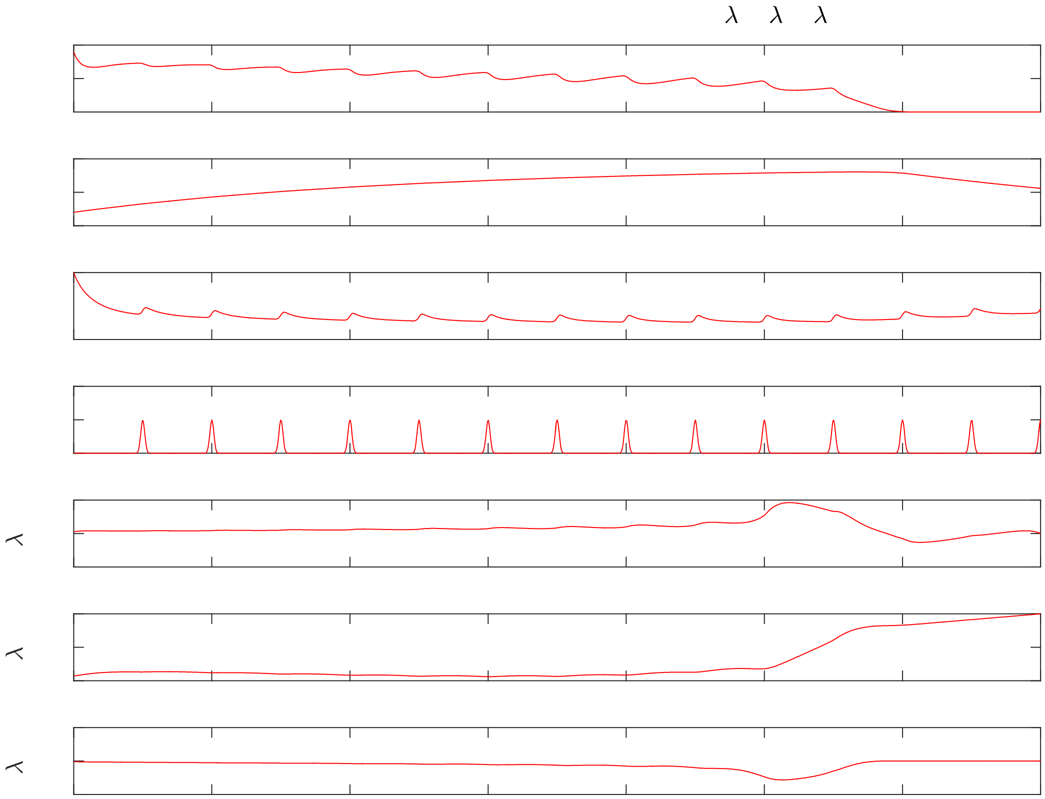

Figure 5: Results for and (b, n, w) = (0.9, 0.2, 0.5) and (A, B, C, D) = (70, 6, 5, 5)

With parameter values of (A, B, C, D) = (70, 6, 5, 5). The number of bacteria in the wound goes to zero around the twelfth day. The peaks of the curve u show how long the therapy is done and this goes on for one day to 14 days. This is more biologically applicable because it shows how much and how long oxygen therapy should be administered, keeping in mind that too much oxygen for long periods could cause oxygen toxicity in the body. It also drives bacteria to zero which is biologically reasonable because bacteria are removed from the wound thus leading to wound healing.

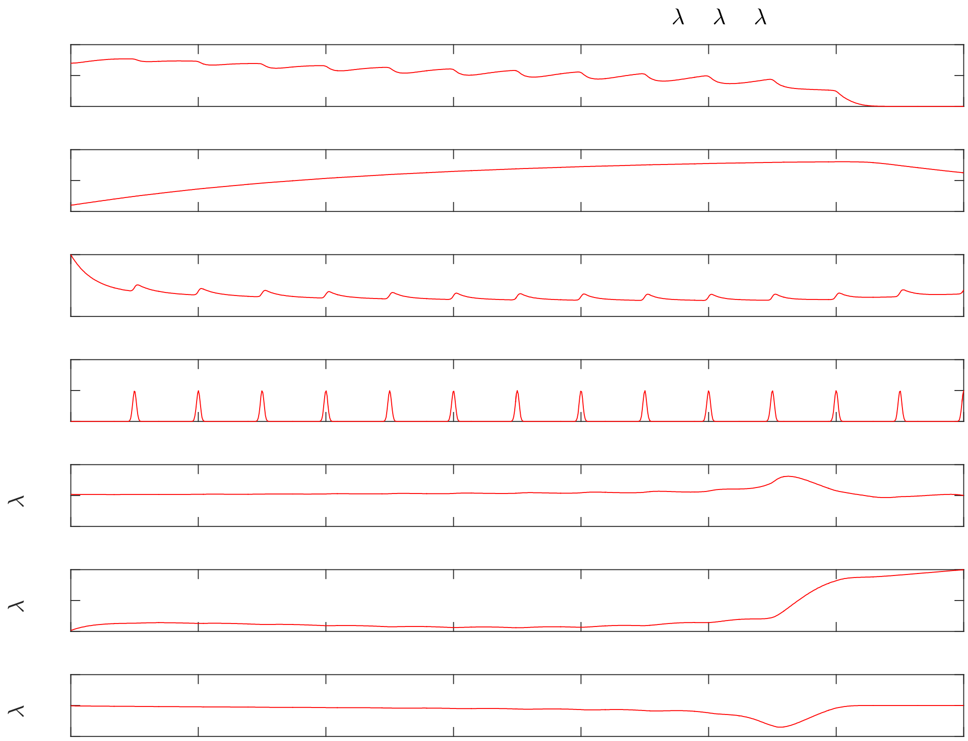

Figure 6: In this case, (b, n, w) = (0.7,0.1,0.4). Still bacteria go to zero. This also gives a more reasonable biological solution since bacteria is removed from the wound and administration of hyperbaric oxygen is still administered on hourly basis for 14 days. Oxygen input u goes to zero after 14th day which still makes biologically applicable since there is no risk of oxygen toxicity.

For the given numerical results, in Figure 5 and 6 bacteria is removed from wound and the oxygen input u also goes to zero. The difference is that different initial conditions gives different number of days it takes for the bacteria to be removed from the wound. These results are more biologically applicable since bacteria is removed from the wound and oxygen input goes to zero after 14 days of therapy.

CHAPTER 4

LINEAR CONTROL

Forming the Hamiltonian for Linear Control

Our linear control is given below as follows:

$$J(u) = \int_{t_0}^{t_1} [b(t) + (\sum_{i=1}^{P} e^{-\delta(t-\tau_i)^2})cu(t)]\, dt) \qquad (12)$$

where 0 ≤ u ≤ M2.

The linear model for our objective functional is called topical oxygen therapy where oxygen is administered to the wound directly. In order to form the Hamiltonian for our linear control, we will still use differential equations used earlier for non-linear case from Schugart and Joyce (Daulton, 2013).

Linear Existence

In order to prove existence of linear solutions, we use 'Optimal Control Theory with Economic Applications' by Filippov – Cesari's work by Seierstad and Sydsaeter (1987, p. 285 Theorem 2) as stated by Daulton (2013) by considering the following problem,

max $\int_{t_0}^{t_1} f_0\big(x(t), u(t)\big)dt, (t_0\,, t_1\, fixed)$

subject to vector differential equation and the initial condition

$\frac{d\vec{x}}{dt} = f(\vec{x}(t), u(t), t), x(t_0) = x^0$ (x^0 fixed) and the terminal conditions are:

$x_i(t_1) = x_i^1$ for *i* = 1,…,*l* (x_i^1 all fixed)

$x_i(t_1) \geq x_i^1$ for *i* = *l* +1,…,*m* (x_i^1 all fixed)

$x_i(t_1)$ free for *i* = *m* +1…, *n*,

and for all *t* ∈ $[t_0, t_1]$ and the constraints

$h_k(x(t), u(t), t) \geq 0$, k = 1, 2…, *s*. (Daulton, 2013)

For the given set of necessary conditions below, we can prove the existence of solutions:

1. There exists admissible pair $(x(t), u(t))$.
2. The set N$(x, t) = \{ f_0(x, u, t) + \rho, f(x, u, t)\big): \rho \leq 0, h(x, u, t) \geq 0\}$ is convex for all x and *t* ∈ $[t_0, t_1]$.
3. There exists a number b such that $\|x(t)\| \leq b$ for all admissible pairs $(x(t), u(t))$, and all *t* ∈ $[t_0, t_1]$.
4. There exists a ball *B*(0, b_1) in R^r which, for all x with U$(x, t) = \{u: h(x, u, t) \geq 0\}$.

Then we say that there exists a measurable optimal control (Daulton, 2013).

Proof:

1. Consider $(x(t), u(t))$ an admissible pair since *u(t)* is piecewise continuous and x(*t*) is both continuous and piecewise continuously differentiable, it satisfies the vector differential equation, initial conditions, and constraints with free terminal conditions (Daulton, 2013).
2. The set N$(x, t) = \{ f_0(x, u, t) + \rho, f(x, u, t)\big): \rho \leq 0, h(x, u, t) \geq 0\}$ is convex for all $\vec{x}$ and all *t* ∈ $[t_0, t_1]$ (Daulton, 2013). A function f(x) is defined as convex on an interval [a, b] if for any two points x_1 and x_2 in [a, b] and any φ where $0 \leq \varphi \leq 1$,

then f [φx_1 + (1- φ)$x2$]≤ φ f(x_1) + (1- φ) f(x_2). (Rudin, 1976, p. 101)

In our case, we have that:

$f_0 = b + Zcu$, where Z is the summation of Gaussian ($\sum_{i=1}^{P} e^{-\delta(t-\tau_i)^2}$)

$f_0(\vec{x}, u_1) + \rho = b + Zcu_1 + \rho$

$f_0(\vec{x}, u_2) + \rho = b + Zcu_2 + \rho$

$f_0(\vec{x}, u_2) - f_0(\vec{x}, u_1) = Zc\ (u_2 - u_1)$

$\frac{\partial f_0(\vec{x},u)}{\partial u} = Zc$

$\Rightarrow (u_2 - u_1)\frac{\partial f_0(\vec{x},u)}{\partial u} = cZ$

Using our third adjoint equation, we have the following:

$f = \beta + \gamma * \sum_{i=1}^{P} e^{-\delta(t-\tau_i)^2}) * u(t) - \lambda_w w - \lambda_{bw} bw - \lambda_{nw} nw$

$f(\vec{x}, u_1) = \beta + \gamma * \sum_{i=1}^{P} e^{-\delta(t-\tau_i)^2}) * u_1 - \lambda_w w - \lambda_{bw} bw - \lambda_{nw}\ nw$

$f(\vec{x}, u_2) = \beta + \gamma * \sum_{i=1}^{P} e^{-\delta(t-\tau_i)^2}) * u_2 - \lambda_w w - \lambda_{bw} bw - \lambda_{nw}\ nw$

$f_0(\vec{x}, u_2) - f_0(\vec{x}, u_1) = \gamma * Z *(u_2 - u_1)\ \ \frac{\partial f_0(\vec{x},u)}{\partial u} = \gamma$

$\Rightarrow Z(u_2 - u_1)\frac{\partial f_0(\vec{x},u)}{\partial u} = \gamma * Z * (u_2 - u_1).$

Using the same argument from Daulton's thesis, we know that if a function *f* is differentiable, then *f* is convex if and only if $f(x_2) - f(x_1) \leq (x_2 - x_1)f'(x_2)$. From this we see that the property holds for our case since $Zcx_2 - Zcx_1 \leq (x_2 - x_1)cZ$.

3. Also, there is a number *b* such that $\|x(t)\| \leq b$ for all admissible pairs, $(x(t), u(t))$, and *t* ∈ $[t_0,\ t_1]$ where *b*= *max*{b_0, n_0, w_{int}, $\frac{\beta+\gamma*Z*M_2}{\lambda_w}$} as shown earlier in the nonlinear problem (Daulton, 2013).

4. There exists a ball B (0, b$_1$) in R^rsuch that for all x

$U(x,u) = \{u\colon h(x,u,t) \geq 0\}$ which is a convex subset of R^r, where r is the number of control variables. This is considered true because u is always between [0, M], $\|x(t)\| \leq b$, and $t \in [t_0,\ t_1]$, where t_1 is the final time. Thus $U(x,u) = \{u\colon h(x,u,t) \geq 0\}$. From this, we can define the convexity of the function using Helly's Theorem 1993 as for a given vector space X, there is a subset K of X which is convex if for any two points x, y $\in K$, we have c $\in V$, for every point, then c= (1– φ) $x + \varphi\ y$, with $0 \leq \varphi \leq 1$ (where $\varphi \in R$).

Let $x, y \in U$, assume without loss of generality $0 \leq x \leq y \leq M$ (Daulton, 2013).

Let $h_1(u) = \ Z * u, h_2 = M - Z * u \ \geq 0$. Then

$h_1(x) \geq 0 \ \Rightarrow Zx \geq 0,$

$h_1(y) \geq 0 \ \Rightarrow y \geq 0,$

$h_2(x) \geq 0 \ \Rightarrow M - Zx \geq 0,$

$h_2(y) \geq 0 \ \Rightarrow M - Zy \geq 0,$

Let w = φx + (1- φ) $y \in U$ for $0 \leq \varphi \leq 1$. Thus we need to show that $h_1(x) \geq 0$ and $h_2(x) \geq 0$

$h_1(w) = \ \varphi Zx$ + (1- φ) Z $y \geq 0$

$h_2(w) = M - \ \varphi Zx$ – (1- φ) Z y

$= M–y + \varphi$y–φx

$= M – Zy + Z\varphi(y–x) \geq 0$ for $x \leq y$.

Thus we can conclude that U is convex hence proving the existence of the linear solution.□

Forming the Hamiltonian for Linear Control

The Hamiltonian is for combining the Integrand which in this case is our objective functional and the right hand side of our differential equations as given below:

$$H = [b + (\sum_{i=1}^{\tau} e^{-\delta(t-\tau_i)^2})\frac{c}{2}u(t)]\,dt + \leftthreetimes_1\,(k_b b(1-b)\text{-}b\frac{k_{nr}n+\delta}{\leftthreetimes_{rb}b+1}\frac{w}{w+k_w} - \leftthreetimes_b\, b)$$

$$+ \leftthreetimes_2\,(k_p e^{-\leftthreetimes_p t}(1-n) + \frac{k_{ni}bn(1-n)\big(g_{nw}(w)\big)}{\leftthreetimes_{ni}n+1} - \frac{\leftthreetimes_b n}{1+eb}) \qquad (13)$$

$$+\leftthreetimes_3\,(\beta + \gamma * \sum_{i=1}^{P} e^{-\delta(t-\tau_i)^2}) * u(t) - \leftthreetimes_w\, w - \leftthreetimes_{bw}\, bw - \leftthreetimes_{nw}\, nw)$$

Using Luke’s Theorem 1.1 stated on page 5 we get the following adjoint equations:

$$\leftthreetimes'_1 = -\frac{\partial H}{\partial b}$$

$$= \text{-}[\,1 + \leftthreetimes_1\,(k_b - 2k_b b - \leftthreetimes_b) + \frac{(k_{nr}n+\delta)b\leftthreetimes_{rb-(\leftthreetimes_{rb}b+1)(k_{nr}n+\delta)}}{(\leftthreetimes_{rb}b+1)^2}\frac{w}{w+k_w} - \leftthreetimes_b\, b)$$

$$+ \leftthreetimes_2\left(\frac{k_{ni}bn(1-n)\big(g_{nw}(w)\big)}{\leftthreetimes_{ni}n+1} + \frac{\leftthreetimes_b ne}{(1+eb)^2}\right) \;\; +\leftthreetimes_3\,(-\leftthreetimes_{bw}\, w)$$

$$\leftthreetimes'_2 = -\frac{\partial H}{\partial n}$$

$$= \qquad \text{-}\Bigg[\leftthreetimes_1\left(\frac{-bk_nr}{\leftthreetimes_{rb}b+1}\frac{w}{w+k_w}\right) + \leftthreetimes_2\left(\frac{g_{nw}(w)[(\leftthreetimes_{ni}n+1)(k_{ni}b-2k_{ni}bn)-k_{ni}bn(1-n)\leftthreetimes_{ni}]}{(\leftthreetimes_{ni}n+1)^2} - \frac{\leftthreetimes_n}{1+eb} -\right.$$

$$\left. k_p e^{-\leftthreetimes_p t}\right) + \leftthreetimes_3\,(-\leftthreetimes_{nw}\, w)\Bigg]$$

$$\leftthreetimes'_3 = -\frac{\partial H}{\partial w}$$

$$= \text{-}\left[\leftthreetimes_1\left(\frac{-b(k_{nr}n+\delta)}{(\leftthreetimes_{rb}b+1)}\frac{k_w}{(w+k_w)^2}\right) + \leftthreetimes_2\left(\frac{k_{ni}bn(1-n)(g'_{nw}(w))}{\leftthreetimes_{ni}n+1}\right) + \leftthreetimes_3\,(-\leftthreetimes_w - \leftthreetimes_{bw}\, b - \leftthreetimes_{nw}\, n)\right]$$

where $g'_{nw}(w) = f(x) = \begin{cases} 6w^2 - 6w \; for \; 0 \le w < 1, \\ \quad 0 \qquad for \;\; w \ge 1, \end{cases}$

and the final time values are:

$\lambda_1\,(T) = 0,\ \lambda_2\,(T) = 0,\ \lambda_3\,(T) =\ 0.$

Linear Solution

In a way to get results for our linear problem, we test different initial parameters for b, n, and w in our code to see which parameters remove the bacteria from the wound. In our linear problem, bacteria are removed from the wound for most of the cases. This shows that our oxygen input (topical oxygen) therapy works relatively well in wound treatment.

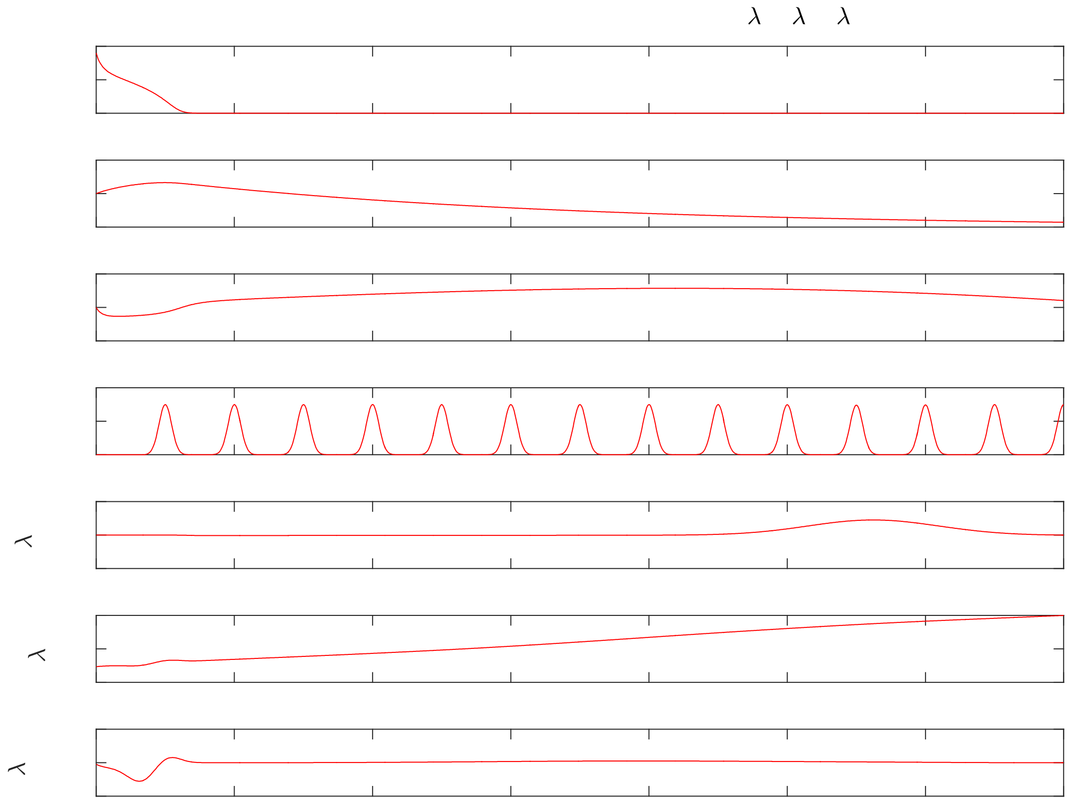

Figure 7: This figure shows results for (b, n, w) = (0.9, 0.2, 0.5).

In this case, bacteria are removed after about a day of the therapy but oxygen therapy goes on for 14 days. This is biologically reasonable since bacteria is removed from the wound and oxygen is administered at an hourly basis every day for 14 days.

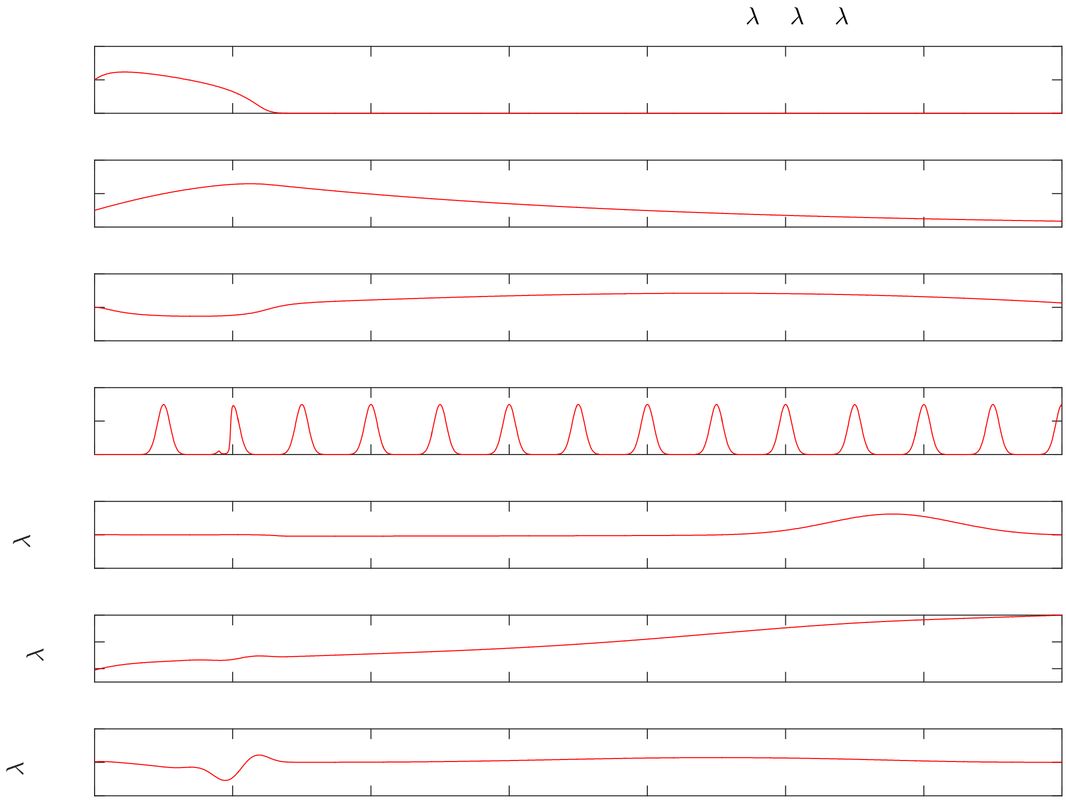

Figure 8: Results for and (b, n, w) = (0.5, 0.1, 0.5).

In this case, bacteria are removed from the wound after about 2.2 days of therapy. This also provides a biologically reasonable solution since bacteria are removed and therapy is done on an hourly basis everyday.

Initial Condition Parameters		**J -value**	**Healing Time (days)**
(b, n, w)	(A, B, C, D)		
(0.9, 0.2, 0.5)			
	(100, 5, 2, 2)	52.5080	1.2
	(80, 8, 3, 3)	35.0598	0.5
	(70, 6, 5, 5)	39.2118	0.8
(0.7, 0.1, 0.4)			
	(100, 5, 2, 2)	94.6806	2.1
	(80, 8, 3, 3)	47.5718	1
	(70, 6, 5, 5)	49.703	1.7
(0.5, 0.1, 0.5)			
	(100, 5, 2, 2)	98.7443	2.2
	(80, 8, 3, 3)	38.8580	0.9
	(70, 6, 5, 5)	53.2882	1.5

Table 3: This table summarizes our results for linear problem. It shows initial condition parameters with respective J-values which is our objective functional.

In all cases for our linear problem, bacteria go to zero which means our hourly therapy was effective. J values for linear control vary between 30 and 100 for all initial conditions and parameters. There is also variation in the values of J for the non-linear problem but it is less. This is because the values we choose for u is between 0 and 1, and squaring a decimal will make the value smaller hence leading to smaller J value. From table, we also notice that the time taken for the wound to heal for our linear problem is shorter. This is because we are putting more oxygen in wound which removes bacteria quickly from the wound. For the non- linear problem, for the cases when the wound healed, it was longer because of less of oxygen input in the therapy. This can be understood by knowing that squaring any decimal between 0 and 1 makes the decimal number smaller hence less amount of oxygen input. Also, important to note is that our oxygen therapy is done daily for 14 days but it would be more biologically applicable if the therapy stopped after bacteria are removed from the would so as to avoid oxygen toxicity.

CHAPTER 5

CONCLUSION AND FUTURE WORK

We developed a non-linear and linear model of exponential functions from existing models of optimal control for hyperbaric and topical oxygen therapy of a chronic wound. Our model reasonably shows how best we can capture the best results and when it is good to administer therapies. It also shows how long the therapy should be administered in a day. We found the value of the Gaussian in order to get the best therapy and length so that we can use the right amount of oxygen and therapy to avoid high cost of therapies and toxicity from excess oxygen.

Our results for a non-linear show that for cases when bacteria are removed from the wound, it takes a bit longer (periods about 10- 13 days of daily hyperbaric oxygen therapy). But still this is biologically applicable since oxygen therapy is done on an hourly basis each day and there will be no risk of oxygen therapy since the amount administered is controlled. There is also variation in our results and the value of J depending on the initial conditions and parameters used.

The linear case captured significant results since bacteria converged to zero for all cases and it took a few days for bacteria to be removed from the wound. The topical oxygen therapy was still administered with in the time frame of 14 days even after bacteria removal. But this not best treatment since oxygen is still administered even after bacteria removal yet we wanted to minimize amount of oxygen used in the therapy to avoid costly therapies.

Our future goal is to see what happens when one administers therapies on different days that is waiting for some days before doing therapy and how long it should be done per day. We also would like to test different parameters to see which ones make the bacteria and neutrophils converge to zero. This is a first step toward customizing treatment for patients based on administering hourly therapies.

Also, other areas of future work include using real data as opposed to making initial guesses for our parameters and doing simulations on them. Real data helps in knowing well which conditions to use and thus helping in making best decisions. The advantage of this is that it helps in choosing the best parameters which makes it easy to know when or when not to administer oxygen therapy thus avoiding the risk of oxygen toxicity and costly therapies.

The future work will also consider what happens if we do therapy for longer periods each day instead of having therapy go on for a short time every day. It may be the case that doing longer therapies every day would make bacteria go to zero quickly especially for our non-linear problem $(J(u) = \int_{t_0}^{t_1} [b(t) + (\sum_{i=1}^{P} e^{-\delta(t-\tau_i)^2})cu(t)]\, dt)$, also referred to as hyperbaric oxygen therapy, where it takes more than 10 days for bacteria to be removed from the wound.

Lastly, we would like to consider a piecewise function as opposed to continuous non-linear and linear problem. A piecewise function could work well because instead of a continuous function for input of oxygen where there is a likelihood of oxygen toxicity in the body, we would have sub-intervals where we can choose to administer therapy on some days. Examples of piecewise functions are absolute value functions which always have subdomains. This function minimizes the amount of oxygen used where oxygen is only administered where necessary.

Appendix

```
function y=delta3()
t=18:0.01:30;
delta=0.00001;
epsilon=0.01;
y=exp(-delta*((t-24)/epsilon).^2);
figure(11)
plot(t./24,y)
t1=0:0.01:48;
y=exp(-delta*((t1-24)/epsilon).^2);
delta1=0.006;
epsilon1=0.01;
t2=0:0.001:18;
y2=exp(-delta1*((t2-1)/epsilon1).^2);
figure(12)
plot(t./24,y,'-k',t2,y2,'-b')

end
```

Figure 9: Code for determining the value of delta 3 used in summation of Gaussian

```
function y = nonlinearproblem5(b0,n0,w0,A,B,C,D)% assigns variable b0,
n0 and w0 as
% inputs, y as output

warning('off','all')
test = -1; % convergence test variable- begins the while loop with a
neg
% number
tf=14;
zeta = .1; % convergence criterion
N = 1000; % number of nodes
t = linspace(0,tf,N+1); % creates N+1=1001 equally spaced nodes
t1=linspace(tf,0,N+1);
h = tf/N; % spacing is assigned as h
h2 = h/2; % short-hand for Runge-Kutta subroutine (h2 short for h/2)
M2 = 16.37; %max bound of u see page 82
M1 =0; %min bound of u see pg 82

delta2 =0.06;
epsilon1=0.01;
k = 0;
u = 0;

while k < 14
    k = k+1;
    u = u+0.5*ones(1,N+1).*(exp(-delta2*((t-k)/epsilon1).^2));
end

%u = zeros(1,N+1);
%u = -M2*t/tf +M2;
u = 0.5*ones(1,N+1);
u1=u;

n = zeros(1,N+1); % vector n and size
n(1) = n0; % initial condition for n because matlab recognizes 1 as the
% first element
 %n(N+1) = n;
b = zeros(1,N+1); % vector b and size
b(1) = b0; % initial condition for b
w = zeros(1,N+1); % vector w and size
w(1) = w0; % initial condition for w
lambda1 = zeros(1,N+1); % lamda1 and size
lambda2 = zeros(1,N+1);
lambda3 = zeros(1,N+1);
%x1exact = zeros(1,N+1);
%x2exact = zeros(1,N+1);
%uexact = zeros(1,N+1);
k=0; %k is my counter see pg 82
temp=0;
j=0;
tau =0;
while(test < 0 && k<1500)%25000) % when convergence occurs test will
become non-negative
    k = k+1;
    c = .000000001;
    oldu = u; % previous value of u
    oldn = n; % previous value of n
    oldb = b; % previous value of b
    oldw = w; % previous value of w
    oldlambda1 = lambda1; % previous value of lambda1
    oldlambda2 = lambda2; % previous value of lambda2
    oldlambda3 = lambda3; % previous value of lambda3
    kb = 14.256;
    knr = 2*A;
    delta = 3.84*B;
    lambdarb = 3.73;
    kw = .75;
    lambdab = 0.14256;
    kp = 0.052;
    lambdap = 3.04;
    kni = 10.28;
    lambdani = 80;
    lambdan = .1728;
    gamma = 1;
    lambdaw = 1.0656;
    beta = 0.7992;
    lambdabw = 12.6593/C;
    lambdanw = 25.5744/D;
    e = 100;
    tau = tau+0;
    %delta2 = 0.06;
    oldj = j;

params1=[kb,knr,delta,lambdarb,kw,lambdab,kp,lambdap,kni,lambdani,lambd
an,beta,gamma,lambdaw,lambdabw,lambdanw,e,delta2,tau];

     [T1,x]=firstfunction(b0,n0,w0,t,t,u1,params1);
     if t~=T1'
         error('time values for x do not match')
     end
     b=x(:,1)';
     n=x(:,2)';
     w=x(:,3)';

    [T2,lambda]=secondfunction(b,n,w,t1,t1,params1);
%     if t~=T2'
%         t
%         T2
%         error('time values for lambda do not match')
%     end
   lambda1=flipud(lambda(:,1))';
   lambda2=flipud(lambda(:,2))';
   lambda3=flipud(lambda(:,3))';

figure(12)
%display
%title(figure(12),'summation of Gaussiana')
plot(t,exp(-delta2*((t-1)/epsilon1).^2) + exp(-delta2*((t-
2)/epsilon1).^2) ...
+exp(-delta2*((t-3)/epsilon1).^2) +exp(-delta2*((t-4)/epsilon1).^2)+
exp(-delta2*((t-5)/epsilon1).^2)+ ...
exp(-delta2*((t-6)/epsilon1).^2)+ ...
```

```
    exp(-delta2*((t-7)/epsilon1).^2)+exp(-delta2*((t-8)/epsilon1).^2)+
...
   exp(-delta2*((t-9)/epsilon1).^2) +exp(-delta2*((t-10)/epsilon1).^2)
...
   +exp(-delta2*((t-11)/epsilon1).^2) ...
    +exp(-delta2*((t-12)/epsilon1).^2)+exp(-delta2*((t-
13)/epsilon1).^2) ...
    +exp(-delta2*((t-14)/epsilon1).^2));
title('Summation of Gaussians')
xlabel({'Time' 'in days'})
%display(figure(12))


 u1 = max(M1.*(exp(-delta2*((t-1)/epsilon1).^2) + exp(-delta2*((t-
2)/epsilon1).^2)+ ...
     exp(-delta2*((t-3)/epsilon1).^2)+exp(-delta2*((t-4)/epsilon1).^2)
...
     +exp(-delta2*((t-5)/epsilon1).^2)+exp(-delta2*((t-
6)/epsilon1).^2)),...
    min(u1.*(exp(-delta2*((t-1)/epsilon1).^2) + exp(-delta2*((t-
2)/epsilon1).^2)+ ...
    exp(-delta2*((t-3)/epsilon1).^2)+ ...
    exp(-delta2*((t-4)/epsilon1).^2)+exp(-delta2*((t-5)/epsilon1).^2)
...
    +exp(-delta2*((t-6)/epsilon1).^2))...
    - h*(c*u1.*(exp(-delta2*((t-1)/epsilon1).^2)  ...
    + exp(-delta2*((t-2)/epsilon1).^2)+exp(-delta2*((t-3)/epsilon1).^2+
...
    exp(-delta2*((t-4)/epsilon1).^2))) +exp(-delta2*((t-
5)/epsilon1).^2)+ ...
    exp(-delta2*((t-6)/epsilon1).^2)...
    +gamma*lambda3),M2.*(exp(-delta2*((t-1)/epsilon1).^2)...
    + exp(-delta2*((t-2)/epsilon1).^2) ...
    +exp(-delta2*((t-3)/epsilon1).^2)+exp(-delta2*((t-4)/epsilon1).^2
...
    +exp(-delta2*((t-5)/epsilon1).^2)+exp(-delta2*((t-6)/epsilon1).^2)
...
    +exp(-delta2*((t-7)/epsilon1).^2)))));
u1 = max(M1, min(u1 - h*(c*u1+gamma*lambda3),M2));
temp = sum(abs(c*u1+gamma*lambda3));
%test = 65-sum(b+(c/2)*u1.^2);
j = sum(b+(c/2)*u1.^2+1);
temp2 = abs(j-oldj);
test = .15-temp2;



    if floor(k/10)==k/10
        display(test)
        %display([temp11,temp21,temp22,temp31,temp23,temp33,temp32]);
    end
end

y(1,:) = t; % defines t
y(2,:) = n; % defines n
y(3,:) = b; % defines b
y(4,:) = w; % defines w
```

```
y(5,:) = lambda1; % defines lambda1
y(6,:) = lambda2; % defines lambda2
y(7,:) = lambda3; % defines lambda3
y(8,:) = u1; % defines u

 J=sum(b+(c/2)*u1.^2);
 display(J)
 figure(1)
 hold on;
 subplot(7,1,1)
 plot(t,b,'r-')
 ylabel('b')
 title('Optimal control results for b, n, w, u(t),\lambda_1, \lambda_2,
\lambda_3')
 subplot(7,1,2)
 plot(t,n,'r-')
 ylabel('n')
 subplot(7,1,3)
 plot(t,w,'r-')
 ylabel('w')
 subplot(7,1,4)
 plot(t,u1,'r-')
 ylabel('u(t)')
 ylim([0,1])
 %xlabel('t (in days)')


 %figure(2)
 %hold on;

 subplot(7,1,5)
 plot(t,lambda1,'r-')
 ylabel('\lambda_1')
 %title('Optimal control results for \lambda_1, \lambda_2, \lambda_3')
 subplot(7,1,6)
 plot(t,lambda2,'r-')
 ylabel('\lambda_2')
 subplot(7,1,7)
 plot(t,lambda3,'r-')
 ylabel('\lambda_3')
 xlabel('t (in days)')

% %display('k =')
% display(k)
end
%----------------------------------------------------------------------
---
function gnwvalue=gnw(w)

if w<1
   % if w<0
    %    error('w is negative')
    %end
    gnwvalue=2*w^3-3*w^2+2;
else
    gnwvalue=1;
```

```
end
end
%-----------------------------------------------------------------------
---
function gnwprimevalue=gnwprime(w)
if w<1
%      if w<0
%         error('w is negative')
%     end
    gnwprimevalue=6*w^2-6*w;
else
    gnwprimevalue=0;
end
end
function [T1,x] = firstfunction(b0,n0,w0,t,tt,u,params)
    ic= [b0 n0 w0];
    options = odeset('RelTol',1e-4,'AbsTol',[1e-4, 1e-4, 1e-4]);
    [T1,x]= ode15s(@firstfunctionode, t,ic,options,tt,u,params);
end
function dx = firstfunctionode(t,x,tt,u,params)
 kb=params(1);
 knr=params(2);
 delta=params(3);
 lambdarb=params(4);
 kw=params(5);
 lambdab=params(6);
 kp=params(7);
 lambdap=params(8);
 kni=params(9);
 lambdani=params(10);
 lambdan=params(11);
 beta=params(12);
 gamma=params(13);
 lambdaw=params(14);
 lambdabw=params(15);
 lambdanw=params(16);
 e=params(17);
u1=interp1(tt,u,t,'cubic');
dx = [kb*x(1)*(1-x(1))-
x(1)*(knr*x(2)+delta)/(lambdarb*x(1)+1)*x(3)/(x(3)+kw)-lambdab*x(1);...
   kp*exp(-lambdap*t)*(1-x(2))+(kni*x(1)*x(2)*(1-
x(2))*gnw(x(3)))/(lambdani*x(2)+1)-lambdan*x(2)/(1+e*x(1));...
   beta + gamma*u1-lambdaw*x(3)-lambdabw*x(1)*x(3)-lambdanw*x(2)*x(3)];
end

function [T2,y] = secondfunction(b,n,w,t,tt,params)
ic=[0 0 0];
options = odeset('RelTol',1e-4,'AbsTol',[1e-4, 1e-4, 1e-4]);
[T2,y] = ode15s(@secondfunctionode,t,ic,options,tt,b,n,w,params);
end

function dy = secondfunctionode(t,y,tt,b,n,w,params)
 kb=params(1);
 knr=params(2);
 delta=params(3);
 lambdarb=params(4);
 kw=params(5);
 lambdab=params(6);
 kp=params(7);
 lambdap=params(8);
 kni=params(9);
 lambdani=params(10);
 lambdan=params(11);
 beta=params(12);
 gamma=params(13);
 lambdaw=params(14);
 lambdabw=params(15);
 lambdanw=params(16);
 e=params(17);
ttt = flipud(tt')';
b1=interp1(ttt,b,t,'cubic');
n1=interp1(ttt,n,t,'cubic');
w1=interp1(ttt,w,t,'cubic');
dy = [-(1+y(1)*(kb-2*kb*b1-lambdab+((knr*n1+delta)*b1*lambdarb-
(lambdarb*b1+1)*(knr*n1+delta))/((lambdarb*b1+1)^2)*w1/(w1+kw))+y(2)*((
(kni*n1*(1-
n1)*(gnw(w1)))/(lambdani*n1+1)+(lambdan*n1*e)/((1+e*b1)^2))+y(3)*(-
lambdabw*w1)));...
    -((y(1)*(-
b1*knr)/(lambdarb*b1+1)*w1/(w1+kw)+y(2)*(gnw(w1)*((lambdani*n1+1)*(kni*
b1-2*kni*b1*n1)-kni*b1*n1*(1-n1)*lambdani))/(lambdani*n1+1)^2)-
lambdan/(1+e*b1)-kp*exp(-lambdap*t)+y(3)*(-lambdanw*w1));...
    -(y(1)*((-
b1*(knr*n1+delta))/(lambdarb*b1+1)*(kw)/((w1+kw)^2))+y(2)*((kni*b1*n1*(
1-n1)*(gnwprime(w1)))/(lambdani*n1+1))+y(3)*(-lambdaw-lambdabw*b1-
lambdanw*n1))];

end
function delta2=gaussian2()
%t=18:0.01:30;
%delta2=0.00001;
%epsilon=0.01;
%tau=24:48:60;
%y=exp(-delta2*((t-tau)/epsilon).^2);
%figure(11)
%plot(t./24,y)
 %t1=0:0.01:24;
%y=exp(-delta2*((t1-tau)/epsilon).^2);
%figure(12)
%plot(t1./24,y)
%delta2 = find(y > 0.99999999999);
%delta1=0.006;
%epsilon1=0.01;
%t2=0:0.001:2;
%y2=exp(-delta1*((t2-tau)/epsilon1).^2);
%plot(t2./24,y)
%function delta2=gaussian2()
t=18:0.01:30;
delta=0.00001;
epsilon=0.01;
y=exp(-delta*((t-24)/epsilon).^2)+ exp(-delta*((t-48)/epsilon).^2+exp(-
delta2*((t-72)/epsilon1).^2));
figure(11)
```

```
plot(t./24,y)
t1=0:0.01:48;
y=exp(-delta*((t1-24)/epsilon).^2);
delta1=0.006;
epsilon1=0.01;
t2=0:0.001:2;
y2=exp(-delta1*((t2-1)/epsilon1).^2) + exp(-delta1*((t2-
2)/epsilon1).^2)
figure(12)
plot(t./24,y,'-k',t2,y2,'-b')
delta2=y2(find(y2 > 0.99999999999999));
delta2=1.2;
end
```

Figure 11: Code for the non-linear problem.

```
function y = linearproblem21(b0,n0,w0,A,B,C,D,epsilon)% assigns
variable n0,b0 and w0 as
%
% Inputs:
% b0 = Initial bacteria level (0 - 1)
% n0 = Initial neutrophil level (0 - 1)
% w0 = Initial oxygen level (0 - 1)
% A = Scalar value (70, 80, 100)
% B = Scalar value (6, 8, 5)
% C = Scalar value (5, 3, 2)
% D = Scalar value (5, 3, 2)
% Epsilon = Small number (0.0001 - 0.01)
%
% Output:
% Graphs

test = -1; % convergence test variable- begins the while loop with a
neg
% number
tf=14;
zeta = .00001; %convergence tolerance requirement
N = 1000; % number of nodes
t = linspace(0,tf,N+1); % creates N+1-1=1000 equally spaced nodes
t1 = linspace(tf,0,N+1);
h = tf/N; % spacing is assigned as h
%M1 = 0;  not used except for lines 87-91
M2 = 2;
M=1.5;
%u = zeros(1,N+1); % initial guess for u where u_i=0
%u = ones(1,N+1);
u = -M2*t/tf +M2;
%u = 0.5;

%u(1) = 0;
u1 = u;
n = zeros(1,N+1); % vector n and size
```

```
n(1) = n0; % initial condition for n because matlab recognizes 1 as the
% first element
% n(N+1) = n;
b = zeros(1,N+1); % vector b and size
b(1) = b0; % initial condition for b
w = zeros(1,N+1); % vector w and size
w(1) = w0; % initial condition for w
lambda1 = zeros(1,N+1); % lamda1 and size
lambda2 = zeros(1,N+1);
lambda3 = zeros(1,N+1);

utwo = zeros(1,N+1);
uthree = zeros(1,N+1);
u2 = utwo;
u3 = uthree;
%x1exact = zeros(1,N+1);
%x2exact = zeros(1,N+1);
%uexact = zeros(1,N+1);
k=0;
j=0;
while(test < 0 && k<5) % when convergence occurs test will become non-
negative
    k = k+1;


    c = 0.1;
    kb = 14.256;
    knr = 2*A;
    delta = 3.84*B;
    lambdarb = 3.73;
    kw = .75;
    lambdab = 0.14256;
    kp = 0.052;
    lambdap = 3.04;
    kni = 10.28;
    lambdani = 80;
    lambdan = .1728;
    gamma = 1;
    lambdaw = 1.0656;
    beta = 0.7992;
    lambdabw = 12.6593/C;
    lambdanw = 25.5744/D;
    e = 100;
    delta2 = 0.006;
    epsilon1 = 0.01;

params1=[kb,knr,delta,lambdarb,kw,lambdab,kp,lambdap,kni,lambdani,lambd
an,beta,gamma,lambdaw,lambdabw,lambdanw,e];

 %reorder for consistency
     [T1,x]=firstfunction(b0,n0,w0,t,t,u1,params1,u2,u3,epsilon);
     if t~=T1'
         error('time values for x do not match')
     end
     b=x(:,1)';
     n=x(:,2)';
     w=x(:,3)';
    [T2,lambda]=secondfunction(b,n,w,t1,t1,params1);
    if t~=T2'
       error('time values for lambda do not match')
    end
   lambda1=flipud(lambda(:,1))';
   lambda2=flipud(lambda(:,2))';
   lambda3=flipud(lambda(:,3))';
end

   figure(12)
%display
%title(figure(12),'summation of Gaussiana')

   plot(t,exp(-delta2*((t-1)/epsilon1).^2) + exp(-delta2*((t-
2)/epsilon1).^2) ...
+exp(-delta2*((t-3)/epsilon1).^2) +exp(-delta2*((t-4)/epsilon1).^2)+
exp(-delta2*((t-5)/epsilon1).^2)+ ...
exp(-delta2*((t-6)/epsilon1).^2)+ ...
    exp(-delta2*((t-7)/epsilon1).^2)+exp(-delta2*((t-8)/epsilon1).^2)+
...
   exp(-delta2*((t-9)/epsilon1).^2) +exp(-delta2*((t-10)/epsilon1).^2)
...
   +exp(-delta2*((t-11)/epsilon1).^2) ...
    +exp(-delta2*((t-12)/epsilon1).^2)+exp(-delta2*((t-
13)/epsilon1).^2) ...
    +exp(-delta2*((t-14)/epsilon1).^2));
title('Summation of Gaussians')
xlabel({'Time' 'in days'})
%
     u1 =
(((M/2)*(c+gamma*lambda3)./sqrt(((c+gamma*lambda3).^2)+(epsilon^2*lambd
a1.^2)+(epsilon^2*lambda2.^2)))+(M/2));
     %u2 =
(epsilon*lambda1)./sqrt(((c+gamma*lambda3).^2)+(epsilon^2*lambda1.^2)+(
epsilon^2*lambda2.^2));
     %u3 =
(epsilon*lambda2)./sqrt(((c+gamma*lambda3).^2)+(epsilon^2*lambda1.^2)+(
epsilon^2*lambda2.^2)) ;

%      u1 = real(max(M1,min(u1-h*(c+gamma*lambda3),M2)));
%      u1t = 2/(M2-M1)*u1-(M2+M1)/(M2-M1);
%      u2 = real(max(M1*sqrt(1-u1t.^2),min(u2-
h*epsilon*lambda1,M2*sqrt(1-u1t.^2))));
%      u2t = 2/(M2-M1)*u2-(M2+M1)/(M2-M1);
%      u3 = real(max(M1,min(u3-h*epsilon*lambda2,M2*sqrt(1-u1t.^2-
u2t.^2))));


     u1 = u1.*(exp(-delta2*((t-1)/epsilon1).^2)+exp(-delta2*((t-
2)/epsilon1).^2)+ ...
         exp(-delta2*((t-3)/epsilon1).^2)+ ...
         exp(-delta2*((t-4)/epsilon1).^2)+exp(-delta2*((t-
5)/epsilon1).^2)+ ...
         exp(-delta2*((t-6)/epsilon1).^2)+exp(-delta2*((t-
7)/epsilon1).^2)+ ...
```

```
         exp(-delta2*((t-8)/epsilon1).^2)+exp(-delta2*((t-
9)/epsilon1).^2)+ ...
         exp(-delta2*((t-10)/epsilon1).^2)+ exp(-delta2*((t-
11)/epsilon1).^2)+ ...
         exp(-delta2*((t-12)/epsilon1).^2)+exp(-delta2*((t-
13)/epsilon1).^2)+ ...
         exp(-delta2*((t-14)/epsilon1).^2))
     ...
         -h*(c+gamma*lambda3);
     u2 = u2-h*epsilon*lambda1;
     u3 = u3-h*epsilon*lambda2;
    oldj = j;
    j = sum(b+(c/2)*u1);
    %temp2 =

%(sum(((c+gamma*lambda3).^2+(epsilon*lambda1).^2+(epsilon*lambda2).^2).
^.5))/N;
    %un-comment temp2 if the display line, line 106, is uncommented
    temp = abs(j-oldj);
    test = zeta-temp;
    if floor(k/10)==k/10
        display(temp)
        %display([temp11,temp21,temp22,temp31,temp23,temp33,temp32]);

end

display(temp)

y(1,:) = t; % defines t
y(2,:) = n; % defines n
y(3,:) = b; % defines b
y(4,:) = w; % defines w
y(5,:) = lambda1; % defines lambda1
y(6,:) = lambda2; % defines lambda2
y(7,:) = lambda3; % defines lambda3
y(8,:) = u1; % defines u
%sum(b)
%sum(u1.^2)
J=sum(b+(c/2)*u1);
display(J)
figure(1)
hold on;
subplot(7,1,1)
plot(t,b,'r-')
ylabel('b')
title('Optimal control results for b, n, w, u(t),\lambda_1, \lambda_2,
\lambda_3')
subplot(7,1,2)
plot(t,n,'r-')
ylabel('n')
subplot(7,1,3)
plot(t,w,'r-')
ylabel('w')
subplot(7,1,4)
plot(t,u1,'r-')
ylabel('u(t)')
```

```
axis([0 14 0 2])%axis([x-min x-max y-min y-max])
%xlabel('t (in days)')


%figure(2)
%hold on;

subplot(7,1,5)
plot(t,lambda1,'r-')
ylabel('\lambda_1')
%title('Optimal control results for \lambda_1, \lambda_2, \lambda_3')
subplot(7,1,6)
plot(t,lambda2,'r-')
ylabel('\lambda_2')
subplot(7,1,7)
plot(t,lambda3,'r-')
ylabel('\lambda_3')
xlabel('t (in days)')

figure(2)
hold on;
subplot(2,1,1)
plot(t,u2,'r-')
ylabel('u2')
subplot(2,1,2)
plot(t,u3,'r-')
ylabel('u3')
xlabel('t (in days)')

%display('k =')
display(k)
end
%----------------------------------------------------------------------
---
function gnwvalue=gnw(w)

if w<1
    if w<0
        error('w is negative')
    end
    gnwvalue=2*w^3-3*w^2+2;
else
    gnwvalue=1;
end
end
%----------------------------------------------------------------------
---
function gnwprimevalue=gnwprime(w)
if w<1
    if w<0
        error('w is negative')
    end
    gnwprimevalue=6*w^2-6*w;
else
    gnwprimevalue=0;
end
```

```
end
%-----------------------------------------------------------------------
---
function [T1,x] = firstfunction(b0,n0,w0,t,tt,u,params,u2,u3,epsilon)
ic= [b0 n0 w0];
options = odeset('RelTol',1e-4,'AbsTol',[1e-4, 1e-4, 1e-4]);
[T1,x]= ode15s(@firstfunctionode,
t,ic,options,tt,u,params,u2,u3,epsilon);
end
%-----------------------------------------------------------------------
---
function dx = firstfunctionode(t,x,tt,u,params,u2,u3,epsilon)
 kb=params(1);
 knr=params(2);
 delta=params(3);
 lambdarb=params(4);
 kw=params(5);
 lambdab=params(6);
 kp=params(7);
 lambdap=params(8);
 kni=params(9);
 lambdani=params(10);
 lambdan=params(11);
 beta=params(12);
 gamma=params(13);
 lambdaw=params(14);
 lambdabw=params(15);
 lambdanw=params(16);
 %e=params(17);
 u
u1=interp1(tt,u,t,'cubic');
u1
u21=interp1(tt,u2,t,'cubic');
u31=interp1(tt,u3,t,'cubic');
dx = [kb*x(1)*(1-x(1))-
x(1)*(knr*x(2)+delta)/(lambdarb*x(1)+1)*x(3)/(x(3)+kw)-
lambdab*x(1)+epsilon*u21;...
kp*exp(-lambdap*t)*(1-x(2))+(kni*x(1)*x(2)*(1-
x(2))*gnw(x(3)))/(lambdani*x(2)+1)-
lambdan*x(2)/(1+exp(1)*x(1))+epsilon*u31;...
beta + gamma*u1-lambdaw*x(3)-lambdabw*x(1)*x(3)-lambdanw*x(2)*x(3)];

end
%-----------------------------------------------------------------------
---
function [T2,y] = secondfunction(b,n,w,t,tt,params)
ic=[0 0 0];
options = odeset('RelTol',1e-4,'AbsTol',[1e-4, 1e-4, 1e-4]);
[T2,y] = ode15s(@secondfunctionode,t,ic,options,tt,b,n,w,params);
end
%-----------------------------------------------------------------------
---
function dy = secondfunctionode(t,y,tt,b,n,w,params)
 kb=params(1);
 knr=params(2);
 delta=params(3);
 lambdarb=params(4);
 kw=params(5);
 lambdab=params(6);
 kp=params(7);
 lambdap=params(8);
 kni=params(9);
 lambdani=params(10);
 lambdan=params(11);
 %beta=params(12);
 %gamma=params(13);
 lambdaw=params(14);
 lambdabw=params(15);
 lambdanw=params(16);
 e=params(17);
ttt = flipud(tt')';
b1=interp1(ttt,b,t,'cubic');
n1=interp1(ttt,n,t,'cubic');
w1=interp1(ttt,w,t,'cubic');

dy = [-(1+y(1)*(kb-2*kb*b1-lambdab+((knr*n1+delta)*b1*lambdarb-
(lambdarb*b1+1)*(knr*n1+delta))/((lambdarb*b1+1)^2)*w1/(w1+kw))+y(2)*((
(kni*n1*(1-
n1)*(gnw(w1)))/(lambdani*n1+1)+(lambdan*n1*e)/((1+e*b1)^2))+y(3)*(-
lambdabw*w1)));...
    -((y(1)*(-
b1*knr)/(lambdarb*b1+1)*w1/(w1+kw)+y(2)*(gnw(w1)*((lambdani*n1+1)*(kni*
b1-2*kni*b1*n1)-kni*b1*n1*(1-n1)*lambdani))/(lambdani*n1+1)^2)-
lambdan/(1+e*b1)-kp*exp(-lambdap*t)+y(3)*(-lambdanw*w1));...
    -(y(1)*((-
b1*(knr*n1+delta))/(lambdarb*b1+1)*(kw)/((w1+kw)^2))+y(2)*((kni*b1*n1*(
1-n1)*(gnwprime(w1)))/(lambdani*n1+1))+y(3)*(lambdaw-lambdabw*b1-
lambdanw*n1))];
end
```

Figure 12: The code for the linear problem.

References

1. Donna, L, Daulton, Western Kentucky Masters Thesis, May 2013 Using Optimal Control Theory to Optimize the Use of Oxygen Therapy in Chronic Wound

2. Brown, P., Maloy, J.P., Oddo, D. 2001 Quick Reference to Wound Care. *Aspen Publications: Gaithersburg, MD*

3. Lenhart, S., Workman, J.T. 2007 Optimal control applied to biological models. *Chapman & Hall/CRC: Boca Raton, FL.*

4. Lukes, D.L. 1982 Differential Equations: classical to controlled. *Academic Press: New York, NY.*

5. Schugart, R. C., Friedman, A., Zhao, R., Sen, C. K. 2008 Wound angiogenesis as a function of tissue oxygen tension: a mathematical model. *Proceedings of the Natural Academy of Science USA 105(7): 2628-2633.*

6. Bartle, R., Sherbert, D. 1927 Introduction to Real Analysis, 3rd Ed. *John Wiley & Sons, Inc.: New York, NY*

7. Chronic Wounds. (n.d.). Retrieved March 22, 2016, from http://www.woundcarecenters.org/article/wound-types/chronic-wounds

8. Weisstein, Eric W. "Convex Function." From *MathWorld*--A Wolfram Web Resource. http://mathworld.wolfram.com/ConvexFunction.html

9. Rudin, W. *Principles of Mathematical Analysis, 3rd ed.* New York: McGraw-Hill, 1976

10. Eckhoff, J. "Helly, Radon, and Carathéodory Type Theorems." Ch. 2.1 in *Handbook of Convex Geometry* (Ed. P. M. Gruber and J. M. Wills). Amsterdam, Netherlands: North-Holland, pp. 389-448, 1993.